Optimierung in der Flugplanung:

Netzwerkentwurf und Flottenzuweisung

Dissertation

zur Erlangung des akademischen Grades
eines Doktors der Naturwissenschaften
(Dr. rer. nat.)
im Fach Informatik

eingereicht an der
Fakultät für Elektrotechnik, Informatik und Mathematik
Universität Paderborn

von
Herrn Dipl.-Inform. Georg Kliewer

Tag der mündlichen Prüfung: 26. August 2005

Mitglieder der Prüfungskommission:

- Prof. Dr. Burkhard Monien (Vorsitzender, Gutachter)
- Prof. Dr. Leena Suhl (Gutachter)
- Prof. Dr. Wilfried Hauenschild
- Dr. Peter Pfahler
- Jun. Prof. Dr. Christian Sohler

Danksagungen

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter an der Universität Paderborn. Mein besonderer Dank gilt Prof. Dr. Burkhard Monien für die Betreuung der Arbeit. Er schafft in seiner Arbeitsgruppe eine hervorragende Forschungsumgebung und motiviert seine Mitarbeiter stets, neue wissenschaftliche Herausforderungen anzunehmen.

Die Arbeit wurde im Rahmen eines von der Deutschen Forschungsgemeinschaft geförderten Projekts im Schwerpunktprogramm "Algorithmik großer und komplexer Netzwerke" und einer Kooperation mit der Firma Lufthansa Systems durchgeführt.

Ich möchte mich bei allen Kollegen in der Arbeitsgruppe und dem Paderborn Center for Parallel Computing (PC^2) für die gute Zusammenarbeit und die vielen interessanten Diskussionen bedanken. Hervorheben möchte ich die Kollegen, mit denen ich eng wissenschaftlich zusammengearbeitet habe: Torsten Fahle, Silvia Götz, Sven Grothklags, Achim Koberstein, Meinolf Sellmann und Larissa Timajev. Stefan Tschöke war mit seinen Visionen und seiner kreativen Energie immer ein Vorbild für mich.

Weiterer Dank gilt den Mitarbeitern der Lufthansa Systems in Frankfurt und Berlin: Georg F. Bolz, Klaus-Peter Keilmann, Michael Lefeld und Klaus Weber für die langjährige Unterstützung der Paderborner Aktivitäten im Bereich Flugplanung. Mein Dank geht auch nach Montréal an Bernard Gendron und nach Pisa an Antonio Frangioni für eine fruchtbare Zusammenarbeit in den letzten Jahren.

Ich möchte mich auch bei Prof. Dr. Leena Suhl für die Begutachtung dieser Dissertation bedanken. Bei Torsten Fahle und Sven Grothklags bedanke ich mich zusätzlich für das Korrekturlesen der Arbeit.

Schließlich möchte ich mich bei Natalia, Jan und den vielen Freunden für ihre Unterstützung und Geduld in den letzten Jahren bedanken – vielen herzlichen Dank!

Paderborn, im Juli 2005 Georg Kliewer

Inhaltsverzeichnis

Verzeichnis der Algorithmen

1

Einleitung

1.1 Motivation

Der Luftverkehr bildet heute eine wichtige Grundlage für die Mobilität der Menschen überall in der Welt. Seine Bedeutung ist in den letzten Jahrzehnten stark gewachsen. Die Statistiken der IATA (International Air Transport Association) belegen eine stetige Zunahme des Passagierflugverkehrs (Abbildung 1.1). Die Ereignisse der letzten fünf Jahre wie die Terroranschläge vom September 2001, Militärkonflikte, die Infektionskrankheit SARS, die extrem hohen Rohölpreise haben diese Wachstumstendenz deutlich gebremst. Die Fluggesellschaften sehen sich in einem hart umkämpften Markt. Bedingt durch hohe fixe Kosten eines Fluges und zurückgehende Einnahmen müssen die meisten Fluggesellschaften trotz Kapazitätsanpassungen hohe Verluste vermelden. Diese haben bereits dazu geführt, dass einige Fluggesellschaften ihren Betrieb einstellen mussten oder von anderen übernommen wurden.

Die Strategie der Fluggesellschaften kann in dieser Situation vor allem darin liegen, die Kosten des Flugbetriebs zu senken. Die Planung des Einsatzes der Ressourcen, in erster Linie der Flugzeuge und der Crews, spielt eine zentrale Rolle bei der Frage der Kosteneffizienz.

Die Planung bei einer Fluggesellschaft wird von mehreren Expertenteams durchgeführt, die die unterschiedlichsten Aufgaben lösen müssen. Der Zeithorizont erstreckt sich über mehrere Monate bis Jahre vor Beginn einer Flugplanperiode (Sommer/Winter-Flugplan). Die Planer profitieren bei ihrer Arbeit stark von der Hilfe der Entscheidungsunterstützungssysteme (*Decision Support Systems*). Diese Systeme stellen die benötigten Informationen zur Verfügung und ermöglichen so einen reibungslosen Ablauf der Planungsprozesse. Einen entscheidenden Beitrag leisten die in diesen Systemen verwendeten Optimierungskomponenten. Sie berücksichtigen komplexe Restriktionen und liefern kostenminimale Lösungen für Planungsszenarien, die von Experten definiert werden. Auf dieser Grundlage können Entscheidungen getroffen werden, die den Ressourceneinsatz entscheidend verbessern.

In der Mittel- und Kurzfristplanung der Fluggesellschaften müssen auch die Aufgaben des Netzwerkentwurfs und der Flottenzuweisung gelöst werden. Die erste Aufgabe besteht darin, das Flugnetz möglichst optimal aufzubauen. Bei der zweiten Aufgabe werden Flugzeugtypen

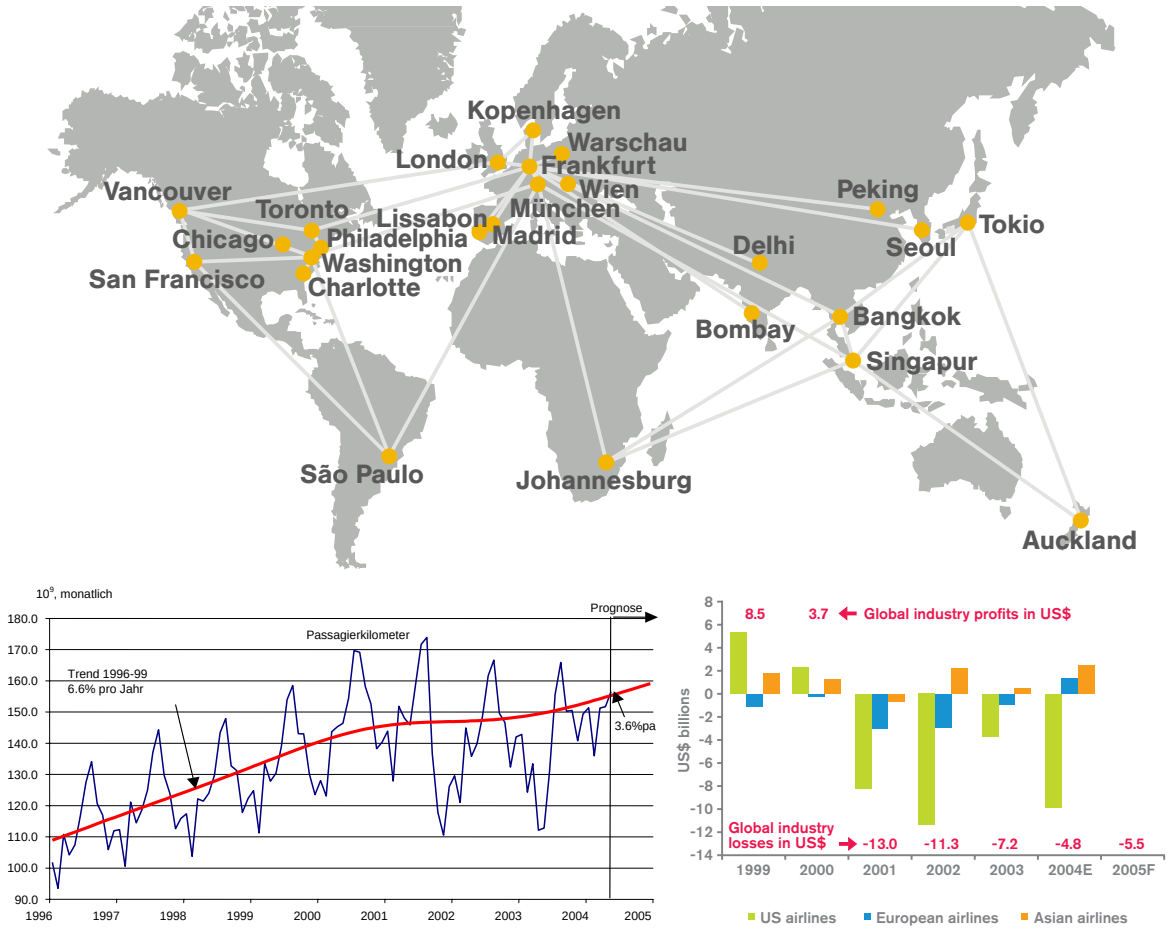


Abbildung 1.1: Internationales Flugnetzwerk, Anzahl der Passagierkilometer und Gewinne und Verluste in der Luftverkehrsbranche (Quelle: StarAlliance, IATA)

für Flugstrecken festgelegt, die auf der einen Seite die Kapazitäten im Netzwerk festlegen und auf der anderen Seite eine der größten Kostenpositionen darstellen. Eine optimale Lösung dieser Probleme kann einen wichtigen Beitrag zur Kostensenkung bei der Fluggesellschaft leisten. Weiterhin erlaubt eine Optimierung über die Grenzen der Planungsphasen hinweg eine bessere Abstimmung zwischen diesen und führt damit zu insgesamt besseren Flugplänen.

1.2 Ziele der Arbeit

Die Zielsetzung dieser Arbeit liegt darin, effiziente Algorithmen zur Lösung der zugrunde liegenden Optimierungsprobleme zu entwickeln, zu untersuchen und in den Entscheidungsunterstützungssystemen einzusetzen.

Das Problem des Netzwerkentwurfs stellt ein schwieriges ganzzahliges Optimierungsproblem dar. Von der Forschungsgemeinde wurden dafür mehrere heuristische und exakte Lösungsverfahren entwickelt. Die Laufzeiten der Algorithmen und die Qualität der Lösungen können aber in vielen Fällen verbessert werden. In dieser Arbeit wird ein Beitrag dazu geleistet, die Leistungsfähigkeit der vorhandenen Algorithmen zu steigern. Es werden weiterhin

neue algorithmische Ideen entwickelt und untersucht.

Die Aufgabe der Flottenzuweisung stellt eine der Aufgaben in der Prozesskette dar. Sie wird in vielen Fällen noch getrennt von anderen Planungsaufgaben betrachtet, obwohl sie sehr eng mit den anderen verzahnt ist. Zwei dieser Aufgaben - die Marktmodellierung und das Ertragsmanagement (Revenue Management) - werden im Rahmen dieser Arbeit untersucht und mit der Phase der Flottenzuweisung integriert. Durch diese Integration können die Qualität der gelieferten Lösungen und die Effizienz der gesamten Prozesskette deutlich verbessert werden.

1.3 Aufbau der Arbeit

Die Arbeit beginnt mit einem Überblick über die Planungsprozesse bei einer Fluggesellschaft. Die zu lösenden Aufgaben werden beschrieben, wobei der Schwerpunkt auf den Aufgaben des Netzwerkentwurfs, der Marktmodellierung, der Flottenzuweisung und des Revenue Managements liegt.

Im **Kapitel 2** wird das grundlegende Optimierungsproblem des Netzwerkentwurfs beschrieben, es werden Modelle aufgebaut und Lösungsverfahren untersucht. Die wesentlichen Komponenten des Branch-and-bound Algorithmus - untere und obere Schranken, Verfahren zum Lösen der Lagrange-Relaxationen in den Knoten des Suchbaums, Verzweigungsstrategien und Strategien zur Variablenfixierung - werden in diesem Kapitel vorgestellt. Wir untersuchen anschließend zusätzliche Ungleichungen, die die Qualität der unteren Schranken verbessern können. Der Relax-and-cut-Algorithmus basiert auf den Lagrange-Relaxationen, die um die zusätzlichen Ungleichungen erweitert werden. Hauptergebnis des Kapitels stellt das entwickelte Optimierungssystem für den Netzwerkentwurf dar. Der Aufbau des Systems aus unterschiedlichen algorithmischen Komponenten erlaubt einen flexiblen Einsatz des Systems in unterschiedlichen Anwendungsfällen.

Im **Kapitel 3** werden die Planungsphasen der Marktmodellierung, der Flottenzuweisung und des Revenue Managements vorgestellt. Anschließend werden die Schwachstellen der Planungsprozesskette identifiziert und das Potential der Integration der Aufgaben untersucht. Wir entwickeln drei Integrationsstrategien, die die unterschiedlichen Aspekte der Abhängigkeiten berücksichtigen und dadurch eine verbesserte Planung ermöglichen.

Das **Kapitel 4** beschreibt die experimentellen Untersuchungen des Optimierungssystems für Netzwerkentwurf. Zunächst werden im **Abschnitt 4.1** die benutzten Datensätze vorgestellt. Eine erste Reihe von Experimenten vergleicht unser System mit anderen Lösungsverfahren für das Problem des Netzwerkentwurfs, die von anderen Wissenschaftlern entwickelt wurden. Im zweiten Teil der Experimente untersuchen wir einzelne wichtige Komponenten des Systems, um die entwickelten Algorithmen zu bewerten. Das Kapitel schließt mit einer Zusammenfassung der gewonnenen Erkenntnisse.

Im **Abschnitt 4.2** stellen wir die benutzten Szenariodaten einiger großer europäischer und amerikanischer Fluggesellschaften zu den Aufgaben der Marktmodellierung, Flottenzuweisung und des Revenue Managements dar. Wir untersuchen die Eigenschaften der Datensätze, die für die Integration wichtig sind. Anschließend präsentieren wir einige ausgewählte Experimente, die die Leistungsfähigkeit der Verfahren untersuchen.

Im **Kapitel 5** werden die Ergebnisse der Arbeit zusammengefasst und ein Ausblick für die weitere Forschung gegeben.

1.4 Ausgewählte Publikationen

Die Ergebnisse dieser Arbeit wurden in folgenden Publikationen veröffentlicht:

Internationale Zeitschriften und begutachtete Konferenzen:

- Georg Kliewer and Larissa Timajev. Relax-and-cut for capacitated network design. In Proceedings of the 13th Annual European Symposium on Algorithms (ESA-2005), Springer, LNCS 3669, pages 47-58, Palma der Mallorca, Spain, 2005.
- Weber K.; Sun J.; Sun Z.; Kliewer G.; Grothklags S.; Jung N. Systems integration for revenue-creating control processes. Journal of Revenue and Pricing Management, July 2003, vol. 2, no. 2, pp. 120-137(18) Henry Stewart Publications.
- Georg Kliewer, Sven Grothklags, and Klaus Weber. Improving revenue by system integration and co-operative optimization. In Proceedings of the 43rd Annual Symposium of the Airline Group of the International Federation of Operational Research Societies (AGIFORS-2002), Honolulu, Hawaii, USA, 2002.
- Meinolf Sellmann, Georg Kliewer, and Achim Koberstein. Lagrangian cardinality cuts and variable fixing for capacitated network design. In Proceedings of the 10th Annual European Symposium on Algorithms (ESA-2002), Springer, LNCS 2461, pages 845-858, Rome, Italy, 2002.
- Georg Kliewer. Integrating market modeling and fleet assignment. In Proceedings of the 2nd international Workshop on the Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CP-AI-OR'00), Paderborn, Germany, 2000.

Internationale Konferenzen und Workshops:

- Georg Kliewer. A Branch-and-Cut System for Capacitated Network Design. In Proceedings of the 18th International Symposium on Mathematical Programming (ISMP-2003), Copenhagen, Denmark, 2003.
- Georg Kliewer, Achim Koberstein, and Meinolf Sellmann. Capacitated network design. In Proceedings of the the sixteenth triennial conference of the International Federation of Operational Research Societies (IFORS-2002), Edinburgh, Scotland, UK, 2002.
- Georg Kliewer and Achim Koberstein. Solving the capacitated network design problem in parallel. In Proceedings of the third meeting of the EURO-PAREO working group on Parallel Processing in Operations Research (PAREO), Guadeloupe, France, 2002.
- Georg Kliewer. Network design: Modeling and solving in an airline alliance context. In Proceedings of the 14. Meeting of the European Chapter on Combinatorial Optimization (ECCO XIV), Bonn, Germany, 2001.
- Georg Kliewer. Cooperative approaches for market modeling and fleet assignment. In Proceedings of the 17th International Symposium on Mathematical Programming (ISMP-2000), Atlanta, GA, USA, 2000.

- Silvia Götz, Sven Grothklags, Georg Kliewer, and Stefan Tschöke. Solving the weekly fleet assignment problem for large airlines. In Proceedings of the Third Metaheuristics International Conference (MIC-1999), pages 241-246, Angra dos Reis, Brasil, 1999.

1.5 Grundlegende Literatur

In dieser Arbeit werden unterschiedliche algorithmische Konzepte und Optimierungsverfahren vorgestellt und untersucht. Die meisten Erklärungen sind in sich abgeschlossen. Zum tieferen Verständnis der benutzten Konzepte der linearen (ganzzahligen) Optimierung, Lagrange-Relaxation, Column Generation, Subgradientenverfahren, Bundle-Methoden, Flussalgorithmen etc. kann folgende Literatur empfohlen werden:

- R. K. Ahuja, T. L. Magnati, and J.B. Orlin. Network Flows. Prentice Hall, 1993.
- T. H. Cormen, C. E. Leiserson, and R. L. Rivest. Introduction to Algorithms. The MIT Press, 1990.
- G. L. Nemhauser and L. A. Wolsey. Integer and Combinatorial Optimization. Wiley, 1988.
- G. L. Nemhauser, A. H. G. Rinnooy Kan, M. J. Todd. Optimization. Elsevier Science. 1989.
- C. Papadimitriou and K. Steiglitz. Combinatorial Optimization. Prentice Hall, 1982.
- L. A. Wolsey. Integer Programming. Wiley, 1998.

— 2 —

Netzwerkentwurf

Dieses Kapitel beginnt mit einem Überblick über die Planung innerhalb einer Fluggesellschaft. Der Schwerpunkt liegt beim Problem des Netzwerkentwurfs. Im Hauptteil des Kapitels beschreiben wir das im Rahmen dieser Arbeit entwickelte Optimierungssystem für den Netzwerkentwurf. Die algorithmischen Komponenten und Strategien werden detailliert vorgestellt und untersucht. Das Kapitel schließt mit einem Überblick über das Gesamtsystem im Abschnitt 2.14.

2.1 Prozess der Flugplanung

In diesem Abschnitt wird ein Überblick über die Planungsaufgaben innerhalb einer Fluggesellschaft gegeben (siehe Abbildung 2.1).

Der Betrieb einer Fluggesellschaft erfordert die effektive Planung und das Management des Flugnetzes. Die angebotene Transportleistung muss an die Kunden verkauft werden. Diese Aufgaben werden vom Marketing und Revenue Management wahrgenommen. Fliegendes Personal muss möglichst optimal eingesetzt werden. In jedem dieser Bereiche sind unterschiedliche Zielsetzungen gegeben, die über die gesamte Planungsdauer variieren können.

In der Langfristplanung, 3 - 1 Jahr vor dem Start des Flugbetriebs, kann die Maximierung des Gewinns aus dem Betrieb des Flugnetzwerks als das wichtigste Ziel identifiziert werden. In der Mittelfristplanung (1 Jahr - 6 Monate) geht es vor allem um eine möglichst optimale Abstimmung zwischen dem Flugplan und den Ressourcen. In der Kurzfristplanung (6 Monate - 4 Wochen) wird der Einsatz der Ressourcen kontinuierlich an die Marktnachfrage angepasst und die entstehenden Kosten werden minimiert. In der Implementierungsphase (ab 4 Wochen vor dem Start) und in der Kontrollphase (ab 2 Wochen) werden Ausfälle und Sonderereignisse unter der Vorgabe der Kostenminimierung behandelt.

Der Prozess der Flugplanung ist in mehrere Schritte unterteilt, die zum Teil zeitlich überlappen. Ergebnisse der jeweiligen Planungsschritte werden an die späteren Phasen in

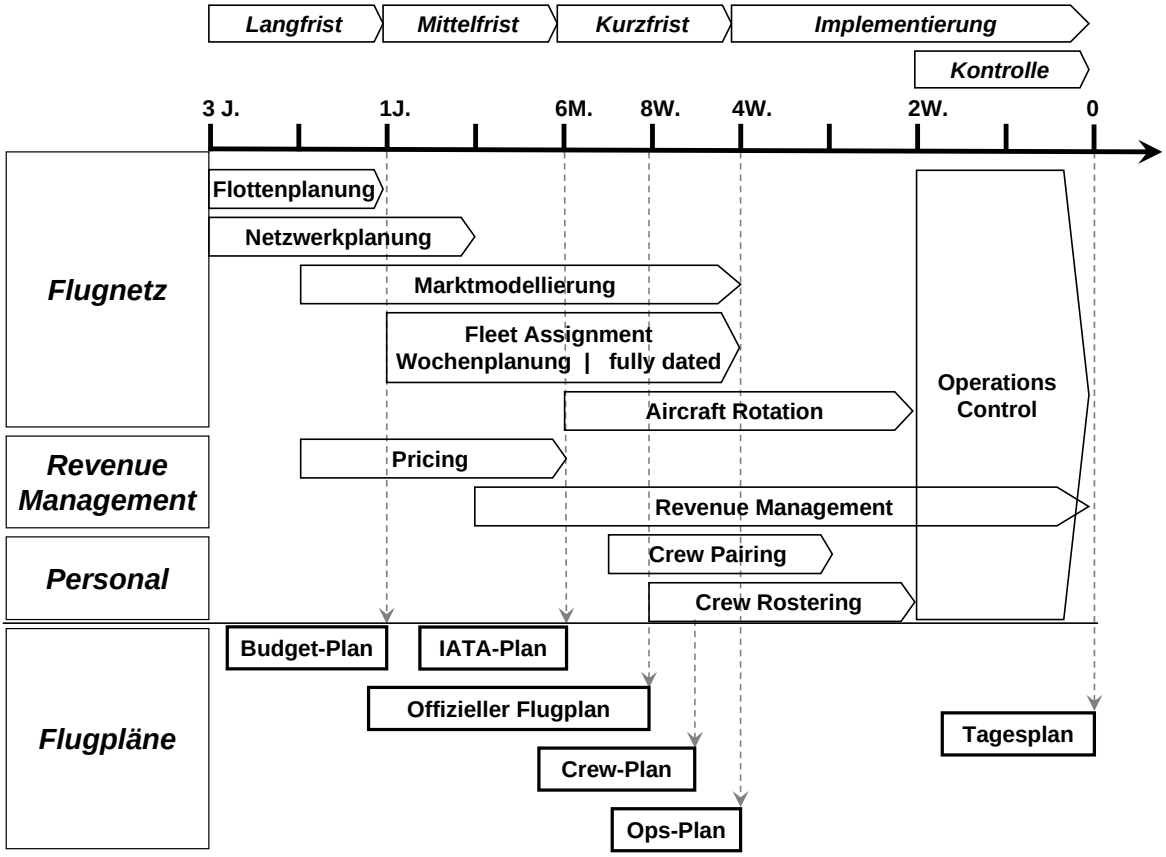


Abbildung 2.1: Übersicht über den Prozess der Flugplanung

unterschiedlichsten Formen weitergegeben. Im unteren Bereich der Abbildung 2.1 sind einige Pläne dargestellt, die im gewissen Sinne Meilensteine im Prozess bilden.

Der Budget-Plan wird 1 Jahr vor dem Start erstellt und definiert die finanzielle Seite des Flugplans für alle Bereiche der Fluggesellschaft. Der IATA-Plan dient als Grundlage für die halbjährliche IATA-Konferenz, bei der Vertreter aller Fluggesellschaften und der Flughäfen zusammenkommen, um über Verkehrs- und Landerechte (*slots*) zu verhandeln. Die Informationen im offiziellen Flugplan werden von Passagieren und Reisebüros zu ihrer Reiseplanung benutzt. Der Crew-Plan wird 6 Wochen vor dem Start erstellt und dient als Grundlage für die verbindliche Zuteilung der Ressourcen (Flugzeuge und Crews). Änderungen können ab diesem Zeitpunkt nur mit relativ hohem Aufwand vorgenommen werden. Der operative Flugplan (4 Wochen vor dem Start) wird zur Flugplanumsetzung benötigt. Kurzfristige Anpassungen sind immer notwendig und werden im Tagesverkehrsplan den Bereichen zur Verfügung gestellt.

Die einzelnen Aufgaben und Planungsphasen werden im Folgenden beschrieben.

Flottenplanung Die Flottenplanung berücksichtigt in erster Linie die strategischen Ziele der Fluggesellschaft: Wachstumsstrategie, angestrebte Marktposition, Zustand der verfügbaren Flotte. Langfristige Nachfrageprognosen bilden die Grundlage für Entscheidungen in dieser Phase. Als Ergebnis werden Flugzeuge geordert, die Flughafenstruktur ausgebaut und

neue Regionen als Flugziele ausgewählt.

Netzwerkplanung Die Planung umfasst in dieser Phase die Definition der anzufliegenden Zielorte, die Anzahl der Flüge zwischen zwei Flughäfen, Flugtage, die Zeiten der Flüge, die Struktur des Netzwerks für die Umsteigeverbindungen (*hub-and-spoke*) und die Verknüpfung von Flügen auf großen Flughäfen (Hub-Struktur). Die Szenarien werden von Planungsabteilungen ausgearbeitet und mit Verfahren der Marktmodellierung bewertet.

Marktmodellierung Die Aufgabe der Marktmodellierung besteht darin, die Nachfrage nach der Transportleistung der Fluggesellschaft vorherzusagen. Die Anzahl der potentiellen Passagiere aus einzelnen Regionen oder Städten wird anhand historischer Daten geschätzt und die Konkurrenzsituation auf Flugstrecken bewertet. Für ein Flugplanszenario wird der erwartete Passagierfluss im eigenen Netzwerk modelliert und auf diese Weise der mögliche Ertrag geschätzt. Diese Profitabilitätsbewertung liefert dem Planer eine Grundlage für weitere Anpassungen des Flugplanszenarios.

Flottenzuweisung (Fleet Assignment) Die Planungsphase der Flottenzuweisung definiert die Kapazitäten im Netzwerk der Fluggesellschaft. Die vorhandene Flotte wird möglichst kostenoptimal eingeplant, um die erwartete Nachfrage zu bedienen. Flugzeugtypen unterscheiden sich in vielen Faktoren, wie z.B. in Kapazität, Reichweite und Kostenstruktur. Zu beachten sind bei der Planung diverse Restriktionen bezüglich der Flugzeuge, Flughäfen, Wartungsanforderungen etc. In der Mittelfristplanung wird auf der Basis einer Standardwoche geplant. In der Kurzfristplanung wird für eine konkrete Zeitperiode geplant (*fully-dated*). Der Planungsstand am Anfang der Periode muss dabei berücksichtigt werden.

Umlaufplanung der Flugzeuge (Aircraft Rotation) In diesem Planungsschritt werden konkrete Flugzeuge für die einzelnen Strecken eingeplant. Die Struktur der Umläufe muss sämtliche Wartungsregeln berücksichtigen sowie die Kosten und die Anzahl der benötigten Flugzeuge minimieren.

Preis- und Konditionenpolitik (Pricing) Fluggesellschaften verkaufen die Plätze auf Flugstrecken nicht zu einem Einheitspreis. Vielmehr werden unterschiedlichen Kundengruppen (Geschäftsreisende, Freizeitreisende) differenzierte Produkte angeboten, mit unterschiedlichen Konditionen, wie z.B. Vorausbuchungsfristen und Mindestaufenthalten. Die Definition der Buchungsklassen und der zugehörigen Preise ist die Aufgabe dieses Planungsschrittes.

Ertragsmanagement (Revenue Management) Unter Revenue Management wird die Steuerung der Sitzplatzverfügbarkeit verstanden. Die spezifischen Eigenschaften eines Sitzes auf einer Flugstrecke sind: niedrige variable und hohe fixe Kosten, keine Lagerfähigkeit, schwankende Nachfrage, Knappheit der Ressource. In dieser Situation muss versucht werden, jeden Platz mit einem möglichst hohen Erlös zu verkaufen. Die Verfahren dazu sind sehr ausgeklügelt und wirksam und haben ihren Weg aus der Airlinebranche in andere Bereiche gefunden, wie z.B. Hotel und Mietwagen.

Crew Pairing Die Erzeugung der Arbeitspläne für fliegendes Personal (Piloten und Kabinenbesatzung) muss komplizierte Dienst- und Ruhe-Regelungen beachten. Es wird eine Menge von Crew-Rotationen (*pairings*) erzeugt, sodass auf jedem Flug das benötigte Personal eingeplant ist und dabei die Personalkosten minimiert werden. Die Flugzeugtypen auf den Strecken sind dabei bereits festgelegt. Die geforderten Eigenschaften der Piloten und der Kabinenbesatzung (z.B. Sprachkompetenzen) sind zu beachten.

Crew Rostering Die Dienstplanzuordnung weist konkreten Personen die zuvor generierten Pairings zu, sodass ein gültiger Monatsdienstplan entsteht. Während in den USA die Dienstpläne nach Seniorität zugeteilt werden, wird in Europa versucht, möglichst viele persönliche Präferenzen unter Einhaltung einer Kostengrenze zu erfüllen.

Operations control Während der Ausführung des Flugplans treten Störungen auf. Flughäfen werden wegen schlechter Witterungsverhältnisse geschlossen oder reduzieren ihre Aufnahmekapazität, Flüge verspäten sich und Personal fällt kurzfristig aus. Auf diese Ereignisse muss schnell reagiert werden, denn Störungen können sich im ganzen Netzwerk ausbreiten. Dem Operations Manager stehen diverse Handlugsalternativen zur Auswahl, wie z.B. Verschiebung von Flugstarts, Einsatz von Ersatzcrews oder Streichung von Flügen. Die Auswirkungen sollten für Passagiere möglichst gering sein, aber auch die Fluggesellschaft ist daran interessiert, möglichst schnell wieder zum Normalbetrieb zurückzukehren.

Literatur über Flugplanung

Zwei Werke von [Sterzenbach and Conrady, 2003] und [Maurer, 2003] stellen betriebswissenschaftliche Lehr- und Handbücher über den Luftverkehr dar. Weitere Publikationen mit Übersichten zum Thema Flugplanung sind: [Barnhart et al., 2003], [Carl and Gesing, 2000], [Emden-Weinert, 1998], [Fahle, 2002], [Klabjan, 2005], [Leibold, 2001], [Sellmann, 2002], [Strauß, 2001], [Suhl, 1995], [Yu and Thengvall, 2002], [Yu and Yang, 1998].

2.1.1 Netzwerkplanung

In der langfristigen Planung wird das Flugnetz der Fluggesellschaft entwickelt. Jedes Jahr wird das Flugnetz an die aktuelle Situation angepasst. Eine kompletter Neuaufbau des Netzwerks erfolgt selten, in den meisten Fällen werden Anpassungen des bestehenden vorgenommen. Die Anpassungen können alle Parameter des Netzwerks betreffen. Neue Städteverbindungen werden in den Flugplan aufgenommen, die Anzahl der Flüge zwischen zwei Städten wird erhöht oder verringert, neue Flughäfen können zu Hubs ausgebaut werden usw. Im Folgenden geben wir eine Übersicht über die einzelnen Aufgaben. Die Hauptursachen für Anpassungen sind Veränderungen im Passagieraufkommen, die sich in einzelnen Märkten vollziehen oder globale Ursachen haben (wie z.B. veränderte Weltwirtschaftslage). Diese Veränderungen müssen durch die Marketingabteilungen erfasst und in einem neuen Flugplan berücksichtigt werden.

Die Aufgaben der Netzwerkplanung im Einzelnen sind (siehe Abbildung 2.2):

Market selection/Auswahl der Städte Welche Städte/Flughäfen sollen angeflogen werden? Welche neuen Märkte werden durch die Aufnahme einer Stadt erschlossen? Ist der

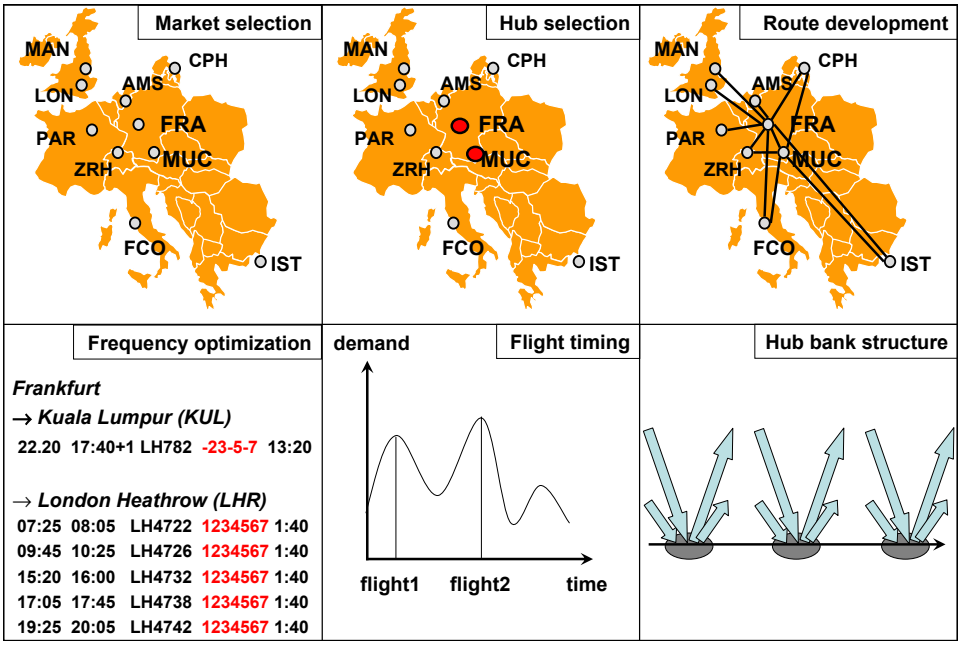


Abbildung 2.2: Aufgaben der Netzwerkplanung

Betrieb einer neuen Strecke operationell machbar, müssen dafür neue Landerechte erworben werden?

Hub selection/Auswahl der Hub-Flughäfen Welche bestehenden Flughäfen können die Funktion eines Hub-Flughafens übernehmen? Der Münchener Flughafen wurde z.B. in den letzten Jahren zu einem zweiten Hub der Lufthansa neben Frankfurt ausgebaut. Dieser bündelt jetzt mehr Passagierverkehr und entlastet dadurch den ersten Hub.

Route development/Aufbau der Streckenverbindungen Die Aufnahme von neuen Flugstrecken oder sogar neuen Städten in den Flugplan erfordert eine genaue Analyse der Potentiale für die Fluggesellschaft. Auch die Etablierung neuer Code-share Verbindungen (Kooperationsflüge) fällt in diesen Bereich.

Frequency Optimization/Frequenz einer Verbindung Die Anzahl der Flüge auf einer Strecke variiert sehr stark und ist von vielen Faktoren abhängig. Eine Langstreckenverbindung (z.B. Frankfurt-Kuala Lumpur) kann einmal oder mehrmals pro Woche angeboten werden, eine Verbindung mit hohem Passagieraufkommen (z.B. Frankfurt-London) kann mit einer stündlichen Frequenz geflogen werden.

Flight Timing/Zeitenlagen der Flüge Die Abflug- und Ankunftszeiten müssen so gewählt werden, dass die angebotene Verbindung möglichst attraktiv für die Passagiere ist. Geographische Faktoren und das Angebot der Konkurrenz spielen dabei eine große Rolle. Die Verteilung des Transportbedarfs über den Tag bildet eine weitere Grundlage für die Auswahl der Zeitenlagen.

Hub bank structure/Verbindungsstruktur auf einem Hub-Flughafen Um die Anzahl der möglichen Verbindungen für Passagiere zu maximieren, versucht jede Airline, die an-

kommenden und abgehenden Flüge an einem Hub-Flughafen in Gruppen einzuteilen. Mehrere Ankünfte werden zeitnah platziert, gefolgt von einer Übergangszeit für den Wechsel des Fluges und einer Reihe von Abflügen. Die Struktur dieser Gruppen spielt eine große Rolle für den wirtschaftlichen Erfolg einer Fluggesellschaft.

Through flight optimization/Flüge ohne Umsteigeverbindungen Passagiere ziehen Verbindungen vor, die ohne Flugzeugwechsel von A nach C über B gehen (through flight). Die Fluggesellschaft kann also versuchen, bestimmte Verbindungen mit einem Flugzeug zu fliegen. Operationelle Machbarkeit muss natürlich, neben einer vorausgesagten Erhöhung des Passagieraufkommens auf dieser Verbindung, gegeben sein.

Literatur zur Netzwerkplanung

Literaturübersichten zum Thema Netzwerkplanung findet man z.B. in [Carl and Gesing, 2000], [Etschmaier and Mathaisel, 1984], [Teodorovic, 1988].

Frühe Arbeiten wie z.B. [Etschmaier and Mathaisel, 1984], [Teodorovic, 1988], [Berge, 1994], [Marsten et al., 1996], [Dobson and Lederer, 1993], [Soumis et al., 1980] lösen das Problem iterativ. Flugpläne werden bewertet, danach werden Flüge ausgewählt, gelöscht und neue hinzugefügt. Die Bewertung der Änderungen erfolgt mit Methoden der Marktmodellierung oder teilweise mit Modellen der Flottenzuweisung.

Neuere Ansätze arbeiten mit mathematischen Modellen, die Entscheidungen der Netzwerkplanung mit der Flottenzuweisung verbinden. So können [Lettovsky et al., 1999] Flugpläne sowohl neu aufbauen als auch in vordefinierter Weise verändern. Die Frequenzen können verändert werden, Flüge können zeitlich verschoben oder ganz gestrichen werden.

[Lohatepanont and Barnhart, 2004], [Lohatepanont, 2002] wählen aus der Menge der Kandidatenflüge solche aus, die den Profit des modellierten Passagierflusses maximieren.
[Yan and Wang, 2001], [Yan and Tseng, 2002] lösen ähnliche mathematische Modelle mit Lagrange-Relaxationen.

Die Arbeit von [Lederer and Nambimadom, 1998] untersucht Eigenschaften des Netzwerks, wie z.B. Entfernung zwischen Städten, Transportbedarf und Größe des Netzwerks, die die optimale Struktur beeinflussen. So sind bei bestimmten Parametern Hub-and-spoke Netzwerke optimal, bei anderen sind Direktverbindungen kostenminimal.

Andere Teilaufgaben der Netzwerkplanung werden in folgenden Arbeiten behandelt:

- Hub Location: [Aykin, 1994], [Jaillet et al., 1996]
- Struktur des Netzwerks: [Lederer and Nambimadom, 1998]
- Route Development: [Berdy, 2002]
- Frequenzplanung: [Teodorovic and Krcmar-Nozic, 1989]
- Hub optimization: [Derigs et al., 2001]
- Auswahl der Wartungsflughäfen: [Feo and J.F., 1989]

Verwandte Probleme tauchen bei der Entwicklung eines Netzwerks für Luftpostverkehr [Armacost et al., 2002], [Grünert et al., 2000] auf. Netzwerke der Charter-Fluggesellschaften werden in [Erdmann, 1999] untersucht.

2.2 Netzwerkentwurf

Beim Netzwerkentwurfproblem geht es darum, für einen gegebenen Graphen eine Auswahl von Kanten zu treffen, wobei ein ebenfalls in der Eingabe spezifizierter Mehr-Güter-Fluss durch das resultierende Netzwerk geroutet werden soll. Die Kanten des Netzwerks können dabei für einen fixen Betrag installiert werden und besitzen außerdem eine beschränkte Transportkapazität. Für den Transport einer Einheit über eine Kante fallen zusätzlich Kosten an, die auch von der Art des zu transportierendes Gutes abhängen können. Das Ziel besteht darin, mit möglichst niedrigen Installations- und Transportkosten die Transportnachfrage zu befriedigen. Eine Übersicht zu diesem Forschungsthema findet man z.B. in [Balakrishnan et al., 1997], [Magnanti and Wong, 1984], [Crainic, 2000].

Die beschriebene Problematik ist relevant für zahlreiche Anwendungsfälle wie Transport, Verkehr und Telekommunikation. Es sind dabei oft noch viele zusätzliche Randbedingungen zu beachten, wobei jedoch das oben informal definierte Problem stets den Kern bildet. Netzwerkentwurfprobleme im Bereich Telekommunikation etwa müssen zusätzlich Aspekte der Ausfallsicherheit beachten.

In hier betrachteten Fall stellt sich die Frage des Aufbaus oder der Verbesserung eines Flugnetzwerks: Für eine Strecke im Eingabe-Netzwerk muss entschieden werden, ob sie unter Berücksichtigung der Passagierflüsse in den Flugplan aufgenommen wird oder nicht.

2.2.1 Modellierung

Ein Transportnetzwerk wird durch einen ungerichteten Graphen $G = (\mathscr{N}, \mathscr{A})$ mit Knotenmenge $\mathscr{N}$ und Kantenmenge $\mathscr{A}$ definiert. Mit $\mathscr{C}$ bezeichnen wir die Menge der Transportgüter (*commodities*). Für ein Gut k sei $O(k)$ sein Startknoten, $D(k)$ sein Zielknoten und sein Transportbedarf betrage d^k. Die Transportkosten c_{ij}^k fallen an, wenn eine Einheit eines Gutes k auf einer Kante $(i,j) \in \mathscr{A}$ transportiert wird, und f_{ij} sind die Fixkosten für die Benutzung bzw. Installation der Kante (i,j). Jede Kante besitze eine Gesamtkapazität u_{ij}, die von allen auf Kante (i,j) transportierten Gütermengen nicht überschritten werden darf.

Die Entscheidungsvariablen des Modells werden bezeichnet mit x_{ij}^k und y_{ij}, wobei x_{ij}^k die auf der Kante (i,j) transportierte Menge des Gutes k darstellt, und die Binärvariable y_{ij} entscheidet, ob die Kante (i,j) in einer Lösung zum Gütertransport benutzt wird oder nicht.

Definition 2.1 (CNDP, kantenbasiertes Modell) *Mit den obigen Bezeichnungen lautet das kantenbasierte Modell des Netzwerkentwurfproblems mit Kantenkapazitäten (engl. Fixed Charge Capacitated Network Design Problem - CNDP) nun wie folgt:*

$$
\begin{aligned}
\min \quad & \sum_{ij} \sum_{k} c_{ij}^k x_{ij}^k + \sum_{ij} f_{ij} y_{ij} \\
\text{u.d.N.} \quad & \sum_{j:(i,j)\in\mathscr{A}} x_{ij}^k - \sum_{j:(j,i)\in\mathscr{A}} x_{ji}^k = b_i^k \qquad \forall i \in \mathscr{N},\ \forall k \in \mathscr{C} \qquad (1) \\
& \sum_{k\in\mathscr{C}} x_{ij}^k \leq u_{ij} y_{ij} \qquad \forall (i,j) \in \mathscr{A} \qquad (2) \\
& x_{ij}^k \geq 0 \qquad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C} \qquad (3) \\
& y_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathscr{A} \qquad (4)
\end{aligned}
$$

$$\text{mit } b_i^k = \begin{cases} d^k & \text{falls} \quad i = O(k) \\ -d^k & \text{falls} \quad i = D(k) \\ 0 & \text{sonst.} \end{cases}$$

Durch die Zielfunktion wird die Summe der Transportkosten über alle transportierten Gütermengen und der Fixkosten für alle benutzten Kanten minimiert.

Die Bedingungen in (1) definieren $|\mathcal{N}||\mathcal{C}|$ Fluss-Bedingungen (*flow-constraints*), die jeden Knoten bezüglich jeden Gutes als einen Start-, Ziel- oder Durchgangsknoten definieren. Bei der vorliegenden Version des CNDP hat jedes Gut genau einen Start- und Zielknoten. Problemstellungen, in denen ein Gut mehrere Start- und/oder Zielknoten besitzt, können ohne weiteres in die Formulierung übertragen werden, indem z.B. Super-Quellen und Super-Senken für jedes Gut definiert werden.

Die Bedingungen in Zeile (2) beschränken den Fluss auf jeder Kante. Diese $|\mathcal{A}|$ Kapazitätsbedingungen (*capacity-*, *bundle-* oder *forcing-constraints*) verbinden die Flussvariablen x mit den Designvariablen y. Die Gesamtkapazität u_{ij} jeder Kante (i,j) wird unter einer Reihe von Gütern aufgeteilt. Die $|\mathcal{A}||\mathcal{C}|$ Nichtnegativitätsbedingungen der Flussvariablen in Zeile (3) und die $|\mathcal{A}|$ Ganzzahligkeitsbedingungen der binären Designvariablen in Zeile (4) vervollständigen das Modell.

Die in Definition 2.1 vorgestellte Version des kantenbasierten Modells wird auch *aggregiert* genannt, keine der Nebenbedingungen sind redundant. Die so genannte *deaggregierte* Form erhält man durch Hinzunahme der folgenden Nebenbedingungen:

$$x_{ij}^k \quad \leq \quad d_{ij}^k y_{ij} \quad \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{C} \quad (5)$$

mit $d_{ij}^k = \min(d^k, u_{ij})$.

Diese $|\mathcal{A}||\mathcal{C}|$ zusätzlichen Kapazitätsbedingungen sind redundant bezüglich zulässigen (und optimalen) Lösungen des Modells. Deaggregierte Modelle führen jedoch oft, so auch in diesem Fall, zu deutlich besseren Schranken der zugehörigen Relaxationen. Andererseits vergrößert sich die Anzahl der Nebenbedingungen deutlich: Die aggregierte Version besitzt (ohne Variablenbeschränkungen) $|\mathcal{N}||\mathcal{C}| + |\mathcal{A}|$ Nebenbedingungen, in der deaggregierte Form kommen noch einmal $|\mathcal{C}||\mathcal{A}|$ Nebenbedingungen hinzu.

Die LP-Relaxation beider Varianten des Modells erhält man, wenn man die Ganzzahligkeitsbedingungen (4) durch $0 \leq y_{ij} \leq 1$ und $y_{ij} \in \mathbb{R}$ für alle $(i,j) \in \mathcal{A}$ ersetzt. In einer optimalen Lösung der aggregierten Variante werden die Variablen y_{ij} nun immer ihren minimal zulässigen Wert $\sum_{k \in \mathcal{C}} x_{ij}^k / u_{ij}$ annehmen. Man kann also die y_{ij} streichen und erhält ein Mehrgüter-Fluss-Problem. In der deaggregierten Variante der LP-Relaxation kann diese Neuformulierung des Problems nicht durchgeführt werden, da hier für die y_{ij} in einer optimalen Lösung gilt: $y_{ij} = \max\{(\sum_{k \in \mathcal{C}} x_{ij}^k / u_{ij}), (\max_{k \in \mathcal{C}} (x_{ij}^k / d_{ij}^k))\}$, was bei der Ersetzung der y_{ij} zu einer nichtlinearen Zielfunktion führen würde.

Aus diesem Grund liefert die LP-Relaxation des deaggregierten Modells deutlich bessere untere Schranken. Dies wird allerdings erkauft durch die höhere Komplexität, die gerade bei Simplex-basierten Lösungsalgorithmen stark von der Anzahl der Nebenbedingungen abhängt. Trotzdem hat es sich als sinnvoll erwiesen, im Rahmen von Lösungsalgorithmen für das CNDP, die auf Branch-and-Bound aufbauen, die unteren Schranken mittels des deaggregierten Modells zu berechnen. Einen ausführlichen Vergleich findet man in [Gendron and Crainic, 1994a].

Das grundlegende Problem des Netzwerkentwurfs ist NP-vollständig, weil das Steiner-Baum Problem als ein Spezialfall des Netzwerkentwurfproblems formuliert werden kann.

2.2.2 Das pfadbasierte Modell

Eine andere Art, das CNDP zu modellieren, ist die pfadbasierte Formulierung. Die Flussvariablen im kantenbasierten Modell definieren den Güterfluss auf einer Kante des Transportnetzwerks. Die Flussvariablen im pfadbasierten Modell geben die transportierte Menge jedes Gutes auf jedem möglichen Pfad von seinem Startknoten zu seinem Zielknoten an. Sei P_k die Menge der Pfade von $O(k)$ nach $D(k)$ für ein Gut k. Für jeden Pfad $p \in P_k$ sei h_p^k der Güterfluss des Gutes k. Sei $\delta_{ij}^p = 1$, falls der Pfad p die Kante $(i,j) \in \mathscr{A}$ enthält, und sonst $\delta_{ij}^p = 0$. Bezeichne $c_p^k = \sum_{(i,j)\in\mathscr{A}} \delta_{ij}^p c_{ij}^k$ die Transportkosten des Pfades p bezüglich des Gutes k.

Definition 2.2 (CNDP, pfadbasiertes Modell) *Mit diesen Bezeichnungen lautet das pfadbasierte Modell des CNDP wie folgt:*

$$
\begin{array}{lrcll}
\min & \displaystyle\sum_{k\in\mathscr{C}}\sum_{p\in P_k} c_p^k h_p^k + \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} & & & \\
\textit{u.d.N.} & \displaystyle\sum_{p\in P_k} h_p^k & = & d^k & \forall k \in \mathscr{C} \qquad (1) \\
& \displaystyle\sum_{k\in\mathscr{C}}\sum_{p\in P_k} \delta_{ij}^p h_p^k & \leq & u_{ij} y_{ij} & \forall (i,j) \in \mathscr{A} \qquad (2) \\
& h_p^k & \geq & 0 & \forall k \in \mathscr{C}, \forall p \in P_k \qquad (3) \\
& y_{ij} & \in & \{0,1\} & \forall (i,j) \in \mathscr{A} \qquad (4)
\end{array}
$$

In Zeile (1) werden die $|\mathscr{N}||\mathscr{C}|$ Flussbedingungen aus dem kantenbasierten Modell durch $|\mathscr{C}|$ Nebenbedingungen ersetzt, die für jedes Gut den Transport der benötigten Gütermenge garantieren. Zeile (2) stimmt im Wesentlichen mit den Kapazitätsbedingungen im vorhergehenden Modell überein, die Zuordnung von Kanten zu Pfaden wird durch die binäre δ-Funktion erreicht. Nichtnegativitätsbedingungen (3) und Ganzzahligkeitsbedingungen (4) vervollständigen das Modell. Auch hier kann eine deaggregierte Version angegeben werden, indem zusätzliche, redundante Kapazitätsbedingungen formuliert werden.

Ein wichtiger Vorteil des pfadbasierten Modells ist die deutlich kleinere Anzahl an Nebenbedingungen: Statt $|\mathscr{N}||\mathscr{C}|+|\mathscr{A}|$ Nebenbedingungen in der kantenbasierten Version kommt es mit nur $|\mathscr{C}|+|\mathscr{A}|$ Nebenbedingungen aus. Dies wird allerdings erkauft mit einer Zahl von Pfadvariablen, die exponentiell in der Anzahl der Kanten ist. Trotzdem wurde in der Literatur versucht, im Rahmen von Branch-and-Price Verfahren die kleinere Zahl von Nebenbedingungen bei der Lösung der LP-Relaxation durch die Anwendung von Column-Generation auszunutzen (vgl. [Clarke et al., 1996]).

Weitere Vorteile dieser Formulierung können sich bei Erweiterungen der Problemstellung durch zusätzliche Bedingungen ergeben, die sich direkt auf die Transportpfade beziehen (z.B. Pfadlängenbeschränkungen). In diesen Fällen muss allerdings berücksichtigt werden, dass das bei der Verwendung von Column-Generation auftretende Kürzeste-Wege-Problem NP-vollständig sein kann.

2.2.3 Das Mehrgüter-Fluss-Problem

Wie in Abschnitt 2.2.1 bereits erwähnt, ergibt sich für die aggregierte Form des CNDP bei gegebenem Designvektor y das so genannte Mehrgüter-Fluss Problem (*Min Cost Multicom-*

modity Flow Problem - MMCF).

Definition 2.3 (MMCF, kantenbasiertes Modell) *Mit denselben Bezeichnungen für Parameter und Variablen wie oben lautet eine kantenbasierte Formulierung des* Mehrgüter-Fluss Problems *wie folgt:*

$$
\begin{aligned}
\min \quad & \sum_{ij}\sum_{k} c_{ij}^k x_{ij}^k & & & & \\
\text{u.d.N.} \quad & \sum_{j:(i,j)\in\mathscr{A}} x_{ij}^k - \sum_{j:(j,i)\in\mathscr{A}} x_{ji}^k & = & b_i^k & \forall i \in \mathscr{N},\ \forall k \in \mathscr{C} & (1)\\
& \sum_{k\in\mathscr{C}} x_{ij}^k & \leq & u_{ij} & \forall (i,j) \in \mathscr{A} & (2)\\
& x_{ij}^k & \geq & 0 & \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C} & (3)
\end{aligned}
$$

Betrachtet man das MMCF für nur ein Gut, so erhält man das grundlegende Problem, einen kostenminimalen Fluss in einem kapazitierten Transportnetzwerk zu finden (*Min Cost Flow Problem - MCF*). Eine wichtige Eigenschaft des MMCF ist, dass bei der Berücksichtigung mehrerer Güter die Integralitätseigenschaft verloren geht – es muss also nicht notwendigerweise eine ganzzahlige Lösung existieren. Wenn man allerdings die Kapazitätsbedingungen (2) streicht, dann zerfällt der Rest des Problems in $|\mathscr{C}|$ Kürzeste-Wege-Probleme, die unabhängig voneinander effizient (z.B. mit Dijkstras Algorithmus) gelöst werden können.

Aufgrund seiner Praxisrelevanz (z.B. in der Telekommunikation), seiner interessanten Struktur und seiner besonderen Komplexität wird das MMCF seit langem in der Literatur untersucht (vgl. Kapitel 17 in [Ahuja et al., 1993]). Das in dieser Arbeit verwendete Lösungsverfahren für das MMCF wird in Abschnitt 2.7.2 im Detail erläutert.

2.2.4 Modellvarianten

In der Literatur werden in einer Vielzahl von Anwendungen Modellerweiterungen des CNDP untersucht. Für die folgenden Varianten haben sich feste Bezeichnungen eingebürgert:

- Das *Uncapacitated Network Design Problem (UNDP)* ergibt sich aus dem CNDP, wenn man die Kapazitätsbedingungen streicht.

- Das *Network Loading Problem (NLP)* tritt insbesondere in Anwendungen in der Telekommunikation auf, in denen den Kanten Kapazitäten in diskreten Schritten (z.B. verschiedene Bandbreiten) zugewiesen werden können. In den entsprechenden Modellen werden für den Designvektor y auch Werte > 1 zugelassen.

- Beim *Service Network Design Problem (SNDP)* existieren zusätzliche Flussbedingungen auf den Designvariablen. Diese Problemstellung taucht z.B. auf, wenn die eingesetzten Transportressourcen (LKWs, Flugzeuge) in jedem Knoten balanciert sein müssen.

- Schließlich werden Netzwerkentwurfprobleme mit zusätzlichen Restriktionen auf den Transportpfaden untersucht. Dies können beispielsweise Längenbeschränkungen, aber auch Beschränkungen durch Zeitfenster sein.

2.2.5 Literaturübersicht

Die Vorarbeiten im Bereich Netzwerkentwurf sind auf Grund der hohen praktischen Relevanz der Fragestellung sehr vielfältig. Eine große Anzahl von Erweiterungen des Kernproblems ist untersucht worden. Die Lösungsmethoden reichen von Lagrange-Relaxationsverfahren über Column-Generation-Verfahren bis hin zu Ansätzen basierend auf Metaheuristiken, wie Tabu Search oder Simulated Annealing.

Crainic, Frangioni und Gendron haben sich in einer Reihe von Arbeiten mit der Generierung von unteren Schranken für das CNDP befasst (siehe z.B. [Gendron and Crainic, 1994a], [Gendron et al., 1998] und [Crainic et al., 2001a]). Ihre wichtigsten Einsichten sind folgende: Die *starke* LP-Relaxation liefert deutlich bessere untere Schranken als die *schwache* LP-Relaxation (s. Abschnitt 2.4.1), ist aber auch sehr viel aufwendiger zu berechnen. Lagrange Relaxationen liefern gute Approximationen der starken Schranke und können sehr viel schneller berechnet werden als die LP-Relaxation. Zur Optimierung des Lagrange Duals schneiden Bundle-Methoden bezüglich Konvergenzverhalten, Lösungsqualität und Robustheit etwas besser ab als Subgradientenverfahren, Letztere sind aber deutlich einfacher zu implementieren.

In [Gendron et al., 1998], [Crainic et al., 2001a] haben Gendron und andere das Netzwerkentwurfproblem untersucht. Es werden eine kanten- und eine pfadbasierte Formulierung des CNDP vorgestellt und mehrere Lagrange-Relaxierungen entwickelt. Auf automatisch generierten Datensätzen mit bis zu 30 Knoten, 700 Kanten und bis zu 400 Gütern vergleichen die Autoren ihre Ansätze mit einem exakten Branch-and-Bound Ansatz in Hinblick auf Laufzeit und Lösungsqualität.

In [Crainic et al., 2000a] stellen Crainic et al. einen kombinierten Column Generation und Tabu Search Ansatz vor, bei dem für die Definition des Suchraumes zunächst alle Design-Entscheidungsvariablen y_{ij} auf 1 gesetzt werden. Das dabei entstehende Problem ist ein Mehr-Güter-Fluss-Problem, das als ein LP formuliert ist. Die Ecken des Polytops werden mit Hilfe lokaler Suche durchsucht. Dabei wird der Übergang von einer Ecke zur anderen durch Simplex-Pivot-Schritte vollzogen. Die Tabu-Kriterien beziehen sich auf die Eigenschaften der Pivot-Variablen. Der Ansatz liefert gute Lösungen für Netzwerke mit bis zu 100 Knoten. Die Laufzeiten des Algorithmus bewegen sich zwischen 30 Minuten bei 30 Knoten, 700 Kanten, 100 Gütern und 4 Stunden für 30 Knoten, 700 Kanten und 400 Gütern.

In [Holmberg and Yuan, 2000] präsentieren die Autoren ein Branch-and-Bound Verfahren für das CNDP, das sowohl exakte als auch heuristische Lösungen liefert. Sie setzen zur Schrankenberechnung die Lagrange Rucksack-Relaxation ein. Zur Lösung des Lagrange-Duals setzen sie ein Subgradientenverfahren ein, in das sie sog. *penalty tests* zur Variablenfixierung und eine schnelle Heuristik zur Generierung von oberen Schranken integrieren. Durch Techniken der heuristischen Variablenfixierung machen sie aus dem Verfahren eine Heuristik. Sie präsentieren sowohl für die exakte als auch für die heuristische Variante des Verfahrens sehr vielversprechende Messergebnisse auf einer Reihe von selbst erzeugten Instanzen, die sie mit Ergebnissen der Standardsoftware CPLEX vergleichen.

In den Arbeiten von Ghamlouche und anderen [Ghamlouche et al., 2003], [Ghamlouche et al., 2004] werden heuristische Verfahren für das CNDP vorgestellt. Aufbauend auf einer aufwendigen Nachbarschaft wird ein Tabu Search Ansatz verfolgt. Die Laufzeiten der Verfahren steigen zwar nicht unwesentlich im Vergleich zu anderen heuristischen

Verfahren, die erzielte Lösungsqualität wird aber entscheidend verbessert. Das in der zweiten Arbeit verfolgte Ansatz des Path Relinking ermöglicht einen weiteren Schritt zur Verbesserung der Lösungsqualität.

In [Crainic et al., 2004] wird ein Slope Scaling Ansatz für das CNDP vorgestellt. Es wird eine Folge von Multicommodity Flussproblemen gelöst, die gegen die optimale Lösung des CNDP konvergiert. Durch die iterative Anpassung der Kostenparameter für den Transport der Güter im Netzwerk erreicht man die gewünschte Konvergenz. Die Qualität der Lösungen wird durch den Einsatz von Techniken aus dem Bereich der heuristischen Optimierung, wie z.B. Langfristgedächtnis, verbessert. Außerdem wird die Anpassung der Kostenparameter von der parallel laufenden Lagrange Relaxation des Problems beeinflusst.

Bienstock et al. beschreiben in [Bienstock et al., 1998] zwei Cutting-Plane Algorithmen für eine Variante des CNDP mit multiplen Kanten (Kanten können mehrfach geöffnet werden). Der erste basiert auf einer Multicommodity Formulierung des CNDP und benutzt sog. *cutset*– und *three partition*– Ungleichungen. Der andere fügt die folgenden *cutting planes* hinzu: *total capacity*, *partition* und *rounded metric* Ungleichungen. Eingebettet in einen Branch-and-Cut Rahmen liefern beide Verfahren gute Ergebnisse auf realistischen Benchmark Daten. In [Bienstock, 1999] wird eine deutliche Verbesserung des Verfahrens auf den gleichen Benchmarks durch den Einsatz von approximierten linearen Programmen erreicht.

Weitere Branch-and-Cut Ansätze für die gleiche Variante des CNDP werden von Günlük, Atamtürk et al. in [Günlük, 1999] und [Atamtürk, 1999] untersucht. Es werden eine Reihe von gültigen Ungleichungen (*valid inequalities*) betrachtet und es wird festgestellt, dass Branch-and-Cut im Vergleich mit Branch-and-Bound auf der benutzten Benchmark deutlich überlegen ist. Durch das Hinzufügen von Cuts konnte außerdem der Integrality-Gap am Wurzelknoten signifikant reduziert und damit auch die Anzahl besuchter Knoten erheblich vermindert werden.

Sridhar und Park stellen in [Sridhar and Park, 2000] eine Implementierung eines Benders-and-Cut Algorithmus vor. Das Verfahren besteht aus drei Teilen: einem Cutting-Plane Algorithmus zur Generierung von guten unteren Schranken, einer Heuristik zur Erzeugung von oberen Schranken und dem Benders-and-Cut Algorithmus selbst. Es werden Messergebnisse für eine große Anzahl von Testinstanzen mit unterschiedlichen Transportbedarfen präsentiert. Die Autoren stellen fest, dass die Komplexität der Testinstanzen stark vom Transportbedarf abhängt und mit steigendem Transportbedarf zunimmt. Das liegt vor allem an der Vergrößerung des Integrality Gaps. Die Anwendung ihres Algorithmus empfehlen sie besonders für die schwierigen Instanzen. Um die LP-Relaxation zu verbessern, erwiesen sich *flow inequalities* als sehr effektiv.

Clark und Gong vergleichen in [Clarke and Gong, 1998] einen Branch-and-Price Ansatz für die pfadbasierte Formulierung des CNDP mit traditionellem Branch-and-Bound für die kantenbasierte Formulierung. Sie stellen fest, dass die pfadbasierte Formulierung etwas effizienter ist und sich dieser Vorsprung durch SOS-Branching noch steigern lässt. Sie erklären diese Überlegenheit mit der schnelleren Lösbarkeit der Unterprobleme (Kürzeste-Wege), die die effizientere Lösung der LP-Relaxationen erlaubt. Messergebnisse werden für Instanzen mit 6, 10 und 15 Knoten präsentiert.

In [Barnhart and Kim, 1997] stellen Barnhart und Kim das Service Network Design Problem vor, das als Erweiterung zum oben definierten Netzwerkentwurfproblem eine neue Kom-

ponente beinhaltet: In jedem Knoten des Netzwerkes muss eine Balance der eingesetzten Transportressourcen sichergestellt werden. Die Autoren stellen einen Branch-and-Bound Algorithmus vor, der Column- und Row-Generation verbindet (Branch-And-Cut-And-Price). Auf diese Weise wird versucht, die riesige Anzahl von binären Entscheidungsvariablen in den Griff zu bekommen und schwache LP-Schranken zu verstärken. Es müssen nur wenige Spalten generiert werden und die Qualität der LP-Relaxationen wird durch hinzugefügte Cuts stark verbessert.

In [Jaillet et al., 1996] stellen Jaillet und andere ein Modell für den Netzwerkentwurf bei Fluglinien auf. Die Reisepfade für die zu transportierenden Passagiere werden in ihrer Länge eingeschränkt: Jeder Passagier darf höchstens über zwei Zwischenstationen fliegen. Die Arbeit beschäftigt sich u.a. mit der Möglichkeit, mehrere unterschiedliche Flugzeugtypen im Netzwerk einzusetzen. Die Ergebnisse werden auf Netzwerken erzielt, die bis zu 39 Knoten (Städte) haben. Eine Skalierung der Passagierzahlen im Netzwerk spielt neben der Frage, wie viele Flugzeugtypen eingesetzt werden, eine große Rolle für die Laufzeit des Programms. Der algorithmische Ansatz benutzt mehrere problemspezifische Heuristiken, die Relaxierungen des Problems verwenden und aus den relaxierten Lösungen solche für das ursprüngliche Problem finden.

2.3 Überblick über das Lösungsverfahren

In diesem Abschnitt geben wir einen Überblick über den in dieser Arbeit benutzten Branch-and-Bound Algorithmus für das Netzwerkentwurfproblem. Die Grundstruktur entspricht einem klassischen Branch-and-Bound Algorithmus, der im Algorithmus 1 beschrieben wird.

Algorithmus 1 Branch-and-Bound für das CNDP

1: *Initialisierung*: *Beste bekannte Lösung:* $\hat{z}_{CNDP} = \infty$
2: **while** *(es gibt noch nicht untersuchte Knoten)* **do**
3: *wähle einen noch nicht untersuchten Knoten aus (Tiefen- oder Bestensuche)*
4: *starte primale Heuristik (siehe 2.7.3)*
5: **if** (*Heuristik findet keine Lösung oder der untersuchte Knoten ist ein Blatt*) **then**
6: *starte exaktes Verfahren (Column Generation)*
7: *berechne untere Schranke* z_R *mittels Subgradient (2.6.1) oder Bundle-Methode (2.6.2)*
8: *führe Variablenfixierung durch (2.9)*
9: *[starte heuristische* α*- oder* β*-Fixierung (2.10)]*
10: *fixiere Variablen anhand zusätzlicher Ungleichungen (2.11)*
11: **if** $(z_R < \hat{z}_{CNDP})$ **then**
12: *wähle eine neue Branchingvariable (2.8)*
13: *erzeuge zwei neue Knoten*

Es wurden zwei Baumsuchstrategien implementiert: die Tiefensuche und die Bestensuche. In beiden Fällen können Informationen aus den Vaterknoten in den Nachfolgerknoten benutzt werden. Für die Suche nach zulässigen Lösungen werden zwei Verfahren benutzt: Das heuristische wird im Abschnitt 2.7.3 beschrieben, das exakte ist ein Column Generation Ansatz für das pfadbasierte Modell. Die untere Schranke wird alternativ durch die Subgradient-Suche

(siehe Abschnitt 2.6.1) oder die Bundle-Methode (siehe Abschnitt 2.6.2) berechnet. Im Abschnitt 2.9 stellen wir die Strategien zur Variablenfixierung vor, im Abschnitt 2.10 werden die heuristischen Strategien erläutert. Durch die Generierung zusätzlicher Ungleichungen (*cuts*) wird der Branch-and-bound Algorithmus zum Relax-and-cut-Algorithmus (siehe Abschnitt 2.11).

2.4 Untere Schranken

2.4.1 LP-Schranken

In [Gendron and Crainic, 1994a] und [Crainic et al., 2001a] werden von den Autoren eine Vielzahl von Relaxationen für das CNDP untersucht. In unserem Branch-and-Bound Algorithmus werden zur Berechnung der unteren Schranke die beiden erfolgversprechendsten Relaxationen eingesetzt, nämlich die sog. *Kürzeste-Wege-Relaxation* und die *Rucksack-Relaxation*, die hier nach den jeweils zu lösenden Unterproblemen benannt wurden. Der Vollständigkeit halber werden wir auch auf die möglichen LP-Relaxationen des Problems sowie auf eine Variante der Kürzeste-Wege-Relaxation eingehen.

Aus Gründen der Übersichtlichkeit möchten wir noch einmal das allen hier vorgestellten Relaxationen zugrunde liegende deaggregierte, kantenbasierte Modell aus Abschnitt 2.2.1 wiederholen:

$$
\begin{aligned}
z_{CNDP} \quad = \quad & \min \quad \sum_{ij}\sum_{k} c_{ij}^k x_{ij}^k + \sum_{ij} f_{ij} y_{ij} & & & & \\
& \text{u.d.N.} \quad \sum_{j:(i,j)\in\mathscr{A}} x_{ij}^k - \sum_{j:(j,i)\in\mathscr{A}} x_{ji}^k &=& b_i^k & \forall i \in \mathscr{N},\, \forall k \in \mathscr{C} & (1)\\
& \qquad\qquad \sum_{k\in\mathscr{C}} x_{ij}^k &\leq& u_{ij} y_{ij} & \forall (i,j) \in \mathscr{A} & (2)\\
& \qquad\qquad x_{ij}^k &\leq& d_{ij}^k y_{ij} & \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C} & (3)\\
& \qquad\qquad x_{ij}^k &\geq& 0 & \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C} & (4)\\
& \qquad\qquad y_{ij} &\in& \{0,1\} & \forall (i,j) \in \mathscr{A} & (5)
\end{aligned}
$$

Wir bezeichnen im Weiteren die Nebenbedingungen in Zeile (2) als *schwache* und jene in Zeile (3) als *starke* Kapazitätsbedingungen. Aus der Redundanz der starken Kapazitätsbedingungen folgen für jede prinzipiell mögliche Relaxation zwei Varianten, nämlich eine *schwache*, bei der die starken Kapazitätsbedingungen gestrichen werden, und eine *starke*, bei der sie in der Relaxation mitberücksichtigt werden. Da die starken Relaxationen deutlich bessere Schranken liefern als die schwachen, werden wir hier nur für die LP-Relaxation auf beide Varianten eingehen, während die Lagrange-Relaxationen allesamt starke Relaxationen sind.

Die schwache LP-Relaxation

Die *schwache LP-Relaxation* erhält man, indem man die starken Kapazitätsbedingungen (3) streicht und die Ganzzahligkeitsbedingungen (5) durch einfache LP-Schranken für die y-Variablen $0 \leq y_{ij} \leq 1$ ersetzt. Da in einer optimalen Lösung des neuen Problems die y-Variablen immer ihren minimalen Wert $y_{ij} = \sum_{k\in\mathscr{C}} x_{ij}^k / u_{ij}$ annehmen, ergibt sich durch Einsetzen das Mehrgüter-Fluss-Problem:

$$
\begin{array}{rcll}
z_{WLP} & = & \min & \sum_{ij}\sum_{k}(c_{ij}^k + f_{ij}/u_{ij})x_{ij}^k \\
 & & \text{u.d.N.} & \sum_{j:(i,j)\in\mathscr{A}} x_{ij}^k - \sum_{j:(j,i)\in\mathscr{A}} x_{ji}^k = b_i^k \quad \forall i \in \mathscr{N},\ \forall k \in \mathscr{C} \\
 & & & \sum_{k\in\mathscr{C}} x_{ij}^k \leq u_{ij} \quad \forall (i,j) \in \mathscr{A} \\
 & & & x_{ij}^k \geq 0 \quad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C}
\end{array}
$$

Das Mehrgüter-Fluss-Problem kann effizient durch Dekompositionsverfahren gelöst werden (siehe Abschnitt 2.7.2).

Die starke LP-Relaxation

Bei der *starken LP-Relaxation*, deren Zielfunktionswert wir mit z_{SLP} bezeichnen wollen, werden nur die Ganzzahligkeitsbedingungen (5) relaxiert. Das entsprechende lineare Programm besitzt damit nicht nur deutlich mehr Nebenbedingungen als die schwache LP-Relaxation, sondern verliert auch die Eigenschaft, ein Mehrgüter-Fluss-Problem zu sein, wie wir schon im Abschnitt 2.2.1 erörtert haben. Der höhere Lösungsaufwand wird jedoch gerade im Rahmen von Branch-and-Bound durch die bessere Qualität der Schranke mehr als wettgemacht.

2.4.2 Lagrange-Schranken: Rucksack und Kürzeste-Wege

Um die **Kürzeste-Wege-Relaxation** (*shortest-path* oder *flow relaxation*) zu erhalten, werden sowohl schwache als auch starke Kapazitätsbedingungen Lagrange-relaxiert. Ihnen werden dabei $|\mathscr{A}|+|\mathscr{A}||\mathscr{C}|$ Lagrangemultiplikatoren $\alpha_{ij} \geq 0$ bzw. $\beta_{ij}^k \geq 0$ für alle $(i,j) \in \mathscr{A}$ und $k \in \mathscr{C}$ zugeordnet. Die Relaxation lautet dann für feste α und β wie folgt [1]:

$$
\begin{array}{rcll}
z_{SP}(\alpha,\beta) & = & \min & \sum_{(i,j)\in\mathscr{A}}\sum_{k\in\mathscr{C}} c_{ij}^k x_{ij}^k + \sum_{ij} f_{ij}y_{ij} + \sum_{(i,j)\in\mathscr{A}} [\alpha_{ij}(\sum_{k\in\mathscr{C}} x_{ij}^k - u_{ij}y_{ij}) \\
 & & & + \sum_{k\in\mathscr{C}} \beta_{ij}^k (x_{ij}^k - d_{ij}^k y_{ij})] \\
 & & \text{u.d.N.} & \sum_{j:(i,j)\in\mathscr{A}} x_{ij}^k - \sum_{j:(j,i)\in\mathscr{A}} x_{ji}^k = b_i^k \quad \forall i \in \mathscr{N},\ \forall k \in \mathscr{C} \\
 & & & x_{ij}^k \geq 0 \quad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C} \\
 & & & y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathscr{A}
\end{array}
$$

Die Zielfunktion lässt sich folgendermaßen nach den Entscheidungsvariablen x und y zusammenfassen:

$$
z_{SP}(\alpha,\beta) = \min \sum_{k\in\mathscr{C}}\sum_{(i,j)\in\mathscr{A}} (c_{ij}^k + \alpha_{ij} + \beta_{ij}^k)x_{ij}^k + \sum_{(i,j)\in\mathscr{A}} (f_{ij} - u_{ij}\alpha_{ij} - \sum_{k\in\mathscr{C}} d_{ij}^k \beta_{ij}^k)y_{ij}
$$

Man sieht nun, dass man das Problem in ein Kürzeste-Wege-Problem für jedes Gut und ein Entscheidungsproblem für die Designvariablen y zerlegen kann. Für ein Gut $k \in \mathscr{C}$ lautet das Kürzeste-Wege Problem $SP^k(\alpha,\beta)$ wie folgt:

[1] Die Lagrange-Multiplikatoren α_{ij} und β_{ij} sollten nicht mit den Parametern α und β der heuristischen Variablenfixierung verwechselt werden.

$$
\begin{aligned}
z^k_{SP}(\alpha,\beta) \quad = \quad & \min \sum_{(i,j)\in\mathscr{A}} (c^k_{ij}+\alpha_{ij}+\beta^k_{ij})x^k_{ij} \\
& \text{u.d.N.} \sum_{j:(i,j)\in\mathscr{A}} x^k_{ij} - \sum_{j:(j,i)\in\mathscr{A}} x^k_{ji} \quad = \quad b^k_i \quad \forall i \in \mathscr{N} \\
& \qquad\qquad\qquad\qquad\qquad\qquad x^k_{ij} \quad \geq \quad 0 \quad \forall (i,j) \in \mathscr{A}
\end{aligned}
$$

Da für die Kantenkosten $\tilde{c}^k_{ij} = c^k_{ij}+\alpha_{ij}+\beta^k_{ij} \geq 0$ für alle $(i,j) \in \mathscr{A}$ gilt, handelt es sich dabei um ein einfaches *Single-Source-Shortest-Path Problem*, das leicht mit Dijkstras Algorithmus gelöst werden kann. Nach einem Aufruf von Dijkstras Algorithmus gilt für den Lösungsvektor x^k:

$$
x^k_{ij} = \begin{cases} d^k & \text{falls Kante } (i,j) \text{ auf dem kürzesten Weg von } O(k) \text{ nach } D(k) \text{ liegt,} \\ 0 & \text{sonst.} \end{cases}
$$

Das Entscheidungsproblem für die y-Variablen lautet:

$$
\begin{aligned}
z^{(i,j)}_{SP}(\alpha,\beta) \quad = \quad & \min \sum_{(i,j)\in\mathscr{A}} (f_{ij} - u_{ij}\alpha_{ij} - \sum_{k\in\mathscr{C}} d^k_{ij}\beta^k_{ij})y_{ij} \\
& y_{ij} \quad \in \quad \{0,1\} \quad \forall (i,j) \in \mathscr{A}
\end{aligned}
$$

Die Lösung dieses Problems lässt sich einfach ablesen:

$$
y_{ij} = \begin{cases} 1 & \text{falls} \quad \tilde{f}_{ij} < 0 \text{ und} \\ 0 & \text{sonst,} \end{cases}
$$

wobei $\tilde{f}_{ij} = f_{ij} - u_{ij}\alpha_{ij} - \sum_{k\in\mathscr{C}} d^k_{ij}\beta^k_{ij}$ *reduzierte Fixkosten* heißen. Insgesamt ergibt sich also der Wert der Relaxation mit

$$
z_{SP}(\alpha,\beta) = \sum_{k\in\mathscr{C}} z^k_{SP}(\alpha,\beta) + \sum_{(i,j)\in\mathscr{A}} z^{(i,j)}_{SP}(\alpha,\beta),
$$

und das Lagrange-Multiplikator Problem lautet (siehe auch Abschnitt 2.5):

$$
z_{SP} = \max_{\alpha\geq 0, \beta\geq 0} z_{SP}(\alpha,\beta)
$$

An dieser Stelle sei darauf hingewiesen, dass die Kürzeste-Wege Relaxation die Integralitätseigenschaft besitzt, denn die y-Lösung würde sich auch dann nicht ändern, wenn man die Ganzzahligkeitsbedingung im Unterproblem streichen würde. Der Wert des Lagrange-Duals z_{SP} stimmt also mit dem optimalen Zielfunktionswert der starken LP-Relaxation z_{SLP} überein.

Das Lagrange-Multiplikator Problem (auch Lagrange-Dual genannt) kann z.B. mit dem Subgradientenverfahren oder dem Bundle-Verfahren gelöst werden. Ein Subgradient für die Kürzeste-Wege Relaxation wird berechnet, indem zunächst die $|\mathscr{C}|$ Kürzeste-Wege Probleme und das y-Entscheidungsproblem gelöst werden. Mit einer Lösung (x,y) lautet der Subgradient $(s(\alpha),s(\beta))$ dann wie folgt:

$$
\begin{aligned}
s_{ij}(\alpha) \quad &= \quad \sum_{k\in\mathscr{C}} x^k_{ij} - u_{ij}y_{ij} \quad \forall (i,j) \in \mathscr{A} \\
s^k_{ij}(\beta) \quad &= \quad x^k_{ij} - d^k_{ij}y_{ij} \qquad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C}
\end{aligned}
$$

In der **Rucksack-Relaxation** (*knapsack relaxation*) werden die $|\mathscr{C}||\mathscr{N}|$ Flussnebenbedingungen (1) Lagrange-relaxiert, indem ihnen Lagrangemultiplikatoren ω_i^k für alle $i \in \mathscr{N}$ und $k \in \mathscr{C}$ zugeordnet werden. Da es sich hier um Gleichungen handelt, sind die ω_i^k nicht vorzeichenbeschränkt. Die Rucksack-Relaxation lautet damit für ein festes ω:

$$\begin{aligned} z_{KS}(\omega) \quad = \quad & \min \sum_{(i,j)\in\mathscr{A}} \sum_{k\in\mathscr{C}} c_{ij}^k x_{ij}^k + \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{k\in\mathscr{C}} \sum_{i\in\mathscr{N}} \omega_i^k \Big(\sum_{j:(i,j)\in\mathscr{A}} x_{ij}^k - \sum_{j:(j,i)\in\mathscr{A}} x_{ji}^k - b_i^k \Big) \\ & \text{u.d.N.} \quad \sum_{k\in\mathscr{C}} x_{ij}^k \leq u_{ij} y_{ij} \quad \forall (i,j) \in \mathscr{A} \\ & \qquad x_{ij}^k \leq d_{ij}^k y_{ij} \quad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C} \\ & \qquad x_{ij}^k \geq 0 \quad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{C} \\ & \qquad y_{ij} \in \{0,1\} \quad \forall (i,j) \in \mathscr{A} \end{aligned}$$

Wieder formen wir die Zielfunktion um, damit die Struktur des Problems sichtbar wird. Dabei nutzen wir aus, dass b_i^k in unserer Version des CNDP nur zwei Einträge ungleich Null besitzt:

$$\begin{aligned} z_{KS}(\omega) \quad &= \quad \min \sum_{(i,j)\in\mathscr{A}} \left[\sum_{k\in\mathscr{C}} (c_{ij}^k + \omega_i^k - \omega_j^k) x_{ij}^k + f_{ij} y_{ij} \right] - \sum_{k\in\mathscr{C}} d^k (\omega_{O(k)}^k - \omega_{D(k)}^k) \\ &= \quad \min_{y\in\{0,1\}^{|\mathscr{A}|}} \sum_{(i,j)\in\mathscr{A}} (g_{ij}(\omega) + f_{ij}) y_{ij} - \sum_{k\in\mathscr{C}} d^k (\omega_{O(k)}^k - \omega_{D(k)}^k) \end{aligned}$$

Man sieht nun, dass sich das Problem in ein kontinuierliches Rucksackproblem $KS^{(i,j)}(\omega)$ und ein einfaches Entscheidungsproblem für y_{ij} für jede Kante $(i,j) \in \mathscr{A}$ und einen konstanten Anteil in der Zielfunktion aufspalten lässt. Mit $\tilde{c}_{ij}^k = c_{ij}^k + \omega_i^k - \omega_j^k$ lautet das Rucksackproblem $KS^{(i,j)}(\omega)$ für die Kante (i,j):

$$\begin{aligned} z_{KS}^{(i,j)}(\omega) \quad = \quad & \min \sum_{k\in\mathscr{C}} \tilde{c}_{ij}^k x_{ij}^k \\ & \text{u.d.N.} \quad \sum_{k\in\mathscr{C}} x_{ij}^k \leq u_{ij} \\ & \qquad x_{ij}^k \leq d_{ij}^k \quad \forall k \in \mathscr{C} \\ & \qquad x_{ij}^k \geq 0 \quad \forall k \in \mathscr{C} \end{aligned}$$

Ähnlich wie in der Kürzeste-Wege-Relaxation kann man nun *reduzierte Fixkosten* $\tilde{g}_{ij} = z_{KS}^{(i,j)}(\omega) + f_{ij}$ definieren und mit ihnen die Belegung der y-Variablen entscheiden:

$$y_{ij} = \begin{cases} 1 & \text{falls} \quad \tilde{g}_{ij} < 0 \text{ und} \\ 0 & \text{sonst,} \end{cases}$$

Im Fall $y_{ij} = 0$ muss gleichzeitig $x_{ij}^k = 0$ für alle $k \in \mathscr{C}$ gesetzt werden, um eine korrekte Lösung (x,y) zu erhalten. Algorithmus 2 löst das Rucksackproblem $g_{ij}(\omega)$ und liefert eine Teillösung (x_{ij}, y_{ij}) für eine Kante $(i,j) \in \mathscr{A}$. Das zugrunde liegende Prinzip ist einfach: Die Kapazität u_{ij} der Kante wird solange mit dem nächstgünstigsten Gut aufgefüllt, bis sie entweder erschöpft ist, oder es kein Gut mit negativen Kosten mehr gibt. Alternativ zu Zeile 3 könnte man auch zu Beginn die Güter nach aufsteigenden Kosten $\tilde{c}_{ij}^k$ sortieren. Die hier gezeigte Version des Algorithmus hat sich aber bei einem Vergleich als deutlich

Algorithmus 2 Kontinuierliches Rucksackproblem

1: *setze* $\tilde{g}_{ij} = f_{ij}$ *und* $x_{ij}^k = 0 \; \forall k \in \mathscr{C}$ /* löse Rucksackproblem */
2: **repeat**
3: *wähle* $k' = argmin_{k \in \mathscr{C}}(\tilde{c}_{ij}^k)$
4: **if** $(\tilde{c}_{ij}^{k'} < 0)$ **then**
5: **if** $(u_{ij} > d_{ij}^{k'})$ **then**
6: *setze* $x_{ij}^{k'} = d_{ij}^{k'}, \quad u_{ij} = u_{ij} - d_{ij}^{k'}, \quad \tilde{g}_{ij} = \tilde{g}_{ij} + \tilde{c}_{ij}^{k'} x_{ij}^{k'}$ *und* $\tilde{c}_{ij}^{k'} = \infty$
7: **else**
8: *setze* $x_{ij}^{k'} = u_{ij}, \quad u_{ij} = 0$ *und* $\tilde{g}_{ij} = \tilde{g}_{ij} + \tilde{c}_{ij}^{k'} x_{ij}^{k'}$
9: **until** $(\tilde{c}_{ij}^{k'} < 0)$ *UND* $(u_{ij} > 0)$
10: **if** $(\tilde{g}_{ij} < 0)$ **then**
11: *setze* $y_{ij} = 1$ /* setze y_{ij} */
12: **else**
13: *setze* $y_{ij} = 0$ *und* $x_{ij}^k = 0 \; \forall k \in \mathscr{C}$

effizienter erwiesen. Das Lagrange-Multiplikator Problem (Lagrange-Dual) der Rucksack-Relaxation lautet nun:

$$z_{KS} = \max_{\omega \in \mathbb{R}} z_{KS}(\omega),$$

und ein Subgradient $s(\omega)$ lässt sich aus einer Lösung (x,y) wie folgt berechnen:

$$s(\omega_i^k) = \sum_{j:(i,j) \in \mathscr{A}} x_{ij}^k - \sum_{j:(j,i) \in \mathscr{A}} x_{ji}^k - b_i^k.$$

2.5 Lagrange-Relaxation

Für die Berechnung der unteren Schranken im Rahmen des Branch-and-Bound-Verfahrens wird in dieser Arbeit eine Lagrange-Relaxation verwendet. In diesem Abschnitt sollen die Grundzüge dieser Relaxationstechnik diskutiert werden.[2] Hierfür betrachten wir das Optimierungsproblem

$$\begin{aligned} (P) \qquad & \min \quad c^T x \\ & \text{u.d.N. } Ax \geq b \qquad (2.1) \\ & \qquad x \in X, \qquad (2.2) \end{aligned}$$

mit $A \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$, und $c \in \mathbb{R}^n$. Wir nehmen an, dass es sich bei (2.2) um eher einfache Nebenbedingungen handelt, während die Einschränkung (2.1) die Aufgabe stark verkompliziert. Bei der Lagrange-Relaxation werden die „schwierigen“ Nebenbedingungen (2.1) gestrichen und die Zielfunktion so modifiziert, dass eine Verletzung dieser Bedingungen mit höheren Kosten bestraft wird. Zu diesem Zweck fügt man der Zielfunktion für jede weggelassene Ungleichung $\sum_{j=1}^n a_{ij} x_j \geq b_i$ einen Term $\omega_i (b_i - \sum_{j=1}^n a_{ij} x_j)$ mit $\omega_i \geq 0$ hinzu, sodass

[2] Die Darstellung der Grundidee und Eigenschaften der Lagrange-Relaxation orientiert sich an Kapitel II.3 in [Nemhauser and Wolsey, 1988] und Kapitel 16 in [Ahuja et al., 1993].

sich die Minimierungsaufgabe

$$LR(\omega) \qquad \begin{array}{ll} \min & c^T x + \omega^T (b - Ax) \\ \text{u.d.N.} & x \in X \end{array}$$

mit $\omega \in \mathbb{R}^m_+$ ergibt. Man bezeichnet diese Minimierungsaufgabe als eine *Lagrange-Relaxation* von (P) bezüglich der Nebenbedingungen (2.1) und den Vektor ω als einen *Lagrange-Multiplikator*.

Sei $z_{LR}(\omega)$ der optimale Zielfunktionswert von $LR(\omega)$ und z_P die Kosten der optimalen Lösung des originalen Optimierungsproblems (P). Es gilt:

Lemma 2.1 *Für jede für das Ausgangsproblem (P) zulässige Lösung x' und alle $\omega \in \mathbb{R}^m_+$ ist $z_{LR}(\omega) \leq c^T x'$.*

Beweis: Wenn x' eine zulässige Lösung von (P) darstellt, ist sie für $LR(\omega)$ ebenfalls zulässig und somit gilt $z_{LR}(\omega) \leq c^T x' + \omega^T (b - Ax')$. Wegen (2.1) und $\omega \geq 0$ ist $\omega^T (b - Ax') \leq 0$. Es folgt $z_{LR}(\omega) \leq c^T x'$. □

Weil dieses Lemma insbesondere auch für die optimale Lösung von (P) wahr ist, gilt $z_{LR}(\omega) \leq z_P$ für alle $\omega \in \mathbb{R}^m_+$. Um die bestmögliche untere Schranke für z_P zu berechnen, müsste man noch den Strafkostenvektor ω finden, der $z_{LR}(\omega)$ maximiert:

$$z_{LM} = \max_{\omega \geq 0} z_{LR}(\omega) \tag{2.3}$$

Diese Maximierungsaufgabe wird im Weiteren als ein mit (P) assoziiertes *Lagrange-Multiplikator-Problem* bezeichnet. Im günstigsten Fall stimmt der optimale Zielfunktionswert des Lagrange-Multiplikator-Problems mit z_P überein. Ist dies nicht der Fall, spricht man von einer *Dualitätslücke* $z_P - z_{LM}$.

Wenn es sich bei den „schwierigen" Nebenbedingungen von (P) nicht um Ungleichungen, sondern um Gleichungen handelt, entfällt die Forderung $\omega \geq 0$ in dem Lagrange-Multiplikator-Problem. Um dies zu zeigen, betrachte das Optimierungsproblem

$$\begin{array}{ll} \min & c^T x \\ \text{u.d.N.} & Ax = b \\ & x \in X. \end{array}$$

Die Gleichungen $Ax = b$ lassen sich äquivalent als ein System von Ungleichungen $Ax \geq b$ und $-Ax \geq -b$ darstellen. Wird nun eine Lagrange-Relaxation durchgeführt, erhält man eine Optimierungsaufgabe

$$\begin{array}{ll} \min & c^T x + (\omega^+ - \omega^-)^T (b - Ax) \\ \text{u.d.N.} & x \in X \end{array}$$

mit $\omega^+, \omega^- \geq 0$. Ersetzt man die Differenz $\omega^+ - \omega^-$ durch einen Vektor ω, bekommt man einen Lagrange-Multiplikator, der sowohl positive als auch negative Einträge haben kann.

2.5.1 Eigenschaften der Lagrange-Relaxation

In diesem Abschnitt werden einige wichtige Eigenschaften der Lagrange-Relaxation wie beispielsweise die Güte der mit dieser Technik berechenbaren unterer Schranke diskutiert.

Wir betrachten erneut die Lagrange-Relaxation $LR(\omega)$ des Optimierungsproblems (P) und setzen im Weiteren voraus, dass die Menge X als

$$X = \{x \in \mathbb{R}^n_+ : Dx \geq q, x_1, \ldots, x_r \text{ sind ganzzahlig}\}$$

definiert und *beschränkt* ist. In diesem Fall stellt die konvexe Hülle[3] von X ein konvexes Polytop dar, dessen Ecken $\{x^i\}_{i=1}^K$ eine Teilmenge von X bilden. Bezeichne $conv(X)$ die konvexe Hülle von X. Für jede Wahl des Vektors ω nimmt die Zielfunktion $c^T x + \omega^T(b - Ax)$ ihr Minimum auf $conv(X)$ in einer Ecke $x^* \in \{x^i\}_{i=1}^K$ an, d.h. in einem Punkt aus X. Wegen $X \subseteq conv(X)$ stellt dieser Punkt x^* gleichzeitig eine optimale Lösung des Lagrange-Teilproblems

$$z_{LR}(\omega) = \min\{c^T x + \omega^T(b - Ax) : x \in X\}$$

dar. Somit kann die Suche nach der optimalen Lösung eines Lagrange-Teilproblems $LR(\omega)$ auf die Ecken der konvexen Hülle von X beschränkt werden:

$$z_{LR}(\omega) = \min_{i=1,\ldots,K} c^T x^i + \omega^T(b - Ax^i). \tag{2.4}$$

Die Gleichung (2.4) zeigt die Abhängigkeit der optimalen Zielfunktionswerte einzelner Lagrange-Relaxationen von der Wahl des Lagrange-Multiplikators ω. Für jeden Punkt x^i hängen die Kosten $z(\omega, x^i) = c^T x^i + \omega^T(b - Ax^i)$ linear von ω ab, sodass $z_{LR}(\omega)$ das Minimum einer endlichen Menge linearer Funktionen darstellt und somit selbst eine stückweise lineare konkave Funktion ist. Die Abbildung 2.3 zeigt einen möglichen Verlauf der Lagrange-Funktion $z_{LR}(\omega)$ für den eindimensionalen Fall $\omega \in \mathbb{R}$.

Das Lagrange-Multiplikator-Problem ist das Maximieren der Lagrange-Funktion $z_{LR}(\omega)$ zur Bestimmung einer möglichst guten unteren Schranke für z_P. Im Abschnitt 2.6 werden einige der bekanntesten Lösungsansätze für dieses Problem behandelt. Nun soll die Frage nach der Güte der mit diesen Verfahren berechenbaren unteren Schranke geklärt werden.

Für den optimalen Zielfunktionswert z_{LM} des Lagrange-Multiplikator-Problems gilt:

Satz 2.2 $z_{LM} = \min\{c^T x : Ax \geq b, x \in conv(X)\}$

Beweis:[4] Die Gleichung (2.4) beschreibt $z_{LR}(\omega)$ als das Minimum einer endlichen Menge linearer Funktionen und ist offenbar äquivalent zu

$$z_{LR}(\omega) = \max_v \left\{v : v \leq c^T x^i + \omega^T(b - Ax^i) \quad (i = 1, \ldots, K)\right\}.$$

Mit dieser Definition von $z_{LR}(\omega)$ lässt sich das Lagrange-Multiplikator-Problem (2.3) als eine lineare Optimierungsaufgabe formulieren:

$$\begin{aligned} z_{LM} = \quad & \max \quad v \\ & \text{u.d.N. } v + \omega^T(Ax^i - b) \leq c^T x^i \qquad (i = 1, \ldots, K) \\ & \qquad\qquad \omega \geq 0 \end{aligned} \tag{2.5}$$

[3] Das ist die Menge aller Konvexkombinationen der Elemente aus X und die kleinste konvexe Menge des $\mathbb{R}^n$, die die Menge X enthält.

[4] Dies ist eine vereinfachte Version des Beweises von [Nemhauser and Wolsey, 1988].

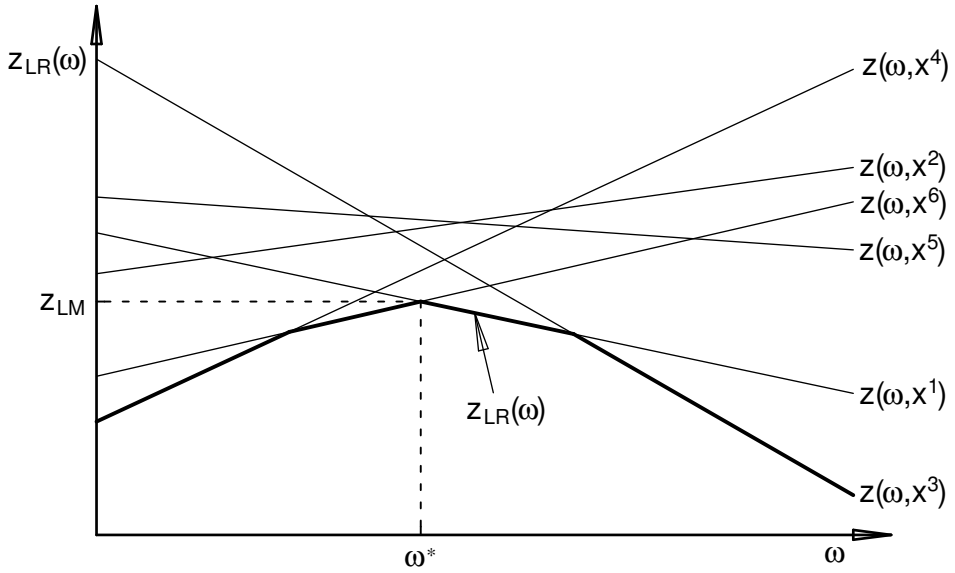


Abbildung 2.3: Lagrange-Funktion $z_{LR}(\omega)$

Hier ist $\{x^i\}_{i=1}^K$ die Menge der Ecken der konvexen Hülle von X. Das zu (2.5) duale Minimierungsproblem heißt:

$$\begin{array}{ll} \min & c^T \sum_{i=1}^K x^i \alpha_i \\ \text{u.d.N.} & A\sum_{i=1}^K x^i \alpha_i \geq b \sum_{i=1}^K \alpha_i \\ & \sum_{i=1}^K \alpha_i = 1 \\ & \alpha \geq 0. \end{array} \tag{2.6}$$

Da die beiden Aufgaben (2.5) und (2.6) laut dem Dualitätstheorem der linearen Optimierung die gleichen optimalen Zielfunktionswerte haben und (2.6) wegen $conv(X) = \{\sum_{i=1}^K x^i \alpha_i : \sum_{i=1}^K \alpha_i = 1, \alpha \geq 0\}$ äquivalent zu

$$\min\left\{c^T x : Ax \geq b,\, x \in conv(X)\right\}$$

ist, folgt der Satz. □

Eine Alternative zur Verwendung der Lagrange-Relaxation ist das Berechnen einer unteren Schranke für z_P durch die Lösung der LP-Relaxation des Problems (P):

$$(LP) \qquad z_{LP} = \min\{c^T x : x \in \mathbb{R}^n_+, Ax \geq b, Dx \geq q\}.$$

Es ist interessant, die beiden Schranken z_{LM} und z_{LP} zu vergleichen. Aus dem Satz 2.2 und $conv\{x \in \mathbb{R}^n_+ : Dx \geq q, x_1, \ldots, x_r \text{ sind ganzzahlig}\} \subseteq \{x \in \mathbb{R}^n_+ : Dx \geq q\}$ folgt

$$z_{LP} \leq z_{LM} \leq z_P.$$

Das heißt die optimale Lösung z_{LM} des Lagrange-Multiplikator-Problems liefert immer eine mindestens genauso gute untere Schranke für z_P wie die LP-Relaxation.

2.6 Lösung des Lagrange-Multiplikator-Problems

Das Lagrange-Multiplikator-Problem, einen Vektor $\omega \in \mathbb{R}_+^m$ zu finden, der die konkave nicht überall differenzierbare Lagrange-Funktion $z_{LR} : \mathbb{R}^m \mapsto \mathbb{R}$ maximiert, ist eine typische Aufgabe der nichtdifferenzierbaren Optimierung. In diesem Abschnitt werden einige der bekanntesten Verfahren für diese Klasse von Optimierungsproblemen behandelt.

Sei $z : \mathbb{R}^m \mapsto \mathbb{R}$ eine konkave, nicht überall differenzierbare Funktion. Man nennt einen Vektor $s \in \mathbb{R}^m$ einen *Subgradienten* von z an der Stelle $\bar{\omega} \in \mathbb{R}^m$, wenn für alle $\omega \in \mathbb{R}^m$ gilt:

$$z(\omega) \leq z(\bar{\omega}) + (\omega - \bar{\omega})^T s. \tag{2.7}$$

Geometrisch gesehen, ist ein Subgradient an der Stelle $\bar{\omega}$ der Gradient einer Hyperebene, die die Funktion z in dem Punkt $\bar{\omega}$ berührt und nicht durch den Raum unterhalb von z geht. Die Abbildung 2.4 zeigt zwei solche Hyperebenen mit den entsprechenden Subgradienten in einem Punkt, wo z einen Knick hat. Die Menge aller Subgradienten von z an der Stelle $\bar{\omega}$ bezeichnet man als ein *Subdifferenzial* $\partial z(\bar{\omega})$. In den Punkten, in denen z differenzierbar ist, stellt der *Gradient*

$$\nabla z(\bar{\omega}) := (\partial z/\partial \omega_1(\bar{\omega}), \ldots, \partial z/\partial \omega_m(\bar{\omega}))^T$$

den einzigen Subgradienten der Funktion z dar.

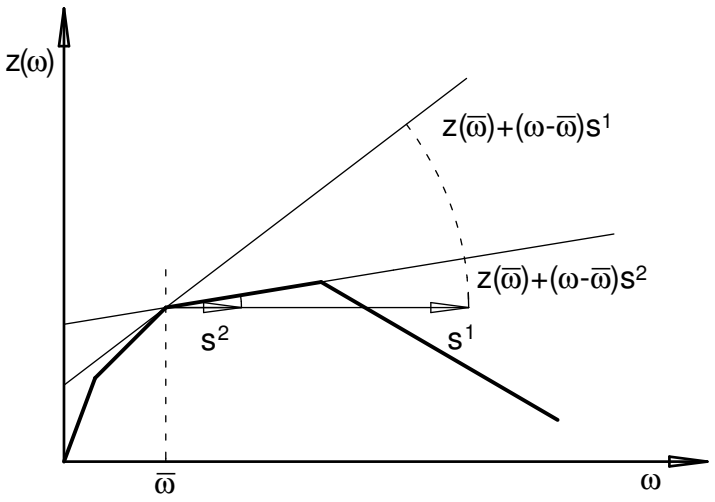


Abbildung 2.4: Zwei Subgradienten s_1 und s_2 in einem Knickpunkt

In Unterschied zu dem Gradienten, der immer die Richtung des steilsten Anstiegs von z in einer kleinen Nachbarschaft von $\bar{\omega}$ zeigt, ist ein Subgradient nicht notwendigerweise eine Verbesserungsrichtung (Abbildung 2.5). Trotzdem liefert er genug Information für die Suche nach dem Maximum einer konkaven Funktion. Denn, wie man der Subgradientenungleichung (2.7) entnimmt, alle Punkte, die besser als $\bar{\omega}$ sind, liegen in dem durch

$$(\omega - \bar{\omega})^T s > 0$$

definierten Halbraum. (Das ist die schraffierte Fläche in der Abbildung 2.5.)

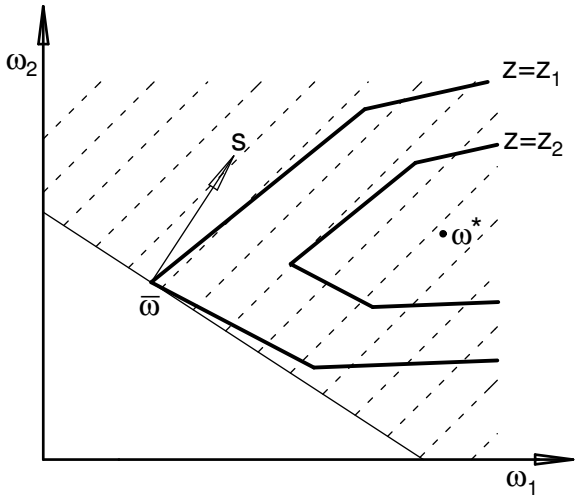


Abbildung 2.5: In einem Punkt, in dem z nicht differenzierbar ist, ist ein Subgradient nicht immer eine Verbesserungsrichtung. Der Verlauf von z ist durch Niveaulinien und die optimale Lösung ω^* dargestellt.

Betrachte nun die Lagrange-Funktion $z_{LR} : \mathbb{R}^m \mapsto \mathbb{R}$. Sei $\bar{x}$ die optimale Lösung eines Lagrange-Teilproblems

$$z_{LR}(\bar{\omega}) = \min\{c^T x + \bar{\omega}^T (b - Ax) : x \in X\}.$$

Weil die Kosten $c^T\bar{x} + \omega^T(b - A\bar{x})$ von $\bar{x}$ für alle $\omega \in \mathbb{R}^m$ eine obere Schranke für den Wert $z_{LR}(\omega)$ darstellen und $z_{LR}(\bar{\omega}) = c^T\bar{x} + \bar{\omega}^T(b - A\bar{x})$ ist, gilt

$$z_{LR}(\omega) \leq c^T\bar{x} + \omega^T(b - A\bar{x}) = z_{LR}(\bar{\omega}) + (\omega - \bar{\omega})^T(b - A\bar{x})$$

für alle $\omega \in \mathbb{R}^m$. Das heißt der Vektor $b - A\bar{x}$ stellt einen Subgradienten der Lagrange-Funktion z_{LR} an der Stelle $\bar{\omega}$ dar.

In dieser Arbeit wird zur Lösung des Lagrange-Multiplikator-Problems neben dem Subgradientenverfahren ein von Antonio Frangioni beschriebener und implementierter Algorithmus eingesetzt [Frangioni, 1997]. Dieser Algorithmus gehört zu der Familie der Bundle-Verfahren. Er sammelt die Information über die Werte und Subgradienten der zu maximierenden Funktion z, um ein Modell dieser Funktion aufzubauen, und verwendet das Modell für die Suche nach dem Extremum von z.

Die Bundle-Verfahren sind von Lemaréchal vorgeschlagen worden [Lemaréchal, 1978] und wurden in den nächsten 20 Jahren zu einem wichtigen Forschungsgebiet der nichtdifferenzierbaren Optimierung ([Kiwiel, 1985], [Lemaréchal et al., 1995]). Heute haben sie einen soliden theoretischen Hintergrund und sind als eine der wichtigsten praktisch anwendbaren Methoden zur Lösung großer nichtdifferenzierbarer Optimierungsprobleme weit anerkannt. Dennoch werden Bundle-Verfahren nur selten angewendet, weil man in der Praxis meist den weniger effizienten aber wesentlich einfacher zu implementierenden Subgradientenalgorithmus bevorzugt. Wir beschreiben in den nächsten Abschnitten beide Verfahren zur Lösung des Lagrange-Multiplikator-Problems.

2.6.1 Subgradientenverfahren

Das Subgradientenverfahren ist ein iteratives Suchverfahren, um nichtlineare und nichtdifferenzierbare Funktionen zu optimieren. Gestartet wird mit einer Initiallösung ω^1, die dann in jeder Iteration mit Hilfe einer Suchrichtung d und einer Schrittweite t aktualisiert wird, bis ein Abbruchkriterium erfüllt ist. Das Grundkonzept ist in Algorithmus 3 dargestellt:

Algorithmus 3 Subgradientenverfahren

1: *initialisiere* ω^1
2: $k = 1$
3: **while** *kein Abbruch* **do**
4: *berechne Suchrichtung* d^k
5: *berechne Schrittweite* t^k
6: $\omega^{k+1} = \omega^k + t^k d^k$
7: $k = k+1$
8: **return** ω^k

Die Wahl der Suchrichtung, der Schrittweite und der Abbruchkriterien ist entscheidend für Effizienz, Konvergenzverhalten und Lösungsqualität des Verfahrens.

Die Suchrichtung d

Für die Berechnung der Suchrichtung d^k wird zunächst ein *Subgradient* s^k an der Stelle ω^k ermittelt, der eine Richtung repräsentiert, in der in einer gewissen Nachbarschaft der aktuellen Lösung die euklidische Distanz zur optimalen Lösung verringert wird. Bei der Anwendung des Subgradientenverfahren für das Lagrange-Dual wird s^k auf den Gradienten von $z_{LR}(\omega^k)$ gesetzt, der immer auch ein Subgradient ist:

$$s^k = b - Ax^k,$$

wobei x^k die optimale Lösung von $LR(\omega^k)$ ist, die meist schnell durch die Lösung von einfachen Unterproblemen (z.B. Kürzeste-Wege-Probleme) berechnet werden kann. An den Stellen, an denen $z_{LR}(\omega^k)$ differenzierbar ist ($LR(\omega^k)$ besitzt eine eindeutige optimale Lösung), ist s eine echt verbessernde Richtung (*ascent direction*) in einer Nachbarschaft von ω^k. An den Stellen jedoch, an denen $z_{LR}(\omega^k)$ nicht differenzierbar ist ($LR(\omega^k)$ besitzt mehr als eine optimale Lösung), kann keine Verbesserung der Zielfunktion in der Richtung s, sondern nur die euklidische Annäherung an die optimale Lösung garantiert werden. Diese Schwäche kann behoben werden, indem in jeder Iteration ein quadratisches Optimierungsproblem gelöst wird, dass die beste (und immer echt verbessernde) Richtung s liefert. In der Praxis wird jedoch aus Effizienzgründen darauf verzichtet. In den frühen Versionen (vgl. [M.Held et al., 1974], [Polyak, 1969]) des Verfahrens wird nun

$$d^k = s^k$$

gesetzt. Zu dieser Wahl werden in der Literatur unzählige Alternativen untersucht und im Hinblick auf Konvergenzverhalten und Lösungsqualität bewertet. Schnell hat sich die Erkenntnis durchgesetzt, dass es günstig ist, bei der Neuberechnung von d^k die Suchrichtung

der letzten Iteration zu berücksichtigen (vgl. [Crowder, 1976]):

$$d^k = \theta^k d^{k-1} + (1-\theta^k)s^k$$

Die Wahl von θ bestimmt die Gewichtung der letzten Suchrichtung und wird in der Standardvariante auf einen konstanten Wert $0 \leq \theta \leq 1$ gesetzt (sog. *Crowder Regel*).

In [Camerini et al., 1975] schlagen die Autoren eine variable, selbst justierende Wahl von θ^k vor:

$$\theta^k = \begin{cases} \|s^k\|/\|d^{k-1}\|, & \text{falls} \quad s^k d^{k-1} \leq 0, \\ 0, & \text{sonst.} \end{cases}$$

Diese sog. *modifizierte Camerini-Fratta-Maffioli Regel* garantiert, dass die neue Suchrichtung d^k mindestens so gut ist wie der Subgradient s^k. Diese Regel wird auch in [Crainic et al., 2001a] empfohlen. Holmberg und Yuan setzen in [Holmberg and Yuan, 2000] dagegen mit gutem Erfolg folgende Variante der Crowder Regel ein, die auch in unseren Experimenten zu guten Ergebnissen geführt hat:

$$d^k = (s^k + \theta d^{k-1})/(1+\theta).$$

Die Schrittweite t

Auch für die Berechnung der Schrittweite gibt es verschiedene Varianten. Für die Wahl von t^k nach einer divergenten Reihe, für die gilt

$$\sum_{k=1}^{\infty} t^k \to \infty, \lim_{k\to\infty} t^k \to 0$$

ist in der Theorie die Konvergenz des Verfahrens zur optimalen Lösung nachgewiesen (vgl. [Allen et al., 1987], [Hiriart-Urruty and Lemaréchal, 1993], [Lemaréchal, 1989]). Praktisch zeigt sich bei dieser Wahl der Schrittweite allerdings ein sehr langsames Konvergenzverhalten. In der Praxis wird t^k deshalb nach einer geometrischen Reihe bestimmt, wobei versucht wird, die euklidische Entfernung zur optimalen Lösung zu berücksichtigen. Sei z_{UB} eine obere Schranke für P, dann berechnet man die Schrittweite folgendermaßen:

$$t^k = \lambda_k \frac{z_{UB} - z_{LR}(\omega^k)}{\|d^k\|^2}.$$

Dabei ist λ_k ein Skalierungsparameter, der in der Regel verkleinert wird, wenn das Verfahren in einer Reihe von aufeinander folgenden Iterationen keine Verbesserung des Zielfunktionswertes erreicht (*stalling*). Um Konvergenz zu gewährleisten, sollte $0 \leq \lambda_k \leq 2$ sein. Statt z_{UB} wird die Verwendung von ηz_{UB} mit $\eta \approx 1.05$ empfohlen, um bei einer Annäherung an die optimale Lösung eine zu kleine Wahl von t_k zu verhindern, was zu vielen unnötigen Iterationen führen kann.

Abbruchkriterien

Wie bei der Wahl der Suchrichtung gibt es auch bei den Abbruchbedingungen solche, die dem Verfahren theoretisch zugrunde liegen, und solche, die sich in der Praxis bewährt haben. In der Literatur werden die folgenden Kriterien empfohlen, die auch allesamt in unserem Optimierungssystem umgesetzt sind:

1. $s^k = 0$
2. $\|d^k\| < \varepsilon$
3. $t^k < \varepsilon$
4. $k > M$
5. $z_{UB} - z_{LR}(\omega^k) < \varepsilon$

Wenn der Subgradient gleich Null ist (Bedingung 1), dann ist mit ω^k die dual optimale Lösung gefunden worden. In diesem Fall ist auch die bei der Lösung von $LR(\omega^k)$ abfallende primale Lösung x^k zulässig für P und damit optimal. Aus numerischen Gründen und weil die Relaxation oft einen duality gap besitzt, tritt dieser Fall in der Praxis jedoch so gut wie nie ein.

Häufiger bricht die Suche ab, wenn die Norm der Suchrichtung (Bedingung 2) oder die Schrittweite (Bedingung 3) einen Wert nahe Null erreichen. Von weiteren Iterationen sind dann keine großen Verbesserungen von z_{LR} mehr zu erwarten. Die Wahl von ε legt die Genauigkeit der Lösung fest und bestimmt auch maßgeblich die Anzahl der durchgeführten Iterationen. Bei kleinem ε können also in manchen Fällen immer noch zu viele unnötige Iterationen durchgeführt werden, was daher mit einer Iterationsobergrenze M (Bedingung 4) verhindert wird. Für die Wahl von M und ε sind einige Sorgfalt und am besten eine Reihe von Testläufen nötig, denn eine zu kleine Wahl von ε führt auch bei einfachen Probleminstanzen zu vielen überflüssigen Iterationen, und ein zu kleiner Wert für M kann die Qualität der Lösung deutlich verschlechtern. Bedingung 5 darf vor allem nicht fehlen, wenn das Verfahren in einem Branch-and-Bound Rahmen eingesetzt wird, denn dort führt die Erfüllung dieser Bedingung sofort zum Abschneiden des entsprechenden Astes im Branch-and-Bound Baum.

2.6.2 Bundle-Verfahren

Ähnlich dem Subgradientenverfahren versucht man in jeder Iteration eines Bundle-Algorithmus in einer Umgebung der aktuellen Lösung $\bar{\omega}$ einen besseren Punkt ω zu finden. Doch während ein Subgradientenverfahren hierfür nur den aktuellen Subgradienten und eventuell die Richtung der letzten Iteration benutzt, speichern Bundle-Verfahren die in den vorangegangenen Iterationen generierten Subgradienten in einem so genannten *Bündel* (bundle) und benutzen sie mit, um die Suchrichtung zu bestimmen. Wird durch einen Schritt in diese Richtung eine (deutlich) bessere Lösung ω gefunden, wird sie als der neue aktuelle Punkt $\bar{\omega}$ akzeptiert. Andernfalls benutzt man einfach die neue Information, um das Bündel zu erweitern und in der nächsten Iteration eine andere und hoffentlich bessere Suchrichtung zu finden.[5]

Suchrichtung

Betrachte die folgende Situation: In den vorangegangenen Iterationen wurden für die Punkte $\omega^1, \ldots, \omega^k$ die Werte der Funktion z samt Subgradienten berechnet. Die Menge der aktuell

[5] Die Darstellung des Bundle-Verfahrens ist an [Frangioni, 1997], [Carraresi et al., 1995] und [Crainic et al., 2001b] angelehnt.

verfügbaren Informationen ist ein Bündel $\beta = \{\langle \omega^i, z^i, s^i \rangle : i \in I_\beta\}$ mit $z^i = z(\omega^i)$, $s^i \in \partial z(\omega^i)$ und $I_\beta \subseteq \{1,\ldots,k\}$. Jedem Tripel $\langle \omega^i, z^i, s^i \rangle \in \beta$ entspricht eine *Linearisierung* von z in ω^i

$$\overline{z}_i(\omega) = z^i + (\omega - \omega^i)^T s^i$$

mit $\overline{z}_i(\omega) \geq z(\omega)$ für alle $\omega \in R^m$. Die durch das Minimum dieser Funktionen definierte stückweise lineare Funktion

$$\hat{z}_\beta(\omega) = \min_{i \in I_\beta} \overline{z}_i(\omega)$$

ist das aktuelle *Schnittebenenmodell* von z. Die Funktion $\hat{z}_\beta$ nähert z von oben an und hat zumindest in den Punkten ω^i, $i \in I_\beta$ die gleichen Werte wie z (Abbildung 2.6).

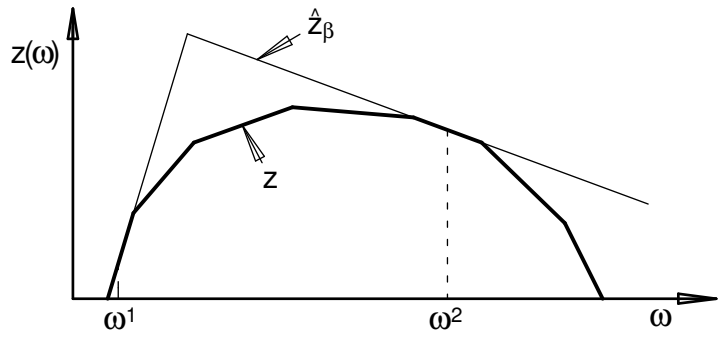


Abbildung 2.6: Ein Schnittebenenmodell von z

Sucht man nun das Maximum der Funktion z, scheint es eine natürliche Wahl, als Nächstes immer den Punkt zu nehmen, in dem das aktuelle Modell $\hat{z}_\beta$ den größten Wert hat. Diese Vorgehensweise führt zu dem klassischen *Schnittebenenverfahren* [Kelley, 1960] und hat einige Schwachstellen. Zum einen besitzt das Schnittebenenmodell $\hat{z}_\beta$ nicht immer ein Extremum. Vor allem am Anfang des Verfahrens, wenn β nur wenige Subgradienten hat, ist diese Funktion häufig nicht nach oben beschränkt. Außerdem neigt das Schnittebenenverfahren dazu, Punkte zu wählen, die weit von den im Bündel gespeicherten Punkten ω^i, $i \in I_\beta$ liegen, das heißt in einer Gegend, wo $\hat{z}_\beta$ vermutlich keine gute Approximation von z ist. [Kiwiel, 1985]

Bei der Bundle-Methode versucht man den Schwächen des Schnittebenenverfahrens entgegenzuwirken, indem man $\hat{z}_\beta$ nur in einer Nachbarschaft der aktuellen Lösung $\overline{\omega}$ maximiert oder (wie hier) größere Entfernungen von diesem Punkt durch das Einführen von Strafkosten vermeidet. Das in dieser Arbeit verwendete Verfahren legt den nächsten zu untersuchenden Punkt $\omega = \overline{\omega} + d$ durch die Lösung des quadratischen Optimierungsproblems

$$\max_{d \in R^m} \left\{ \hat{z}_\beta(\overline{\omega} + d) - \frac{1}{2t}\|d\|^2 \right\} \tag{2.8}$$

fest. Der Parameter $t > 0$ bestimmt die Höhe der Strafkosten für die Entfernung von der aktuellen Lösung $\overline{\omega}$. Er beeinflusst die Schrittweite und somit auch das Konvergenzverhalten des Algorithmus.

Später wird auf die Lösung der Optimierungsaufgabe (2.8) etwas genauer eingegangen. Doch zuerst soll das dem Bundle-Verfahren zugrunde liegende Konzept des approximativen Subgradienten erörtert werden. Ein Vektor $s \in R^m$ ist ein ε-*Subgradient* einer konkaven

Funktion $z : R^m \mapsto R$ in $\omega' \in R^m$, wenn er die Bedingung

$$z(\omega) \leq z(\omega') + (\omega - \omega')^T s + \varepsilon \qquad \forall \omega \in R^m$$

erfüllt, das heißt wenn er den Gradienten einer oberhalb von z verlaufenden Hyperebene darstellt, die in dem Punkt ω' in einem Abstand von höchstens ε über der Funktion z liegt (Abbildung 2.7). Die Menge aller ε-Subgradienten der Funktion z an der Stelle ω' ist das *ε-Subdifferenzial*, bezeichnet mit $\partial_\varepsilon z(\omega')$. Ein Punkt ω' ist ε-optimal für z genau dann, wenn $\partial_\varepsilon z(\omega')$ den Vektor 0 enthält.

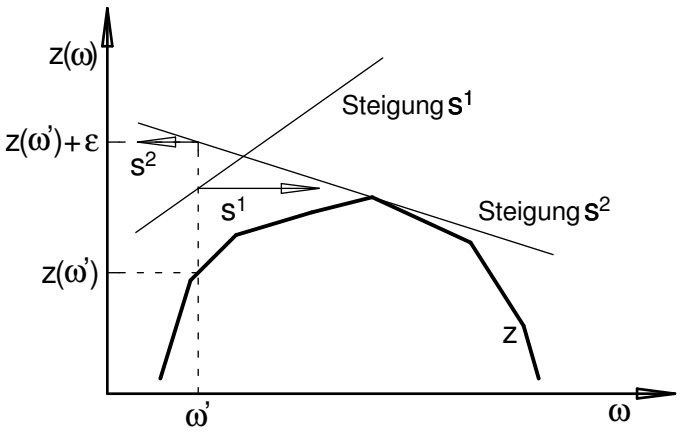


Abbildung 2.7: Zwei ε-Subgradienten von z an der Stelle $\bar{\omega}$

Der Vorteil der Verwendung approximativer Subgradienten ist, dass man die gesammelte Information über das Verhalten von z in der Umgebung eines Punktes ω' leicht auf jeden anderen Punkt ω übertragen kann. Es gilt:

$$s \in \partial_\varepsilon z(\omega') \quad \Rightarrow \quad s \in \partial_\mu z(\omega) \quad \forall \omega \, \forall \mu \geq z(\omega') + (\omega - \omega')^T s - z(\omega) + \varepsilon.$$

Insbesondere lässt sich auch für jeden Subgradienten s^i im Bündel β ein Wert

$$\alpha^i = z^i + (\bar{\omega} - \omega^i)^T s^i - z(\bar{\omega}) = \bar{z}_i(\bar{\omega}) - z(\bar{\omega})$$

angeben, sodass der Vektor s^i einen α^i-Subgradienten von z in dem aktuellen Punkt $\bar{\omega}$ darstellt. Die Größe α^i gibt den in $\bar{\omega}$ vorliegenden Abstand zwischen der echten Funktion z und der durch $\langle \omega^i, z^i, s^i \rangle$ definierten Linearisierung dieser Funktion an (Abbildung 2.8) und wird deswegen als ein *Linearisierungsfehler* bezeichnet. Drückt man die Tatsache $s^i \in \partial_{\alpha^i} z(\bar{\omega})$, $i \in I_\beta$ durch

$$z(\omega) \leq z(\bar{\omega}) + (\omega - \bar{\omega})^T s^i + \alpha^i \qquad \forall i \in I_\beta, \quad \forall \omega \in R^m$$

aus, ergibt sich mit einer Konvexkombination

$$z(\omega) \leq z(\bar{\omega}) + (\omega - \bar{\omega})^T \textstyle\sum_{i \in I_\beta} s^i \theta_i + \sum_{i \in I_\beta} \alpha^i \theta_i \qquad \forall \omega \in R^m \tag{2.9}$$

für alle θ aus $\Theta = \{\theta : \sum_{i \in I_\beta} \theta_i = 1, \theta \geq 0\}$. Somit stellt $\sum_{i \in I_\beta} s^i \theta_i$ für alle $\theta \in \Theta$ einen $\sum_{i \in I_\beta} \alpha^i \theta_i$-Subgradienten von z in dem aktuellen Punkt $\bar{\omega}$ dar.

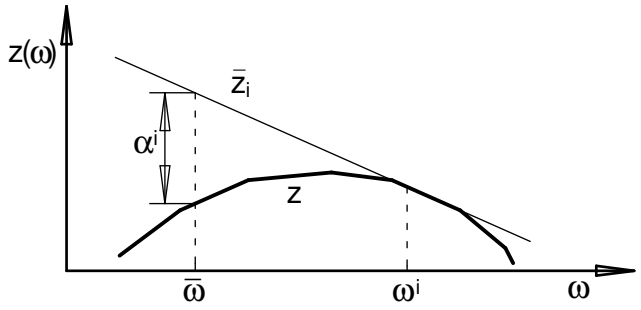


Abbildung 2.8: Linearisierungsfehler α^i

Im Folgenden wird gezeigt, dass sich die Maximierungsaufgabe (2.8) auf die Lösung des quadratischen Optimierungsproblems

$$(\Delta_{\beta t}) \qquad \min\left\{\frac{t}{2}\|\sum_{i\in I_\beta} s^i\theta_i\|^2 + \sum_{i\in I_\beta}\alpha^i\theta_i : \theta\in\Theta\right\}$$

zurückführen lässt. Mit (2.9) kann dieses Problem wie folgt interpretiert werden: Gesucht ist eine Hyperebene mit einer möglichst kleinen Norm des Gradienten $\sum_{i\in I_\beta} s^i\theta_i$, die oberhalb von z und deren Schnittebenenmodell $\hat{z}_\beta$ verläuft und an der Stelle $\bar{\omega}$ einen möglichst kleinen Abstand $\sum_{i\in I_\beta}\alpha^i\theta_i$ zur Funktion z hat. Man betrachtet diese Hyperebene als eine Linearisierung von z in einer Umgebung der aktuellen Lösung $\bar{\omega}$ und orientiert sich bei der Wahl der Suchrichtung d an ihren Gradienten.

Betrachte wieder das Optimierungsproblem (2.8). Mit dem oben definierten Linearisierungsfehler α^i ist

$$\hat{z}_\beta(\bar{\omega}+d) = \min_{i\in I_\beta}\left\{z^i + (\bar{\omega}+d-\omega^i)^T s^i\right\} = \min_{i\in I_\beta}\left\{d^T s^i + \alpha^i\right\} + z(\bar{\omega}).$$

Folglich lässt sich die Optimierungsaufgabe (2.8) äquivalent als

$$(\Pi_{\beta t}) \qquad \max_{d,v}\left\{v - \frac{1}{2t}\|d\|^2 : v \le d^T s^i + \alpha^i,\, i\in I_\beta\right\}[+z(\bar{\omega})]$$

formulieren.

Behauptung 2.3 *Ist $\theta^*\in\Theta$ die optimale Lösung von $(\Delta_{\beta t})$, kann die optimale Lösung (d^*, v^*) der Maximierungsaufgabe $(\Pi_{\beta t})$ durch*

$$d^* = t\sum_{i\in I_\beta} s^i\theta_i^*, \qquad v^* = t\|\sum_{i\in I_\beta} s^i\theta_i^*\|^2 + \sum_{i\in I_\beta}\alpha^i\theta_i^* \tag{2.10}$$

berechnet werden.

Um dies zu zeigen, benötigt man einige Ergebnisse aus der Theorie der nichtlinearen Optimierung.[6] Bei der Bestimmung von Extremwerten von Funktionen mit Nebenbedingungen

[6] Ihre Darstellung ist an [Neumann and Morlock, 1993] angelehnt.

können die Restriktionen in die Zielfunktion mit einbezogen werden. Hierbei ergibt sich für das Optimierungsproblem

$$\max\{f(x) : g_i(x) \geq 0,\ i = 1,\ldots,m\} \tag{2.11}$$

eine Lagrange-Funktion

$$F(x,u) = f(x) + \sum_{i=1}^{m} u_i g_i(x).$$

Ein Punkt (x^*,u^*) mit $u^* \geq 0$ ist ein *Sattelpunkt* der Lagrange-Funktion F, wenn für alle x und alle $u \geq 0$ gilt:

$$F(x,u^*) \leq F(x^*,u^*) \leq F(x^*,u)$$

Satz 2.4 *Ist (x^*,u^*) mit $u^* \geq 0$ ein Sattelpunkt von F, so ist x^* eine optimale Lösung von (2.11). [Neumann and Morlock, 1993]*

Beweis der Behauptung 2.3: Bezeichne θ^* die optimale Lösung von $(\Delta_{\beta t})$ und seien d^* und v^* wie in (2.10) festgelegt. Zunächst wird gezeigt, dass (d^*,v^*) die Nebenbedingungen des Optimierungsproblems $(\Pi_{\beta t})$

$$d^{*T}s^i + \alpha^i \geq v^* \quad \text{für alle } i \in I_\beta \tag{2.12}$$

erfüllt und folglich eine zulässige Lösung von $(\Pi_{\beta t})$ darstellt.
Annahme: Es gibt ein $k \in I_\beta$ mit $d^{*T}s^k + \alpha^k < v^*$.
Für ein $\gamma \in [0,1]$ definiere $\theta^{(\gamma)}$ durch $\theta_i^{(\gamma)} = (1-\gamma)\theta_i^*$ für $i \in I_\beta \setminus \{k\}$ und $\theta_k^{(\gamma)} = \gamma + (1-\gamma)\theta_k^*$. Wegen

$$\sum_{i\in I_\beta} \theta^{(\gamma)} = \gamma + \sum_{i\in I_\beta}(1-\gamma)\theta_i^* = 1$$

stellt $\theta^{(\gamma)}$ für alle $\gamma \in [0,1]$ eine zulässige Lösung von $(\Delta_{\beta t})$ dar. Bezeichne $c(\gamma)$ den Zielfunktionswert von $\theta^{(\gamma)}$

$$c(\gamma) \;=\; \frac{t}{2}\|\gamma s^k + \sum_{i\in I_\beta} s^i(1-\gamma)\theta_i^*\|^2 + \gamma\alpha^k + \sum_{i\in I_\beta}\alpha^i(1-\gamma)\theta_i^*.$$

Die erste Ableitung von $c(\gamma)$ lautet

$$c'(\gamma) = t(\gamma s^k + \sum_{i\in I_\beta} s^i(1-\gamma)\theta_i^*)^T(s^k - \sum_{i\in I_\beta} s^i\theta_i^*) + \alpha^k - \sum_{i\in I_\beta}\alpha^i\theta_i^*.$$

Mit (d^*,v^*) aus (2.10) ergibt sich für $\gamma = 0$

$$c'(0) = t(\sum_{i\in I_\beta} s^i\theta_i^*)^T(s^k - \sum_{i\in I_\beta} s^i\theta_i^*) + \alpha^k - \sum_{i\in I_\beta}\alpha^i\theta_i^* = d^{*T}s^k + \alpha^k - v^*.$$

Die Annahme $d^{*T}s^k + \alpha^k < v^*$ impliziert $c'(0) = \lim_{\gamma\to 0}\frac{c(\gamma)-c(0)}{\gamma} < 0$. Hiermit gilt für kleine $\gamma \in]0,1]$ $c(\gamma) < c(0)$. Da der Zielfunktionswert von θ^* gleich $c(0)$ ist, bedeutet dieses, dass die Aufgabe $(\Delta_{\beta t})$ bessere Lösungen als θ^* besitzt. Somit steht die Annahme $d^{*T}s^k + \alpha^k < v^*$ im Widerspruch zur Optimalität von θ^*.

Die Lagrange-Funktion der quadratischen Optimierungsaufgabe $(\Pi_{\beta t})$ lautet

$$F(d,v,\theta) = v - \frac{1}{2t}\|d\|^2 + \sum_{i\in I_\beta}\theta_i(d^Ts^i + \alpha^i - v).$$

Um die Optimalität von (d^*, v^*) für $(\Pi_{\beta t})$ zu beweisen, wird nun gezeigt, dass (d^*, v^*, θ^*) ein Sattelpunkt der Lagrange-Funktion F ist, das heißt dass für alle $(d, v) \in R^{m+1}$ und alle $\theta \geq 0$

$$F(d, v, \theta^*) \leq F(d^*, v^*, \theta^*) \leq F(d^*, v^*, \theta) \tag{2.13}$$

gilt. Für ein festes θ ist F eine stetig differenzierbare konkave Funktion, die ihr Maximum in einem Punkt mit $(\partial F/\partial d_1, \ldots, \partial F/\partial d_m, \partial F/\partial v)^T = 0$ annimmt. Also ist die erste Ungleichung der Sattelpunktbedingung (2.13) erfüllt, wenn

$$\begin{aligned} \frac{\partial F}{\partial d_j}(d^*, v^*, \theta^*) &= -\frac{1}{t} d_j^* + \textstyle\sum_{i \in I_\beta} \theta_i^* s_j^i = 0, \quad j = 1, \ldots, m \\ \frac{\partial F}{\partial v}(d^*, v^*, \theta^*) &= 1 - \textstyle\sum_{i \in I_\beta} \theta_i^* = 0 \end{aligned}$$

ist. Diese Bedingung ist offenbar wahr, denn d^* ist als $d^* = t \sum_{i \in I_\beta} \theta_i^* s^i$ definiert und θ^* stellt als eine zulässige Lösung von $(\Delta_{\beta t})$ ein Element aus $\Theta = \{\theta : \sum_{i \in I_\beta} \theta_i = 1, \theta \geq 0\}$ dar. Da außerdem

$$\textstyle\sum_{i \in I_\beta} \theta_i^* (d^{*T} s^i + \alpha^i - v^*) = t \| \sum_{i \in I_\beta} s^i \theta_i^* \|^2 + \sum_{i \in I_\beta} \alpha^i \theta_i^* - v^* = 0$$

gilt, kann die zweite Ungleichung der Sattelpunktbedingung (2.13) auch als

$$0 \leq \textstyle\sum_{i \in I_\beta} \theta_i (d^{*T} s^i + \alpha^i - v^*) \quad \text{für alle } \theta \geq 0$$

formuliert werden. Weil die Summe auf der rechten Seite dieser Ungleichung laut (2.12) keine negativen Summanden enthält, ist diese Bedingung ebenfalls erfüllt und (d^*, v^*, θ^*) ist tatsächlich ein Sattelpunkt der Lagrange-Funktion F. Nach dem Satz 2.4 bedeutet dies, dass (d^*, v^*) eine optimale Lösung der Optimierungsaufgabe $(\Pi_{\beta t})$ ist.

Die Wahl der Suchrichtung wird noch etwas komplizierter, wenn der zulässige Bereich wie in dem Lagrange-Multilikator-Problem auf R_+^m beschränkt ist. Um die Einschränkung $\omega \geq 0$ einzuhalten, werden der Aufgabe $(\Pi_{\beta t})$ zusätzliche Nebenbedingungen $d_j \geq \bar{\omega}_j,\ j = 1 \ldots, m$ hinzugefügt. Das zu $(\Pi_{\beta t})$ duale Problem $(\Delta_{\beta t})$ bekommt neue Variablen und wird dadurch etwas schwieriger zu lösen.

In jeder Iteration des Bundle-Algorithmus löst man, um den nächsten zu untersuchenden Punkt zu bestimmen, das quadratische Optimierungsproblem $(\Delta_{\beta t})$. Die Lösung dieses Problems ist häufig der aufwendigste Schritt des Bundle-Verfahrens, aufwendiger als die Lösung eines Lagrange-Teilproblems für die Berechnung von $z(\omega)$. In der Implementierung von Antonio Frangioni [Frangioni, 1997] wird zur Effizienzsteigerung ein spezialisiertes Lösungsverfahren angewandt, das die Struktur der Optimierungsaufgabe $(\Delta_{\beta t})$ ausnutzt. Zusätzlich wird ausgenutzt, dass sich die Aufgabe $(\Delta_{\beta t})$ zwischen zwei aufeinander folgenden Iterationen nur wenig ändert und die Lösung des aktuellen Problems $(\Delta_{\beta t})$ stark beschleunigt werden kann, wenn man auf die Information aus der letzten Iteration zurückgreift.

Verwaltung des Bündels

Eine andere Möglichkeit, um den Rechenaufwand für die Lösung des Problems $(\Delta_{\beta t})$ zu reduzieren, ist das Bündel β klein zu halten. In der Regel entfernt man einen Subgradienten

aus dem Bündel, wenn er in einer bestimmten Anzahl aufeinander folgender Iterationen *inaktiv* war, das heißt in der optimalen Lösung von $(\Delta_{\beta t})$ ein Gewicht $\theta_i = 0$ besaß.

Zusätzlich wird auch eine maximale Größe des Bündels festgelegt. Jedes Mal wenn das Bündel voll ist und ein neuer Subgradient hinzugefügt werden soll, werden einige der alten Einträge gelöscht. Damit das Verfahren trotz dieses Informationsverlustes konvergiert, fügt man ein Element $\langle \bar{\omega}, z(\bar{\omega}) + \alpha_{agr}, s_{agr} \rangle$ mit dem *aggregierten Subgradienten* $s_{agr} = \sum_{i \in I_\beta} s^i \theta_i$ und dem *aggregierten Linearisierungsfehler* $\alpha_{agr} = \sum_{i \in I_\beta} \alpha^i \theta_i$ dem Bündel β hinzu, immer wenn Subgradienten mit einem von Null verschiedenen Gewicht θ_i gelöscht werden. Diese aggregierten Subgradienten werden mit den „normalen“ Subgradienten von z gleichgesetzt. Man verwendet sie genauso zum Aufbauen eines Schnittebenenmodells der Funktion z und für die Wahl der Suchrichtung d.

Bundle-Algorithmus von Antonio Frangioni

Der Algorithmus 4 zeigt den allgemeinen Ablauf des in dieser Arbeit verwendeten Bundle-Verfahrens von Antonio Frangioni. Das Verfahren startet mit einer beliebigen Initiallösung $\bar{\omega}$ und einem Bündel β, das mit einem Subgradienten der Funktion z an der Stelle $\bar{\omega}$ initialisiert wird.

Algorithmus 4 Bundle-Verfahren

wähle $\omega^1 \in R^m$, $m_1 \in [0,1]$, $m_2 > 0$, $t_0 > 0$, $t^* > 0$ und $\varepsilon > 0$
bestimme $z(\omega^1)$ und einen Subgradienten $s^1 \in \partial z(\omega^1)$
$\bar{\omega} := \omega^1$, $t := t_0$, $\beta := \{\langle \omega^1, z(\omega^1), s^1 \rangle\}$, $i := 1$
repeat
  bestimme die optimalen Lösungen θ und (d, v) von $(\Delta_{\beta t})$ und $(\Pi_{\beta t})$
  $s_{agr} := \sum_{j \in I_\beta} s^j \theta_j$, $\alpha_{agr} := \sum_{j \in I_\beta} \alpha^j \theta_j$
  if $t^* \|s_{agr}\|^2 + \alpha_{agr} \leq \varepsilon z(\bar{\omega})$ **then**
    ABBRUCH
  else
    $i := i + 1$, $\omega^i := \bar{\omega} + d$
    bestimme $z(\omega^i)$ und einen Subgradienten $s^i \in \partial z(\omega^i)$
    if $z(\omega^i) - z(\bar{\omega}) \geq m_1 v$ **then** {ernster Schritt}
      $\bar{\omega} := \omega^i$
      if $d^T s^i > 0$ **then**
        erhöhe_t()
    else {Nullschritt}
      bestimme den Linearisierungsfehler α^i
      if $\alpha^i > m_2 \alpha_{agr}$ **then**
        verringere_t()
    lösche lange nicht benutzte Subgradienten aus β
    füge $\langle \omega^i, z(\omega^i), s^i \rangle$ in β ein
until ABBRUCH

In jeder Iteration löst man die quadratischen Optimierungsprobleme $(\Delta_{\beta t})$ und $(\Pi_{\beta t})$, um in der Umgebung der aktuellen Lösung $\bar{\omega}$ einen Probepunkt ω^i auszuwählen. Für diesen

Punkt berechnet man den Wert der Funktion z und einen Subgradienten $s^i \in \partial z(\omega^i)$. Jetzt muss entschieden werden, ob man den Punkt ω^i als die neue aktuelle Lösung $\bar{\omega}$ akzeptiert (*ernster Schritt*) oder den aktuellen Punkt $\bar{\omega}$ unverändert lässt (*Nullschritt*). Mit diesem Ziel wird der durch den Schritt nach ω^i erreichbare Anstieg des Zielfunktionswertes $z(\omega^i) - z(\bar{\omega})$ mit der von dem Schnittebenenmodell $\hat{z}_\beta$ vorausgesagten Verbesserung $v = \hat{z}_\beta(\omega^i) - z(\bar{\omega})$ verglichen. Wenn $\hat{z}_\beta$ das Verhalten der Funktion z gut genug modelliert und für einen Parameter $m_1 \in [0,1]$ $z(\omega^i) - z(\bar{\omega}) \geq m_1 v$ gilt, wird der aktuelle Punkt $\bar{\omega}$ nach ω^i verlegt. Andernfalls macht man einen Nullschritt. In diesem Fall sorgt das Hinzufügen der neuen Information $< \omega^i, z(\omega^i), s^i >$ zu dem Bündel β für eine Verbesserung des Schnittebenenmodells und für einen anderen (und hoffentlich besseren) Punkt ω^{i+1} in der nächsten Iteration. Für den Parameter m_1 wird in [Frangioni, 1997] der Wert 0.1 empfohlen. Crainic et al. bemerken in [Crainic et al., 2001b], dass $m_1 = 0$ in einigen Fällen etwas besser ist.

Das Verfahren hält an, wenn für den, durch eine Konvexkombination der Subgradienten aus dem Bündel β definierten, aggregierten Subgradienten s_{agr} und seinen Linearisierungsfehler α_{agr} gilt:

$$t^* \|s_{agr}\|^2 + \alpha_{agr} \leq \varepsilon z(\bar{\omega}). \tag{2.14}$$

Hier ist ε eine (relative) Fehlertoleranz und $t^* > 0$ ein problemabhängiger von dem Benutzer festgelegter Parameter. Die Bedingung (2.14) ist im Grunde nur eine andere Form des häufig verwendeten Abbruchkriteriums[7]

$$\|s_{agr}\| \leq \varepsilon_Q \quad \text{und} \quad \alpha_{agr} \leq \varepsilon_L, \tag{2.15}$$

bei dem die Suche beendet wird, sobald $\|s_{agr}\|$ und α_{arg} klein genug sind. Weil s_{agr} laut (2.9) einen α_{agr}-Subgradienten von z in $\bar{\omega}$ darstellt, ist (2.15) ein Zeichen dafür, dass

$$z(\omega) \leq z(\bar{\omega}) + \varepsilon_Q \|\omega - \bar{\omega}\| + \varepsilon_L \qquad \forall \omega$$

gilt. Das heißt: In der Umgebung von $\bar{\omega}$ mit dem Radius r gibt es keine Lösungen mit größeren Zielfunktionswerten als $z(\bar{\omega}) + \varepsilon_Q r + \varepsilon_L$. Das verwendete Kriterium (2.14) beschreibt die Situation, in der für jeden Punkt ω mit $\|\omega - \bar{\omega}\| \leq t^* \|s_{agr}\|$

$$z(\omega) \leq z(\bar{\omega}) + t^* \|s_{agr}\|^2 + \alpha_{agr} \leq (1+\varepsilon) z(\bar{\omega})$$

gilt. Wenn t viel kleiner als t^* ist, heißt dieses, dass der Abstand zwischen der aktuellen Lösung $\bar{\omega}$ und einer eventuell vorhandenen Lösung mit einem höheren Wert als $(1+\varepsilon)z(\bar{\omega})$ verglichen mit der Schrittweite $\|d\| = t\|s_{arg}\|$ sehr groß ist und ein möglicher Anstieg der Lösungsqualität in einem schlechten Verhältnis zu dem erwarteten Aufwand steht.

Das Konvergenzverhalten des Bundle-Algorithmus hängt ganz entscheidend von der Wahl des Parameters t ab. Er bestimmt, wie weit der jeweilige Probepunkt ω^i von der aktuellen Lösung $\bar{\omega}$ entfernt ist. Ein zu kleines t führt dazu, dass das Bundle-Verfahren kurze Schritte macht, die jeweils eine kleine Verbesserung mit sich bringen. Wird t dagegen zu groß gewählt, kann die Anzahl der Nullschritte zwischen zwei ernsten Schritten zu groß werden. Der „richtige“ Wert des Parameters t hängt davon ab, in welcher Umgebung von $\bar{\omega}$ das Schnittebenenmodell eine gute Approximation von z darstellt. Dieser Wert ändert sich

[7] Dieses Abbruchkriterium wird beispielsweise in [Hiriart-Urruty and Lemaréchal, 1993] und [Kiwiel, 1985] angewandt.

im Laufe des Verfahrens und lässt sich nur schwierig abschätzen. In dem Bundle-Verfahren von Antonio Frangioni werden zum Einstellen des Parameters t einige einfache Heuristiken eingesetzt.

So wird t vergrößert, wenn in einem ernsten Schritt $d^T s^i > 0$ ist, sodass man sich von einem größeren Schritt in die Richtung d einen weiteren Anstieg der Funktion z verspricht. Bei einem Nullschritt wird t verkleinert, wenn der neue Subgradient s^i einen vergleichsweise großen Linearisierungsfehler $\alpha^i = z(\omega^i) - d^T s^i - z(\bar{\omega})$ hat und so nur wenig zur Verbesserung des Schnittebenenmodells $\hat{z}_\beta$ in der Nähe der aktuellen Lösung $\bar{\omega}$ beiträgt. In diesem Fall sorgt das Verkleinern des Parameters t dafür, dass der nächste Punkt ω^{i+1} „näher“ zur aktuellen Lösung $\bar{\omega}$ gewählt wird, in einem Bereich, in dem das Schnittebenenmodell (vielleicht) etwas besser ist. Der bei der Entscheidung über das Verringern von t (Zeile 19) benutzte Parameter m_2 hat einen großen Einfluss auf das Verhalten des Algorithmus. Setzt man diesen Parameter wie in [Schramm and Zowe, 1992] gleich 0.9, ist die Bedingung $\alpha^i > m_2 \alpha_{agr}$ sehr häufig erfüllt, sodass der Parameter t schnell sehr klein wird und sich die Konvergenz des Algorithmus nach einer guten Anfangsphase stark verlangsamt. Ein höherer Wert zum Beispiel $m_2 = 3$ hemmt das Herabsetzen von t und führt dazu, dass sich dieser Parameter fast nie ändert. In diesem Fall kann eine gute Wahl des Initialwertes t_0 wichtig sein. [Crainic et al., 2001b]

Neben Heuristiken, die den Parameter t ausgehend von den Ergebnissen der letzten Iteration verändern, werden in dem Algorithmus von Antonio Frangioni auch so genannte *Langzeitstrategien* für die Wahl von t umgesetzt. Die *harte Langzeitstrategie* unterhält eine *minimale erwartete Verbesserung* $\bar{\varepsilon}$. Ein Punkt $\omega^i = \bar{\omega} + d$ wird bei dieser Strategie gar nicht untersucht, wenn die maximale durch den Schritt zu diesem Punkt erreichbare Verbesserung v kleiner als $\bar{\varepsilon} z(\bar{\omega})$ ist. Stattdessen erhöht man den Parameter t und löst die beiden Optimierungsaufgaben $(\Delta_{\beta t})$ und $(\Pi_{\beta t})$ nochmal, um einen Punkt mit einer größeren potentiellen Verbesserung v zu finden. Der Wert $\bar{\varepsilon}$ wird verringert, wenn die Abbruchbedingung (2.14) für die Fehlertoleranz $\bar{\varepsilon}$ erfüllt ist und man nicht mehr mit einer Verbesserung in dieser Größe rechnet.

Die andere *milde Langzeitstrategie* für t greift etwas weniger in den Ablauf des Algorithmus ein. Genau wie bei der harten Langzeitstrategie wird auch hier eine minimale erwartete Verbesserung $\bar{\varepsilon}$ gespeichert. Doch statt eine Erhöhung von t durchzusetzen, verbietet man bei dieser Strategie lediglich das Verkleinern von t, wenn der aktuelle Wert v zu klein ist, das heißt wenn $v \leq m_3 \bar{\varepsilon} z(\bar{\omega})$ für einen festen Parameter m_3 gilt. [Frangioni, 1997]

In seiner Dissertation [Frangioni, 1997] zeigt Antonio Fragioni, dass der oben beschriebene Bundle-Algorithmus für alle nach oben beschränkten Funktionen z und jeden Toleranzwert $\varepsilon > 0$ nach einer endlichen Anzahl der Iterationen terminiert, wenn der Parameter t in einem festen Interval $[\underline{t}, \overline{t}]$ mit $\underline{t} > 0$ bleibt und das Entfernen von Subgradienten mit $\theta_j > 0$ aus dem Bündel β immer durch das Hinzufügen des aggregierten Subgradienten $\langle \bar{\omega}, z(\bar{\omega}) + \alpha_{agr}, s_{agr} \rangle$ kompensiert wird.

2.7 Obere Schranken

Obere Schranken, also zulässige Lösungen, werden in unserem Algorithmus an drei Stellen benötigt:

1. im Subgradientenverfahren zur Berechnung der Schrittweite t,

2. im Branch-and-Bound-Algorithmus zum Abschneiden von Ästen im Branch-and-Bound Baum und

3. als beste gefundene Lösung, wenn das Verfahren endet.

Besonders für den zweiten Punkt ist es wichtig, schnell eine gute obere Schranke parat zu haben, um frühzeitig Bereiche des Lösungsraumes ausschließen zu können. Es reicht daher nicht, obere Schranken nur an den Blättern des Branch-and-Bound Baums zu berechnen, sondern man benötigt auch schnelle Heuristiken, um in jedem Knoten im Verlauf des Subgradientenverfahrens verbessernde zulässige Lösungen zu finden.

2.7.1 Das primale Unterproblem

Ein Knoten im Branch-and-Bound Baum bezeichnen wir im Folgenden mit dem Tupel $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$, wobei $\mathscr{A}_0$ die Menge der auf 0 fixierten, $\mathscr{A}_1$ die Menge der auf 1 fixierten und $\mathscr{A}_*$ die Menge der unfixierten Kanten darstellt. Um in einem Knoten des Branch-and-Bound Baums eine zulässige Lösung für das CNDP zu berechnen, betrachten wir das folgende *primale Unterproblem* PS($\bar{\mathscr{A}}$), wobei $\bar{\mathscr{A}} = \mathscr{A}_1 \cup \mathscr{A}_* \subseteq \mathscr{A}$ die Teilmenge der offenen (also der auf 1 oder nicht fixierten) Kanten von $\mathscr{A}$ ist.

Definition 2.4 (Primales Unterproblem PS) *Das* primale Unterproblem PS($\bar{\mathscr{A}}$) *lautet wie folgt:*

$$
\begin{aligned}
z_{PS}(\bar{\mathscr{A}}) \quad = \quad \min \quad & \sum_{(i,j)\in\bar{\mathscr{A}}} \sum_{k\in\mathscr{C}} c_{ij}^k x_{ij}^k + \sum_{(i,j)\in\mathscr{A}_1} f_{ij} + \sum_{(i,j)\in\mathscr{A}_* : \sum_k x_{ij}^k > 0} f_{ij} \\
\textit{u.d.N.} \quad & \sum_{j:(i,j)\in\bar{\mathscr{A}}} x_{ij}^k - \sum_{j:(j,i)\in\bar{\mathscr{A}}} x_{ji}^k = b_i^k \quad \forall i \in \mathscr{N},\ \forall k \in \mathscr{C} \\
& \sum_{k\in\mathscr{C}} x_{ij}^k \leq u_{ij} \quad \forall (i,j) \in \bar{\mathscr{A}} \\
& x_{ij}^k \geq 0 \quad \forall (i,j) \in \bar{\mathscr{A}}, \forall k \in \mathscr{C}
\end{aligned}
$$

Wie man leicht sieht, handelt es sich dabei um ein Mehrgüter-Fluss-Problem (MMCF) (s. Definition 2.3), wobei der Zielfunktionswert um die Summe der Fixkosten der benutzten unfixierten und auf 1 fixierten Kanten ergänzt wird. Als exaktes Lösungsverfahren für das MMCF benutzen wir die im nächsten Abschnitt 2.7.2 vorgestellten, auf Column Generation basierenden Algorithmen. Obwohl sich diese Verfahren besonders für die Reoptimierung eines nur leicht veränderten Modells als sehr effizient erwiesen haben, lohnt es sich doch nicht, das MMCF innerhalb des Subgradientenverfahrens mit ihrer Hilfe jedesmal exakt zu lösen. Im nächsten Abschnitt beschreiben wir daher eine Heuristik, die sehr schnell eine Abschätzung für PS erzeugt.

Die exakte Lösung von PS($\bar{\mathscr{A}}$) liefert neben einer oberen auch eine untere Schranke für einen Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$, die wie die Lagrange-Schranken für alle weiteren Teilprobleme des Knotens gültig ist. Sei

$$
z_{MMCF}(\bar{\mathscr{A}}) = z_{PS}(\bar{\mathscr{A}}) - \sum_{(i,j)\in\mathscr{A}_1} f_{ij} - \sum_{(i,j)\in\mathscr{A}_* : \sum_k x_{ij}^k > 0} f_{ij},
$$

also der reine Routinganteil der Zielfunktion von PS($\bar{\mathscr{A}}$). Dann gilt:

$$z_{MMCF}(\bar{\mathscr{A}}) + \sum_{(i,j)\in\mathscr{A}_1} f_{ij} \leq z_{CNDP}.$$

Die Gültigkeit der Schranke folgt aus der Tatsache, dass die Routingkosten durch weitere Fixierung von Variablen im Verlauf der Baumsuche höchstens schlechter werden können und auch der Anteil der auf 1 fixierten Variablen nur zunehmen kann. Einen ausführlichen Beweis findet man bei Holmberg und Yuan (s. [Holmberg and Yuan, 2000]).

2.7.2 Dantzig-Wolfe-Dekomposition

Das Dekompositionsverfahren nach Dantzig-Wolfe ist ein allgemeines Prinzip, um die Lösung von linearen Programmen zu beschleunigen, die eine spezielle Struktur besitzen. Bei unserer Darstellung orientieren wir uns an Kapitel 4.4 in [Papadimitriou and Steiglitz, 1982] und Kapitel 17 in [Ahuja et al., 1993].

Grundprinzip

Im Allgemeinen lässt sich die DW-Dekomposition auf ein lineares Programm P der folgenden Form anwenden:

$$\begin{array}{rcll} z_P & = & \min & c^T x \\ & & \text{u.d.N.} & Ax \geq b \quad (1) \\ & & & Nx \geq d \quad (2) \\ & & & x \geq 0 \quad (3) \end{array}$$

Dabei nehmen wir an, dass die Nebenbedingungen in Zeile (1) das Problem "schwierig" machen und es sich bei jenen in Zeile (2) um "einfache" Bedingungen handelt (z.B. ein Kürzeste-Wege Problem). Seien $x^1,\ldots,x^n$ die Extrempunkte der Menge der zulässigen Lösungen $S_N = \{x \mid Nx \geq d, x \geq 0\}$ des Unterproblems in Zeile (2). Aus der Theorie der linearen Programmierung wissen wir, dass wir dann jede zulässige Lösung x von P als Konvexkombination der Extrempunkte von S_N darstellen können:

$$x = \sum_{j=1}^{n} \lambda_j x^j, \quad \sum_{j=1}^{n} \lambda_j = 1, \quad \lambda_1,\ldots,\lambda_n \geq 0$$

Durch Einsetzen in unsere Ausgangsproblemstellung P eliminieren wir die Nebenbedingungen in Zeile (2) und erhalten das so genannte *Masterproblem*:

Definition 2.5 (Masterproblem der DW-Dekomposition) *Seien die $\{x^1,\ldots,x^n\}$ und P definiert wie oben. Dann heißt*

$$\begin{array}{rcll} z_{MP} & = & \min & \sum_{j=1}^{n} (cx^j)\lambda_j \\ & & \textit{u.d.N.} & \sum_{j=1}^{n} (Ax^j)\lambda_j \geq b \quad (4) \quad \pi \\ & & & \sum_{j=1}^{n} \lambda_j = 1 \quad (5) \quad \pi_0 \\ & & & \lambda \geq 0 \quad (6) \end{array}$$

das Masterproblem MP *der Dantzig-Wolfe-Dekomposition von P.*

Für jeden Extrempunkt von S_N ergibt sich also eine Spalte in MP. Für die meisten Problemstellungen wird es jedoch nicht möglich sein, die exponentiell vielen Extrempunkte von S_N explizit aufzuzählen und MP direkt zu lösen. Das ist aber auch gar nicht nötig, denn zu diesem Zweck kann ein sog. Column-Generation-Verfahren eingesetzt werden. Dabei werden nur solche Spalten bei der Lösung von MP berücksichtigt, die im Verlauf des Simplex-Verfahrens die besten reduzierten Kosten besitzen und als Pivotspalten in Frage kommen. Sei π der Vektor der dualen Variablen zu den Nebenbedingungen in Zeile 4 und π_0 die duale Variable zur Nebenbedingung (5). Für die reduzierten Fixkosten $\bar{c}_j$ einer Spalte j gilt dann:

$$\bar{c}_j = c^T x^j - \pi^T A x^j - \pi_0$$

In einer Simplexiteration lässt sich die Spalte mit den geringsten reduzierten Kosten nun mit folgendem Unterproblem SP bestimmen, dass wegen der Struktur von N besonders einfach zu lösen ist:

$$\begin{array}{rcll} z_{SP} & = & \min & (c^T - \pi^T A)x - \pi_0 \\ & & \text{u.d.N.} & Nx \geq d \\ & & & x \geq 0 \end{array}$$

Damit haben wir nun einen Lösungsalgorithmus für P (s. Algorithmus 5): Im ersten Schritt stellen wir das Masterproblem MP^0 von P für eine kleine Anzahl explizit generierter Extrempunkte von S_N und künstlichen Variablen für jede Nebenbedingung auf, die die Lösbarkeit garantieren. In jeder weiteren Iteration i lösen wir dann MP^i mit dem Simplexverfahren, erhalten duale Variablen π, π_0 und lösen das Unterproblem SP^i. Falls $z_{SP^i} \geq 0$ ist, gibt es keine lohnende Spalte mehr und wir sind fertig. Im anderen Fall erzeugen wir aus der Lösung x^* von SP eine Spalte für das Masterproblem und starten die nächste Iteration.

Algorithmus 5 Dantzig-Wolfe-Dekomposition

1: $l = 0$
2: *erzeuge* MP^0
3: **repeat**
4: *löse* MP^l, *erhalte Dualvariablen* π^l
5: *löse* SP^l
6: **if** $(z_{SP^l} < 0)$ **then**
7: $MP^{l+1} = MP^l + \begin{bmatrix} Ax^* \\ 1 \end{bmatrix}$
8: $l = l + 1$
9: **until** $(z_{SP^{l-1}} \geq 0)$
10: **return** $z_{MP^{l-1}}$

Column-Generation für das MMCF

Das kantenbasierte Modell des Mehrgüter-Fluss Problems aus Definition 2.3 besitzt genau die Struktur, die für die Dantzig-Wolfe-Dekomposition benötigt wird. Die Kapazitätsbedingungen für jede Kante bilden die Menge der schwierigen Nebenbedingungen. Die Flussbedingungen

für jeden Knoten und jedes Gut sind die einfachen Nebenbedingungen und es ist $S_N = \{x | Nx^k = b^k \; \forall k \in \mathscr{C}, x \geq 0\}$. Bei der Matrix N handelt es sich hier um die Adjazenzmatrix des Transportnetzwerks $G = (\mathscr{A}, \mathscr{N})$. Die Extrempunkte von S_N repräsentieren also Transportwege zwischen den Quellen und Senken der Güter in $\mathscr{C}$ im Graphen G. Die Menge der Transportwege für jedes Gut nennen wir P^k und führen für jeden Pfad $p \in P^k$ eine Variable $0 \leq h_p^k \leq 1$ ein, die die anteilige Transportmenge des Gutes $k \in \mathscr{C}$ bezeichnet, die auf p transportiert wird. Sei $\delta_{ij}^p = 1$, falls der Pfad p die Kante $(i,j) \in \mathscr{A}$ enthält, und sonst $\delta_{ij}^p = 0$. Die Transportkosten für den gesamten Transportbedarf d^k eines Gutes $k \in \mathscr{C}$ auf einem Pfad p ergeben sich damit als $c_p^k = \sum_{(i,j)\in\mathscr{A}} \delta_{ij}^p c_{ij}^k d^k$. Mit diesen Bezeichnungen lautet das Masterproblem MP wie folgt:

$$
\begin{array}{rcll}
z_{MP(P)} = \min & \displaystyle\sum_{k\in\mathscr{C}} \sum_{p\in P^k} c_p^k h_p^k & & \\
\text{u.d.N.} & \displaystyle\sum_{k\in\mathscr{C}} \sum_{p\in P_k} \delta_{ij}^p d^k h_p^k \leq u_{ij} & \forall (i,j) \in \mathscr{A} & \pi_{ij} \\
& \displaystyle\sum_{p\in P_k} h_p^k = 1 & \forall k \in \mathscr{C} & \pi_k \\
& h \geq 0 & &
\end{array}
$$

Damit haben wir auch für das MMCF eine pfadbasierte Modellvariante definiert. Die reduzierten Kosten $\bar{c}_p^k$ eines Pfades $p \in P^k$ ergeben sich nun folgendermaßen:

$$\bar{c}_p^k = c_p^k - \sum_{(i,j)\in\mathscr{A}} \delta_{ij}^p d^k \pi_{ij} - \pi_k$$

Das Unterproblem SP zerfällt für das MMCF in $|\mathscr{C}|$ Unterprobleme SP^k, eins für jedes Gut k. Zur Verkürzung der Schreibweise definieren wir Kosten $\tilde{c}_{ij}^k = c_{ij}^k - \pi_{ij}$ für jede Kante (i,j). Es gilt $\tilde{c}_{ij}^k \geq 0$, denn $\pi_{ij} \leq 0$. Damit lautet das Unterproblem SP^k:

$$
\begin{array}{rcll}
z_{SP^k} = \min & \displaystyle\sum_{(i,j)\in\mathscr{A}} \tilde{c}_{ij}^k x_{ij}^k & & \\
\text{u.d.N.} & Nx^k = b^k & & \\
& x^k \geq 0 & &
\end{array}
$$

Es handelt sich offensichtlich um ein Kürzeste-Wege Problem, dass sich wegen der positiven Kosten leicht mit dem Dijkstra Algorithmus lösen lässt. Nach diesen Vorarbeiten können wir nun unser Lösungsverfahren in Algorithmus 6 zusammenfassen. $MP(P)$ ist das Masterproblem, das neben künstlichen Variablen nur die Spalten enthält, deren Pfade in P sind.

2.7.3 Heuristik für das primale Unterproblem

Die Heuristik funktioniert nach dem Greedy-Prinzip. Wir wählen ein Gut $k \in \mathscr{C}$ und betrachten das MMCF nur für dieses Gut. Es ergibt sich ein einfaches Min-Cost-Flow Problem (MCF) der folgenden Form:

Algorithmus 6 Column-Generation für das MMCF

1: *setze* $P = \emptyset$
2: *erzeuge* $MP(P)$
3: **repeat**
4: *setze* $count = 0$
5: *löse* $MP(P)$ *mit dem primalen Simplexalgorithmus*
6: **for all** $(k \in \mathscr{C})$ **do**
7: *löse* SP^k *mit Dijkstra's Algorithmus, erhalte kürzesten Weg* p_k^*
8: **if** $(z_{SP^k} - \pi_k < 0)$ **then**
9: *setze* $P = P \cup p_k^*$
10: *setze* $count = count + 1$
11: **until** $(count = 0)$
12: **return** $z_{MP(P)}$

$$
\begin{aligned}
z^k_{MCF}(\bar{\mathscr{A}}) \quad = \quad & \min \quad \sum_{(i,j)\in\bar{\mathscr{A}}} c^k_{ij} x^k_{ij} \\
& \text{u.d.N.} \quad \sum_{j:(i,j)\in\bar{\mathscr{A}}} x^k_{ij} - \sum_{j:(j,i)\in\bar{\mathscr{A}}} x^k_{ji} \quad = \quad b^k_i \quad \forall i \in \mathscr{N} \\
& \qquad\qquad x^k_{ij} \quad \leq \quad u_{ij} \quad \forall (i,j) \in \bar{\mathscr{A}} \\
& \qquad\qquad x^k_{ij} \quad \geq \quad 0 \quad \forall (i,j) \in \bar{\mathscr{A}}
\end{aligned}
$$

Dieses Problem lösen wir mit dem sehr effizienten Network-Simplex von CPLEX ([ILOG, 2003]) und erhalten entweder einen optimalen Flussvektor x^k oder die Rückmeldung, dass keine zulässige Lösung existiert. In diesem Fall brechen wir ab. Im ersten Fall reduzieren wir die Kapazitäten des Transportnetzwerks für jede Kante (i, j) um x^k_{ij}, wählen das nächste Gut und fahren fort, bis entweder alle Güter berücksichtigt wurden oder Unlösbarkeit festgestellt wurde. Die Summe der z^k_{MCF} zuzüglich der Fixkosten der benutzten unfixierten und auf 1 fixierten Kanten ergibt die obere Schranke für das CNDP. Algorithmus 7 zeigt das Verfahren im Überblick. Die gelieferte Lösung hängt natürlich von der Reihenfolge ab, in der die Güter berücksichtigt werden. Für die Rucksack-Relaxation hat sich gezeigt, dass es günstig ist, die Reihenfolge mittels der aktuellen Lagrangemultiplikatoren festzulegen. Zu diesem Zweck ordnen wir jedem Gut die Differenz zwischen den Multiplikatoren für den Start- und den Zielknoten des Gutes, multipliziert mit dem Transportbedarf zu, d.h. $(\omega^k_{D(k)} - \omega^k_{O(k)})d^k$, und wählen die Güter dann in absteigender Reihenfolge.

Unlösbarkeit durch die Heuristik bedeutet natürlich nicht, dass das MMCF tatsächlich keine zulässige Lösung besitzt. Besonders bei eng kapazitierten Problemen kann leicht der Fall eintreten, dass das MCF für ein Gut nicht mehr lösbar ist, weil wichtige Kanten schon durch andere Güter geblockt sind. In diesem Fall muss das MMCF mit einem exakten Verfahren gelöst werden, das die Unlösbarkeit zweifelsfrei nachweist oder eine Lösung findet.

2.8 Branching-Strategien

Für jedes Branch-and-Bound Verfahren ist von zentraler Bedeutung, wie die Suche innerhalb des Branch-and-Bound Baums organisiert wird, nach welchen Kriterien also das nächste zu behandelnde Teilproblem ausgewählt wird, und wie beim Branching neue Teilprobleme

Algorithmus 7 Heuristik für das primale Unterproblem

initialisiere $K = \mathscr{C}$
repeat
  wähle $k \in K$ *und setze* $K = K \setminus \{k\}$
  berechne $z^k_{MCF}(\bar{\mathscr{A}})$ *und* x^k
  if (MCF hat keine zulässige Lösung) **then**
    break
  else
    for all $(i,j) \in \bar{\mathscr{A}}$ **do**
      setze $u_{ij} = u_{ij} - x^k_{ij}$
until K $= \emptyset$
if K $= \emptyset$ **then**
  return $\sum_k z^k_{MCF}(\bar{\mathscr{A}}) + \sum_{(i,j)\in\mathscr{A}_1} f_{ij} + \sum_{(i,j)\in\mathscr{A}_*:\sum_k x^k_{ij}>0} f_{ij}$
else
  return ∞

generiert werden. Als Baumsuchstrategie haben wir in unserem Verfahren unter anderem Tiefensuche implementiert. Damit wird zum einen das schnelle Finden von guten oberen Schranken begünstigt und zum anderen eine gute Integration des Subgradientenverfahrens durch die Wiederverwendung der Lagrange-Multiplikatoren des Vaterknotens ermöglicht. Auf die implementierten Branchingstrategien wollen in den nächsten beiden Abschnitten genauer eingehen.

2.8.1 Branching mit Variablendichotomie

Variablendichotomie ist die Standardvorgehensweise für das Branching bei gemischt-ganzzahligen Programmen. Sei $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ der aktuelle Knoten (das aktuelle Teilproblem), auf dem gebrancht werden soll. Wir wählen nun eine Designvariable y_{ij} aus mit $(i,j) \in \mathscr{A}_*$ und generieren zwei neue Knoten, indem wir in dem einen Teilproblem $y_{ij} = 1$ setzten und damit die Kante als geöffnet fixieren, und im zweiten Teilproblem $y_{ij} = 0$ setzten und damit die Kante schließen. Die neuen Knoten $(\mathscr{A}_0, \mathscr{A}_1 \cup \{(i,j)\}, \mathscr{A}_* \setminus \{(i,j)\})$ und $(\mathscr{A}_0 \cup \{(i,j)\}, \mathscr{A}_1, \mathscr{A}_* \setminus \{(i,j)\})$ fügen wir zur Menge $\mathscr{H}$ der noch nicht untersuchten Branch-and-Bound Knoten hinzu.

Es ergibt keinen Sinn auf einer Kante zu branchen, die in der primalen Lösung des aktuellen Knotens nicht benutzt wird (beschrieben durch die Menge $\mathscr{A}^0 = \{(i,j) \in \mathscr{A}_* : y_{ij} = 0\}$. Bei einer solchen Wahl wäre in beiden neuen Teilproblemen keine bessere primale Lösung zu erwarten. Um unter den verbleibenden Designvariablen eine Branchingvariable y_{ij} auszuwählen, benutzen wir wie in [Holmberg and Yuan, 2000] die reduzierten Fixkosten aus der Berechnung der unteren Schranke (hier am Beispiel der Rucksack-Relaxation). Für das Branching mit Variablendichotomie haben wir die beiden folgenden Alternativen implementiert:

1. $(i,j) = argmin_{(i,j)\in\mathscr{A}_*\setminus\mathscr{A}^0} |\tilde{g}_{ij}|$
2. $(i,j) = argmax_{(i,j)\in\mathscr{A}_*\setminus\mathscr{A}^0} |\tilde{g}_{ij}|$

Alternative 1 wählt diejenige Variable mit den betragsmäßig kleinsten reduzierten Fixkosten aus. Bei der Lösung der Subprobleme der Lagrange-Relaxation werden die y-Variablen auf

0 bzw. 1 gesetzt, deren reduzierte Fixkosten positiv bzw. negativ sind. Man kann diese Entscheidung als umso sicherer ansehen, je betragsmäßig größer der entsprechende reduzierte Fixkostenwert ist. In diesem Fall wird also diejenige Variable zum Branching gewählt, deren Belegung in der Lösung des Lagrange-Duals am unsichersten ist.

Alternative 2 folgt der Überlegung, solche Variablen frühzeitig zum Branching auszuwählen, deren Belegung im Lagrange-Dual sehr sicher ist und somit wahrscheinlich auch in guten primalen Lösungen ähnlich vermutet wird. Auf diese Weise sollen schneller gute obere Schranken gefunden werden.

Schließlich bleibt noch zu entscheiden, welcher der neu erzeugten Knoten im Rahmen der Tiefensuche zuerst behandelt werden soll. Hier kann man sich wiederum eine Reihe von Alternativen vorstellen. Man könnte beispielsweise immer zuerst den "1-Ast" bzw. den "0-Ast" verfolgen. Vielversprechender ist es jedoch auch hier, die Wahl von der Lösung des Lagrange-Duals abhängig zu machen. Wenn y_{ij} dort auf 1 gesetzt wurde, wird also als Nächstes der Knoten $(\mathscr{A}_0, \mathscr{A}_1 \cup \{(i,j)\}, \mathscr{A}_* \setminus \{(i,j)\})$ behandelt, sonst der Knoten $(\mathscr{A}_0 \cup \{(i,j)\}, \mathscr{A}_1, \mathscr{A}_* \setminus \{(i,j)\})$. Unsere Erfahrungen in diesem Punkt stimmen mit denen von Holmberg und Yuan überein.

2.8.2 Branching mit Kardinalitätsbedingungen

Die Grundidee beim Branching mit Kardinalitätsbedingungen besteht darin, durch verbesserte Schranken und erweiterte Variablenfixierung schnell die Bereiche des Lösungsraumes auszuschließen, in denen keine guten Lösungen enthalten sein können. Die Zerlegung in Teilprobleme wird hier nicht durch die Fixierung einer Branchingvariablen erreicht, sondern durch Hinzufügen von zwei weiteren Nebenbedingungen der folgenden Form zu jedem Teilproblem:

$$\sum_{(i,j)\in\mathscr{A}} y_{ij} \geq a \qquad \textit{und} \qquad \sum_{(i,j)\in\mathscr{A}} y_{ij} \leq b.$$

Die Parameter $a, b \in \{0 \ldots |\mathscr{A}|\}$ geben jeweils die minimale bzw. maximale Anzahl von Kanten an, die in einer zulässigen Lösung eines Teilproblems mindestens installiert werden müssen. Sei beispielsweise $S^{[a,b]} = (\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*, a, b)$ ein Knoten im Branch-and-Bound Baum, auf dem gebrancht werden soll. Mit $s \in [a,b]$ erzeugen wir die neuen Teilprobleme $S^{[a,s]} = (\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*, a, s)$ und $S^{[s+1,b]} = (\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*, s+1, b)$. Offensichtlich ist dies eine Partitionierung von $S^{[a,b]}$. Mit einem weiteren Parameter $c \in \mathbb{N}$ legen wir die minimal zulässige Intervallgröße fest. Gilt für ein Teilproblem $s - a \leq c$ oder $b - s - 1 \leq c$, so schalten wir auf Branching durch Variablendichotomie um. Für ein Beispielproblem mit 100 Kanten und $s = \lfloor (a+b)/2 \rfloor$ ergibt sich bei dieser Vorgehensweise wieder ein Binärbaum, wie er in Abbildung 2.9 dargestellt ist.

Lösung der Lagrange-Relaxation mit Kardinalitätsbedingungen

Die zusätzlichen Kardinalitätsbedingungen können bei der Berechnung der Lagrange-Relaxationen $z_{SP}(\alpha,\beta)$ und $z_{KS}(\omega)$ einfach berücksichtigt werden, denn sie haben keine Auswirkungen auf die Lösung der Kürzeste-Wege- und Rucksack-Unterprobleme. In beiden Relaxationen wird die Belegung der y-Variablen allein aufgrund der reduzierten Fixkosten $\tilde{f}_{ij}$ und $\tilde{g}_{ij}$ entschieden (s. Abschnitte 2.4.2 und 2.4.2).

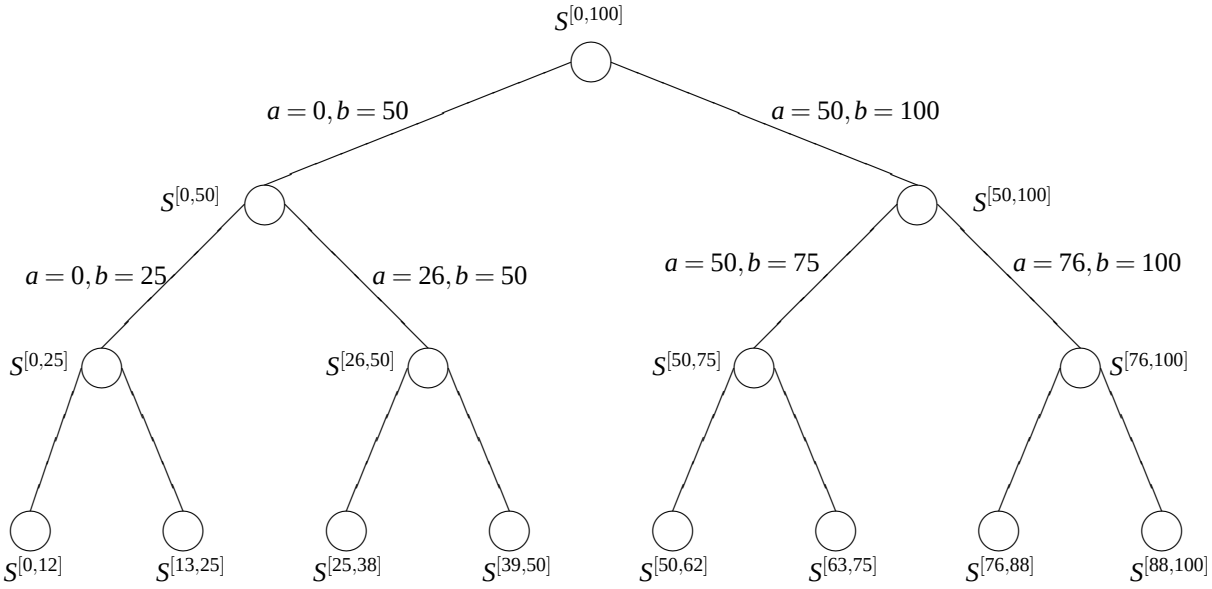


Abbildung 2.9: Enumerationsbaum beim Branching mit Kardinalitätsbedingungen

Für die y-Entscheidungsprobleme bedeutet ein Kardinalitätsintervall $[a,b]$, dass mindestens a und höchstens b y-Variablen auf 1 gesetzt werden müssen. Für die Kanten $(i,j) \in \mathscr{A}_1$ ist natürlich $y_{ij} = 1$. Um die Zielfunktion der Relaxation zu minimieren, sortieren wir nun die reduzierten Fixkosten in aufsteigender Reihenfolge und setzen die ersten $a - |\mathscr{A}_1|$ y-Variablen auf 1. Anschließend setzen wir $y_{ij} = 1$, solange die entsprechenden reduzierten Fixkosten negativ sind. Sobald diese für eine Kante positiv sind oder b y-Variablen auf 1 gesetzt wurden, brechen wir ab. Dieses Vorgehen ist in Algorithmus 8 für die Rucksack-Relaxation zusammengefasst. Hier wird nun auch klar, warum sich die untere Schranke vergrößert. Sei

Algorithmus 8 y-Entscheidungsproblem mit Kardinalitätsbedingungen

```
1:  setze y = 0
2:  for all ((i,j) ∈ A_1) do
3:     setze y_ij = 1
4:  setze s = |A_1|
5:  repeat
6:     (i,j) = argmin_{(i,j)∈A_*} g̃_ij
7:     if (s < a)  ODER  (g_ij < 0) then
8:        setze y_ij = 1
9:        setze g_ij = ∞
10:       setze s = s + 1
11: until ((s ≥ a)  UND  (g_ij ≥ 0))  ODER  (s = b)
```

$S = \{(i,j) | \tilde{g}_{ij} < 0\}$ die Menge der Kanten, für die es sich also lohnen würde, sie auf 1 zu setzen. Ohne Kardinalitätsbedingungen lässt sich der Anteil der reduzierten Fixkosten an der Lagrangezielfunktion $z_{KS}(\omega)$ bezeichnen mit $\tilde{z}_{KS}(\omega) := \sum_{(i,j)\in S} \tilde{g}_{ij}$. Der konstante Anteil

von $z_{KS}(\omega)$ ändert sich durch Kardinalitätsbedingungen nicht. Wir können nun drei Fälle unterscheiden:

1. Im Fall $|S| < a$ werden durch Algorithmus 8 auch solche y_{ij} auf 1 gesetzt für die $\tilde{g}_{ij} \geq 0$ ist.

2. Im Fall $a \leq |S| \leq b$ wird genau $y_{ij} = 1$ für $(i,j) \in S$ gesetzt.

3. Im Fall $b < |S|$ gibt es Kanten $(i,j) \in S$, für die $y_{ij} = 0$ gesetzt wird.

In allen drei Fällen folgt $\tilde{z}_{KS}(\omega) \leq \sum_{(i,j)\in\mathscr{A}} \tilde{g}_{ij} y_{ij}$, die untere Schranke wird also höchstens größer.

Zusätzliche Einschränkung des Kardinalitätsintervalls

Wenn eine gute obere Schranke $\hat{z}_{CNDP}$ vorliegt, können wir das Kardinalitätsintervall in einem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*, a, b)$ während der Lösung des y-Entscheidungsproblems unabhängig vom Branching weiter einschränken. Die Idee basiert auf der Tatsache, dass die Lagrange-Relaxationen für jede Belegung der y-Variablen eine untere Schranke liefern, also auch beispielsweise in jeder Iteration der Repeat-Schleife in Algorithmus 8. Wenn wir eine Sortierung $e_1, \ldots, e_{|\mathscr{A}|}$ der Kanten annehmen, sodass für $k < l$ und $e_k, e_l \in \mathscr{A}$ gilt $\tilde{g}_{e_k} \leq \tilde{g}_{e_l}$, dann lautet diese Schranke für die Rucksack-Relaxation in einer Iteration v:

$$z^v_{KS}(\omega) = \sum_{u \leq v} \tilde{g}_{e_u} + \omega^T d$$

Sie gilt für das Kardinalitätsintervall $[0, v]$. Solange diese untere Schranke größer als die beste bekannte obere Schranke ist, heißt das, dass in diesem Intervall keine bessere obere Schranke gefunden werden kann. Falls $a < v$ kann das Kardinalitätsintervall des Knoten also eingeschränkt werden auf $[v+1, b]$. Mit der gleichen Argumentation folgt, dass auch die andere Intervallgrenze verkleinert werden kann, wenn für $v > s$ gilt $z^v_{KS}(\omega) > \hat{z}_{CNDP}$, nämlich auf $[a, v-1]$. Diese Überlegungen fassen wir in Satz 2.1 für die Rucksack-Relaxation zusammen.

Satz 2.1 *Es sei mit $e_1, \ldots, e_{|\mathscr{A}|}$ die obige Ordnung auf den Kanten gegeben und außerdem $z_{KS}(\omega) < \hat{z}_{CNDP}$. Dann gilt:*

a) Es existieren Grenzen

$$\begin{aligned} \hat{a} &= \mathit{argmin}_{0\leq v\leq|\mathscr{A}|}\{z^v_{KS}(\omega) < \hat{z}_{CNDP}\} \quad \mathit{und} \\ \hat{b} &= \mathit{argmax}_{0\leq v\leq|\mathscr{A}|}\{z^v_{KS}(\omega) < \hat{z}_{CNDP}\}, \end{aligned}$$

b) und für jede verbesserte obere Schranke gilt

$$\hat{a} \leq \sum_{(i,j)\in\mathscr{A}} y_{ij} \leq \hat{b}.$$

Beweis. Für a) reicht es zu zeigen, dass ein $0 \leq l \leq |\mathscr{A}|$ existiert, sodass $z_{KS}^l(\omega) < \hat{z}_{CNDP}$. Sei l so gewählt, dass $z_{KS}^l(\omega) = z_{KS}(\omega)$. Dann gilt $z_{KS}^l(\omega) \leq \hat{z}_{CNDP}$ nach Voraussetzung.

b) Sei $L^{>}[\hat{b}] := L[\sum_{(i,j)\in\mathscr{A}} y_{ij} > \hat{b}]$ das Optimierungsproblem, das aus L durch Hinzufügen der Nebenbedingung $\sum_{(i,j)\in\mathscr{A}} y_{ij}$ entsteht und $z^{>}[\hat{b}]$ sein Zielfunktionswert. Aus der Definition von $\hat{b}$ folgt, dass $z_{KS}^{>}[\hat{b}] > \hat{z}_{CNDP}$. Angenommen, es existierte eine zulässige Lösung (x,y) für CNDP mit $\hat{b} < \sum_{(i,j)\in\mathscr{A}} y_{ij} \leq |\mathscr{A}|$. Dann gilt:

$$c^T x + f^T y \geq z_{CNDP}^{>}[\hat{b}] \geq z_{KS}^{>}[\hat{b}] > \hat{z}_{CNDP}.$$

Der Fall für $\sum_{(i,j)\in\mathscr{A}} y_{ij} < \hat{a}$ folgt analog. □

2.9 Variablenfixierung

Die Idee von Variablenfixierung (*variable fixing*) im Rahmen von Branch-and-Bound Verfahren besteht darin, bei der Schrankenberechnung anfallende Informationen zu nutzen, um den Lösungsraum des aktuellen Teilproblems zu verkleinern. Dabei wird versucht, durch ein Ausschlussverfahren für möglichst viele Lösungsvariablen die Werte zu ermitteln, die sie in einer optimalen Lösung des Teilproblems annehmen müssen. Diese Variablen werden dann auf die gefundenen Werte *fixiert*, d.h. sie kommen bei der weiteren Lösung des Teilproblems nicht mehr als Branchingvariable in Frage, was in der Regel zu einer enormen Verkleinerung des Branch-and-Bound-Baums führt. In der Literatur findet man für dieses Vorgehen auch die Begriffe *penalty tests* und *problem reduction*.

Sowohl für die Kürzeste-Wege-Relaxation als auch für die Rucksack-Relaxation sind einfache Strategien zur Variablenfixierung bekannt. Neu ist die Kombination dieser Strategien, unabhängig von der gerade verwendeten Relaxation. Die Grundidee ist dabei, im Verlauf des Subgradientenverfahrens immer auch die Lagrange-Multiplikatoren der jeweils anderen Relaxation zu finden und mit ihrer Hilfe deren Regeln zur Variablenfixierung zusätzlich anzuwenden.

2.9.1 Grundidee

Variablenfixierung in der SP-Relaxation

In jeder Iteration des Subgradientenverfahrens werden in der Kürzeste-Wege-Relaxation für gegebene Lagrange-Multiplikatoren $(\bar{\alpha},\bar{\beta})$ zunächst die $|\mathscr{C}|$ Kürzeste-Wege-Probleme und das y-Entscheidungsproblem gelöst, um dann mit der erhaltenen Lösung $(\bar{x},\bar{y})$ die Lagrange-Zielfunktion $z_{SP}(\bar{\alpha},\beta)$ auszuwerten:

$$z_{SP}(\bar{\alpha},\bar{\beta}) = \sum_{k\in\mathscr{C}} \sum_{(i,j)\in\mathscr{A}} \tilde{c}_{ij}^k \bar{x}_{ij}^k + \sum_{(i,j)\in\mathscr{A}} \tilde{f}_{ij}\bar{y}_{ij}.$$

Wir wissen dass $z_{SP}(\bar{\alpha},\bar{\beta})$ eine untere Schranke für das aktuelle Teilproblem darstellt. Außerdem können wir nun leicht sehen, um welchen Wert sich die Schranke mindestens vergrößern würde, wenn eine Variable y_{ij} entgegen der Lösung des y-Entscheidungsproblems fixiert wäre, nämlich genau um den Betrag $|\tilde{f}_{ij}|$. Bei festen $(\bar{\alpha},\bar{\beta})$ gilt also:

1. Falls $\tilde{f}_{ij} \geq 0$ (im y-Entscheidungsproblem wurde $\bar{y}_{ij} = 0$ gesetzt):

a) $z_{SPy_{ij}=1} \geq z_{SP}(\bar{\alpha},\bar{\beta})_{y_{ij}=1} = z_{SP}(\bar{\alpha},\bar{\beta}) + \tilde{f}_{ij}$

b) $z_{SPy_{ij}=0} \geq z_{SP}(\bar{\alpha},\bar{\beta})_{y_{ij}=0} \geq z_{SP}(\bar{\alpha},\bar{\beta})$

2. Falls $\tilde{f}_{ij} < 0$ (im y-Entscheidungsproblem wurde $\bar{y}_{ij} = 1$ gesetzt):

a) $z_{SPy_{ij}=1} \geq z_{SP}(\bar{\alpha},\bar{\beta})_{y_{ij}=1} = z_{SP}(\bar{\alpha},\bar{\beta})$

b) $z_{SPy_{ij}=0} \geq z_{SP}(\bar{\alpha},\bar{\beta})_{y_{ij}=0} \geq z_{SP}(\bar{\alpha},\bar{\beta}) + |\tilde{f}_{ij}|$

Zu den Fällen, in denen y_{ij} auf 0 fixiert wird (1.(b) und 2.(b)) ist anzumerken, dass hier der Wert von $z_{SP}(\bar{\alpha},\bar{\beta})_{y_{ij}=0}$ nur abgeschätzt werden kann, denn das Schließen einer Kante kann eventuell zu höheren Kosten in den Kürzeste-Wege-Problemen führen. Wir erhalten also Abschätzungen für die untere Schranke der Teilprobleme, die durch ein Branching auf y_{ij} entstehen, ohne diese Verzweigung und die exakte Berechnung der Schranken tatsächlich durchführen zu müssen. Die Fälle 1.(a) und 2.(b) können wir nun nutzen, um mit Hilfe der aktuell besten oberen Schranke $\hat{z}_{CNDP}$ eine mögliche Fixierung von y_{ij} zu prüfen:

- Fixiere y_{ij} auf 0, falls $\tilde{f}_{ij} \geq 0$ und $z_{SP}(\bar{\alpha},\bar{\beta}) + \tilde{f}_{ij} \geq \hat{z}_{CNDP}$
- Fixiere y_{ij} auf 1, falls $\tilde{f}_{ij} < 0$ und $z_{SP}(\bar{\alpha},\bar{\beta}) + |\tilde{f}_{ij}| \geq \hat{z}_{CNDP}$

Prinzipiell kann dieser Test in jeder Iteration des Subgradientenverfahrens oder des Bundle-Verfahrens für alle unfixierten Variablen y_{ij} durchgeführt werden, denn die unteren Schranken sind für alle zulässigen Lagrange-Multiplikatoren gültig. Effizienzüberlegungen legen jedoch nahe, den Test nur in jeder verbessernden Iteration oder etwa nur am Ende mit den optimalen Lagrange-Multiplikatoren vorzunehmen. Es stellt sich zudem die Frage, ob es besser ist, die Variablen schon im Verlauf des Subgradientenverfahrens oder erst bei der Generierung der neuen Teilprobleme zu fixieren.

Variablenfixierung in der KS-Relaxation

Variablenfixierung in der Rucksack-Relaxation funktioniert analog zur Kürzeste-Wege-Relaxation. Die Zielfunktion der Lagrange-Relaxation lautet für festes $\bar{\omega}$ und eine Lösung $(\bar{x},\bar{y})$ der Unterprobleme wie folgt:

$$z_{KS}(\bar{\omega}) = \min_{y \in \{0,1\}^{|\mathscr{A}|}} \sum_{(i,j)\in\mathscr{A}} \tilde{g}_{ij}\bar{y}_{ij} - \sum_{k\in\mathscr{C}} d^k(\bar{\omega}^k_{O(k)} - \bar{\omega}^k_{D(k)})$$

Wieder ergeben sich die Schätzungen für die unteren Schranken, die für die Fixierung einer Variablen y_{ij} gelten, in den Fällen, in denen sich die Fixierung von der Lösung des y-Entscheidungsproblems unterscheidet, durch die Addition des Betrags der reduzierten Kosten und $z_{KS}(\bar{\omega})$:

1. Falls $\tilde{g}_{ij} \geq 0$ (im y-Entscheidungsproblem wurde $\bar{y}_{ij} = 0$ gesetzt):

a) $z_{KSy_{ij}=1} \geq z_{KS}(\bar{\omega})_{y_{ij}=1} = z_{KS}(\bar{\omega}) + \tilde{g}_{ij}$

b) $z_{KSy_{ij}=0} \geq z_{KS}(\bar{\omega})_{y_{ij}=0} = z_{KS}(\bar{\omega})$

2. Falls $\tilde{g}_{ij} < 0$ (im y-Entscheidungsproblem wurde $\bar{y}_{ij} = 1$ gesetzt):

a) $z_{KSy_{ij}=1} \geq z_{KS}(\bar{\omega})_{y_{ij}=1} = z_{KS}(\bar{\omega})$

b) $z_{KSy_{ij}=0} \geq z_{KS}(\bar{\omega})_{y_{ij}=0} = z_{KS}(\bar{\omega}) + |\tilde{g}_{ij}|$

Auch die Bedingungen für die Fixierung einer Variablen decken sich mit denen der Kürzeste-Wege-Relaxation:

- Fixiere y_{ij} auf 0, falls $\tilde{g}_{ij} \geq 0$ und $z_{KS}(\bar{\omega}) + \tilde{g}_{ij} \geq \hat{z}_{CNDP}$.
- Fixiere y_{ij} auf 1, falls $\tilde{g}_{ij} < 0$ und $z_{KS}(\bar{\omega}) + |\tilde{g}_{ij}| \geq \hat{z}_{CNDP}$.

Beide Verfahren zur Variablenfixierung sind nur jeweils in ihrer Relaxation anwendbar. Für einen Branch-and-Bound Algorithmus wird man sich jedoch zur Berechnung der unteren Schranke für diejenige der beiden Relaxationen entscheiden, die nach den Gesichtspunkten Effizienz und Qualität der Schranke überlegen ist. In den nächsten beiden Abschnitten wird erläutert, wie man im Rahmen eines Branch-and-Bound Algorithmus trotzdem beide Methoden der Variablenfixierung nutzen kann.

2.9.2 Kombinierte Variablenfixierung in der Kürzeste-Wege-Relaxation

Nehmen wir also an, wir berechnen im Branch-and-Bound Algorithmus die unteren Schranken mit Hilfe der Kürzeste-Wege-Relaxation. In einer Iteration i des Subgradientenverfahrens lösen wir mit den aktuellen Lagrange-Multiplikatoren (α^i, β^i) die $|\mathscr{C}|$ Kürzeste-Wege-Probleme $SP^k(\alpha^i, \beta^i)$ mit Dijkstras Algorithmus, berechnen einen Subgradienten und eine neue Suchrichtung und erhalten schließlich neue Lagrange-Multiplikatoren $(\alpha^{i+1}, \beta^{i+1})$. Eventuell starten wir nach der Lösung der Unterprobleme die Kürzeste-Wege-Variablenfixierung.

Um hier nun die Rucksack-Variablenfixierung anwenden zu können, müssen wir in der Lage sein, die Zielfunktion der Rucksack-Relaxation $z_{KS}(\omega)$ auszuwerten. Zu diesem Zweck benötigen wir "sinnvolle" Lagrange-Multiplikatoren ω, d.h. solche, für die $z_{KS}(\omega)$ konvergiert und die bei optimalen (α^*, β^*) ebenfalls optimal sind mit $z_{KS}(\omega^*) = z_{SP}(\alpha^*, \beta^*)$. Die Grundidee bei der kombinierten Variablenfixierung besteht nun darin, die optimale duale Lösung der Unterprobleme $SP^k(\alpha, \beta)$ als Lagrange-Multiplikatoren ω zu verwenden.

Für die Kürzeste-Wege-Unterprobleme $SP^k(\alpha, \beta)$ liefert der Dijkstra Algorithmus direkt eine dual optimale Lösung als Distanzen $\bar{\omega}_i^k$ vom Startknoten $O(k)$ zu den übrigen Knoten $i \in \mathscr{N}$ (für einen Beweis siehe z.B. [Papadimitriou and Steiglitz, 1982], Abschnitt 5.4). Wir lösen nun die entsprechenden Rucksack-Probleme $KS^{(i,j)}(\bar{\omega})$, erhalten reduzierte Fixkosten $\tilde{g}_{ij}$ und einen Wert für $z_{KS}(\bar{\omega})$, und können damit die Bedingungen der Rucksack-Variablenfixierung prüfen.

2.9.3 Kombinierte Variablenfixierung in der Rucksack-Relaxation

Wenn zur Berechnung der unteren Schranke die Rucksack-Relaxation verwendet wird, lösen wir in jeder Iteration $|\mathscr{A}|$ Rucksack-Probleme $KS^{(i,j)}(\omega)$ mittels Algorithmus 2. Zur Anwendung der Kürzeste-Wege-Variablenfixierung werden wieder Lagrange-Multiplikatoren $(\bar{\alpha}, \bar{\beta})$ zur Auswertung der Lagrange-Zielfunktion $z_{SP}(\bar{\alpha}, \bar{\beta})$ benötigt. In diesem Fall ist nicht sofort klar, wie wir aus Algorithmus 2 dual optimale Lösungen für die $KS^{(i,j)}(\omega)$ erhalten.

Mit Hilfe eines Simplextableaus kann man jedoch leicht die gewünschten Dualwerte $(\bar{\alpha}_{ij}, \bar{\beta}_{ij})$ für das Rucksackproblem $KS^{(i,j)}(\omega)$ der Kante $(i,j) \in \mathscr{A}$ herleiten. Sei $s \in \mathbb{N}$ die Anzahl der negativen Kostenkoeffizienten $\tilde{c}^k_{ij}$ in $KS^{(i,j)}(\omega)$ und seien die entsprechenden Variablen $\vec{x}^k_{ij}$ für $k \leq s$ nach aufsteigenden Kostenkoeffizienten sortiert. Sei $\hat{k}$ das Element, mit dem der Algorithmus abbricht, also $\hat{k} = \max\{k \,|\, \vec{x}^k_{ij} > 0\} + 1$. Dann setzen wir $(\bar{\alpha}_{ij}, \bar{\beta}_{ij})$ wie folgt:

1. Falls $\hat{k} < s+1$, d.h. die Kantenkapazität u_{ij} wird voll ausgeschöpft, setze
   - $\bar{\alpha}_{ij} = -\tilde{c}^{\hat{k}}_{ij}$ und
   - $\bar{\beta}^l_{ij} = \tilde{c}^{\hat{k}}_{ij} - \tilde{c}^l_{ij}$ für alle $l < \hat{k}$ und $\bar{\beta}^l_{ij} = 0$ für alle $l \geq \hat{k}$.
2. Falls $\hat{k} = s+1$, d.h. die Kantenkapazität u_{ij} wird nicht voll ausgeschöpft, setze
   - $\bar{\alpha}_{ij} = 0$ und
   - $\bar{\beta}^l_{ij} = -\tilde{c}^l_{ij}$ für alle $l < \hat{k}$ und $\bar{\beta}^l_{ij} = 0$ für alle $l \geq \hat{k}$.

Satz 2.2 *$(\bar{\alpha}_{ij}, \bar{\beta}_{ij})$ ist eine dual optimale Lösung für $KS^{(i,j)}(\omega)$.*

Beweis. Das duale Problem $DKS^{(i,j)}(\omega)$ zu $KS^{(i,j)}(\omega)$ lautet:

$$\begin{array}{rcll} z^{(i,j)}_{DKS}(\omega) & = & \min -u_{ij}\alpha_{ij} - \sum\limits_{k \in \mathscr{C}} d^k_{ij}\beta^k_{ij} & \\ & & \text{u.d.N.} \quad \alpha_{ij} + \beta^k_{ij} \geq -\tilde{c}^k_{ij} & \forall k \in \mathscr{C} \\ & & \qquad\quad \alpha_{ij} \geq 0 & \\ & & \qquad\quad \beta^k_{ij} \geq 0 & \forall k \in \mathscr{C} \end{array}$$

Als Erstes zeigen wir, dass unsere Lösung dual zulässig ist:

1. Fall: $\hat{k} < s+1$.
   Wegen $\hat{k} \leq s$ ist $\bar{\alpha}_{ij} = -\tilde{c}^{\hat{k}}_{ij} \geq 0$. Außerdem ist $\bar{\beta}^l_{ij} = \tilde{c}^{\hat{k}}_{ij} - \tilde{c}^l_{ij} \geq 0$ für $l < \hat{k}$, denn wegen der Sortierung ist ja $\tilde{c}^l_{ij} \leq \tilde{c}^{\hat{k}}_{ij}$. Für $l \geq \hat{k}$ ist $\bar{\beta}^l_{ij} = 0$. Alle Werte sind also nicht-negativ. Bezüglich der zweiten Nebenbedingung gilt $\bar{\alpha}_{ij} + \bar{\beta}^l_{ij} = -\tilde{c}^{\hat{k}}_{ij} + \tilde{c}^{\hat{k}}_{ij} - \tilde{c}^l_{ij} = -\tilde{c}^l_{ij}$ für $l < \hat{k}$ und $\bar{\alpha}_{ij} + \bar{\beta}^l_{ij} = -\tilde{c}^{\hat{k}}_{ij} \geq -\tilde{c}^l_{ij}$ für $l \geq \hat{k}$.
2. Fall: $\hat{k} = s+1$.
   Für alle $l < \hat{k}$ gilt $\bar{\alpha}_{ij} = 0$ und $\bar{\beta}^l_{ij} = -\tilde{c}^l_{ij} \geq 0$, denn es ist $l \leq s$. Für alle $l \geq \hat{k}$ ist $\bar{\beta}^l_{ij} = 0$, alle Multiplikatoren sind also nicht-negativ. Außerdem ist in diesem Fall $\bar{\alpha}_{ij} + \bar{\beta}^l_{ij} = -\tilde{c}^l_{ij}$ für alle $1 \leq l \leq |\mathscr{C}|$.

Bleibt für den Beweis der Optimalität der dualen Lösung (und gleichzeitig auch der primalen aus Algorithmus 2) die Gleichheit der Zielfunktionswerte $z^{(i,j)}_{KS}(\omega)$ und $z^{(i,j)}_{DKS}(\omega)$ für $\bar{x}_{ij}$ und $(\bar{\alpha}_{ij}, \bar{\beta}_{ij})$ zu zeigen:

1. Fall: $\hat{k} < s+1$.
   Es gilt $z^{(i,j)}_{KS}(\omega) = \sum_{l \in \mathscr{C}} \tilde{c}^l_{ij}\vec{x}^l_{ij} = \sum_{l<\hat{k}} \tilde{c}^l_{ij} d^l_{ij} + \tilde{c}^{\hat{k}}_{ij}(u_{ij} - \sum_{l<\hat{k}} d^l_{ij}) = \sum_{l<\hat{k}} d^l_{ij}(\tilde{c}^l_{ij} - \tilde{c}^{\hat{k}}_{ij}) + u_{ij}\tilde{c}^{\hat{k}}_{ij} = -\sum_{l \in \mathscr{C}} d^l_{ij}\bar{\beta}^l_{ij} - u_{ij}\bar{\alpha}_{ij} = z^{(i,j)}_{DKS}(\omega)$.

2. Fall: $\hat{k} = s+1$.
   Es gilt $z_{KS}^{(i,j)}(\omega) = \sum_{l\in\mathscr{C}} \tilde{c}_{ij}^l \vec{x}_{ij}^l = \sum_{l\leq s} \tilde{c}_{ij}^l d_{ij}^l = -\sum_{l\in\mathscr{C}} d_{ij}^l \bar{\beta}_{ij}^l - u_{ij}\bar{\alpha}_{ij} = z_{DKS}^{(i,j)}(\omega)$.

□

Wir sind also nun in der Lage, in beiden Relaxationen die Algorithmen zur Variablenfixierung der jeweils anderen Relaxation zusätzlich anzuwenden. Dies kann in jeder oder nur in ausgewählten Iterationen des Subgradientenverfahrens geschehen. Zur Auswertung der zusätzlichen Lagrange-Zielfunktion müssen jeweils auch die entsprechenden Unterprobleme gelöst werden, was natürlich zu einem Mehraufwand bei der Berechnung der Schranke führt. Es gilt also, unter den vielen denkbaren Varianten diejenige Kombination der Variablenfixierungsstrategien zu finden, die den günstigsten Tradeoff zwischen dem nötigen Mehraufwand und der erreichten Verkleinerung des Branch-and-Bound Baums liefert.

2.9.4 Variablenfixierung mit Kardinalitätsbedingungen

Eine Hinzunahme der Kardinalitätsbedingungen macht die Verfahren zur Variablenfixierung etwas aufwendiger, aber auch effektiver. Der Mehraufwand entsteht, wenn sich die Lösung des y-Entscheidungsproblem durch Algorithmus 8 an den Grenzen des Kardinalitätsintervalls befindet. Nehmen wir also an, z_R sei der Zielfunktionswert der verwendeten Relaxation, also die aktuelle untere Schranke, $\tilde{f}$ seien die zugehörigen reduzierten Fixkosten, $\hat{z}_{CNDP}$ die aktuelle obere Schranke und y die Lösung des Entscheidungsproblems durch Algorithmus 8.

Abbildung 2.10 zeigt eine Lösung y, aufsteigend sortiert nach den reduzierten Fixkosten $\tilde{f}$, für ein gegebenes Kardinalitätsintervall $[a,b]$ und $s = \sum y_{e_i}$. Mit e_k und e_l bezeichnen wir im Folgenden Kanten, für die gilt:

$$\begin{aligned} k &= \mathit{argmax}_{0<i<|\mathscr{A}|}\{\tilde{f}_{e_i} : y_{e_i} = 1\} \quad \mathit{und} \\ l &= \mathit{argmin}_{0<i<|\mathscr{A}|}\{\tilde{f}_{e_i} : y_{e_i} = 0\}. \end{aligned}$$

Mit diesen Vorbetrachtungen unterscheiden wir nun für beide Fixierungsrichtungen drei mögliche Fälle. Für $y_{e_i} = 1$ gilt:

1. Fall: $a < s < b$.

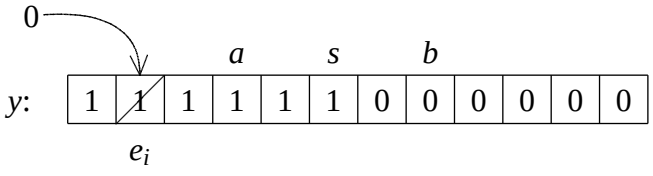


Abbildung 2.10: Lösung y im Fall $y_{e_i} = 1$ und $a < s < b$

In diesem Fall ändert sich nichts. Hier gilt:

$$\mathit{Fixiere}\, y_{e_i}\, \mathit{auf}\, 1 \iff z_R - \tilde{f}_{e_i} \geq \hat{z}_{CNDP}.$$

2. Fall: $s = a$.

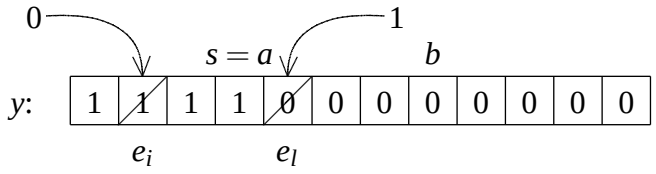


Abbildung 2.11: Lösung y im Fall $y_{e_i} = 1$ und $s = a$

Hier müssen wir bei der Berechnung der unteren Schranke für $y_{e_i} = 0$ zur Einhaltung der unteren Intervallgrenze a die nächstbeste Kante, nämlich e_l, auf 1 setzen. Für die Variablenfixierung gilt:

$$Fixiere\, y_{e_i}\, auf\, 1 \iff z_R - \tilde{f}_{e_i} + \tilde{f}_{e_l} \geq \hat{z}_{CNDP}.$$

3. Fall: $s = b \wedge \tilde{f}_{e_l} < 0$.

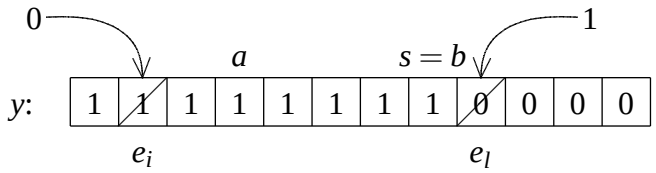


Abbildung 2.12: Lösung y im Fall $y_{e_i} = 1$ und $s = b$

In diesem Fall wurde eine eigentlich lohnende Kante zur Einhaltung der oberen Intervallgrenze b auf 0 gesetzt. Für den Test $y_{e_i} = 0$ müssen wir diese Kante mitberücksichtigen:

$$Fixiere\, y_{e_i}\, auf\, 1 \iff z_R - \tilde{f}_{e_i} + \tilde{f}_{e_l} \geq \hat{z}_{CNDP}.$$

Für $y_{e_i} = 0$ gilt analog:

1. Fall: $a < s < b$.

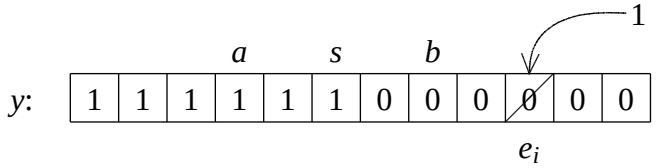


Abbildung 2.13: Lösung y im Fall $y_{e_i} = 0$ und $a < s < b$

Dies ist wieder der Standardfall:

$$Fixiere\, y_{e_i}\, auf\, 1 \iff z_R + \tilde{f}_{e_i} \geq \hat{z}_{CNDP}.$$

2. Fall: $s = b$.

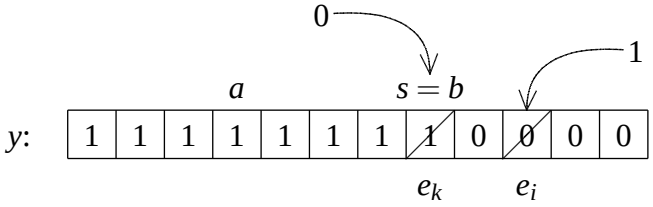


Abbildung 2.14: Lösung y im Fall $y_{e_i} = 0$ und $s = b$

Hier streichen wir die schlechteste auf 1 gesetzte Kante, um e_i hinzunehmen zu können. Hier gilt also:

$$Fixiere\, y_{e_i}\, auf\, 1 \iff z_R + \tilde{f}_{e_i} - \tilde{f}_{e_k} \geq \hat{z}_{CNDP}.$$

3. Fall: $s = a \wedge \tilde{f}_{e_k} \geq 0$.

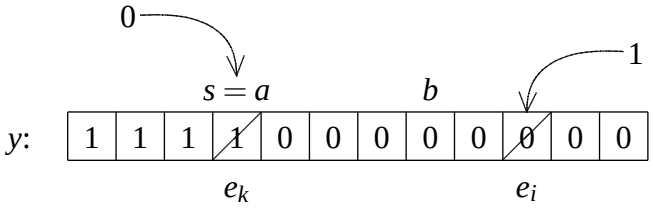


Abbildung 2.15: Lösung y im Fall $y_{e_i} = 0$ und $s = a$

In diesem Fall musste mindestens eine nicht lohnende Kante hinzugenommen werden, um die Mindestkantenzahl a zu erreichen. Diese Kante können wir streichen, wenn wir $y_{e_i} = 1$ testen. Für die Fixierung gilt:

$$Fixiere\, y_{e_i}\, auf\, 1 \iff z_R + \tilde{f}_{e_i} - \tilde{f}_{e_k} \geq \hat{z}_{CNDP}.$$

2.10 Heuristische Variablenfixierung

Bisher haben wir unseren Branch-and-Bound Algorithmus immer als exaktes Verfahren dargestellt. Die Erfahrung der letzten Jahre hat jedoch gezeigt, dass größere Probleminstanzen des CNDP nicht exakt gelöst werden können, und es sind daher eine Reihe von Heuristiken entwickelt worden. Auch unser Verfahren kann leicht in eine Heuristik umgewandelt werden. Zu diesem Zweck haben wir zwei Ansätze aus [Holmberg and Yuan, 2000] implementiert, die beide darauf basieren, zusätzliche Variablen zu fixieren und dafür die Information aus der Berechnung der Lagrange-Schranke zu nutzen. Das Verfahren in [Holmberg and Yuan, 2000] wurde von den Autoren für das Subgradientenverfahren präsentiert. In dieser Arbeit führten wir eine entsprechende Anpassung für das Bundle-Verfahren durch.

α–Fixierung

Die Idee des ersten Ansatzes ist folgende: Wenn in der y-Lösung der Lagrange-Relaxation eine Designvariable y_{ij} im Verlauf des Subgradientenverfahrens (oder des Bundle-Verfahrens) ständig auf 1 gesetzt wird, so wird die entsprechende Kante höchstwahrscheinlich auch in einer optimalen Lösung des Gesamtproblems enthalten sein. Andererseits wird eine Kante (i,j) wahrscheinlich nicht in der optimalen Lösung enthalten sein, wenn y_{ij} in den meisten Subgradienteniterationen auf 0 gesetzt wird. Es macht also Sinn, Fixierungsregeln zu definieren, die auf den Häufigkeiten der y_{ij} basieren.

Idealerweise wird man eine Variable fixieren, wenn ihr Wert in allen Iterationen des Subgradientenverfahrens (oder des Bundle-Verfahrens) übereinstimmt. Um den Ansatz etwas flexibler zu machen, wird ein Parameter $\alpha \in [0, 0.5]$ eingeführt, der eine gewisse Anzahl von Abweichungen erlaubt. Wenn $\alpha = 0$ ist, dann gilt obiger Idealfall. Sonst wird der Wert $(1-\alpha) * M$ als Grenze benutzt, um zu entscheiden, ob eine Variable nach M Iterationen fixiert werden soll. Sei y_{ij}^l der Wert von y_{ij} in der l-ten Iteration. Dann können wir diese Fixierungsregel wie folgt zusammenfassen:

- Fixiere y_{ij} auf 1, wenn $\sum_{l=1}^{M} y_{ij}^l \geq (1-\alpha) * M$.
- Fixiere y_{ij} auf 0, wenn $\sum_{l=1}^{M} y_{ij}^l \leq \alpha * M$.

Der Parameter α kann auch dynamisch im Verlauf des Branch-and-Bound Verfahrens verändert werden. Ein vielversprechende Strategie ist, mit einem relativ großen Wert für α zu beginnen, um schnell eine zulässige Lösung zu generieren. Ein guter Kompromiss zwischen Laufzeit und Lösungsqualität kann dann erreicht werden, indem der Wert von α immer weiter reduziert wird.

β–Fixierung

Dieser Ansatz benutzt die reduzierten Fixkosten $\tilde{f}$ (bzw. $\tilde{g}$) zur heuristischen Fixierung von Variablen. Die reduzierten Fixkosten sind eng mit der y-Lösung im Lagrange-Unterproblem verknüpft, denn nach ihrem Vorzeichen wird jeweils die Belegung von y bestimmt. Wenn für eine Kante (i,j) etwa $\tilde{f}_{ij} \ll 0$ oder $\tilde{f}_{ij} \gg 0$ gilt, so deutet dies darauf hin, dass die Kante wahrscheinlich auch in einer optimalen Lösung enthalten bzw. nicht enthalten ist. β–Fixierung benutzt daher den Wert $|\tilde{f}_{ij}|$, um Variablen zu fixieren.

Der Parameter $\beta \in [0,1]$ gibt dabei den Anteil an der Gesamtzahl der Designvariablen an, der in jedem Knoten des Branch-and-Bound Baums fixiert werden soll. Sei n die Anzahl der unfixierten Kanten im ersten Knoten (meist ist $n = |\mathscr{A}|$), dann fixieren wir in jedem Knoten genau $\lceil \beta n \rceil$ Variablen (bzw. $|\mathscr{A}_*|$, falls $|\mathscr{A}_*| < \lceil \beta n \rceil$). Um die Variablen zu bestimmen, die fixiert werden, akkumulieren wir die reduzierten Fixkosten für jede Kante über alle Iterationen zu einem Wert $|R_{ij}|$. Dazu setzen wir in der ersten Iteration $R_{ij} = \tilde{f}_{ij}^1$ und dann $R_{ij} = \gamma * R_{ij} + \tilde{g}_{ij}^l$. In unserer Implementierung ist $\gamma = 0.5$. Dieser Parameter sorgt dafür, dass die reduzierten Fixkostenwerte umso stärker berücksichtigt werden, je besser die untere Schranke ist. Nach Beendigung des Subgradientenverfahrens (oder des Bundle Verfahrens) fixieren wir dann Variablen mit Algorithmus 9.

Der Vorteil von β–Fixierung im Vergleich zu α–Fixierung ist, dass mit dem Parameter β die maximale Größe des Branch-and-Bound Baums festgelegt werden kann. Für $\beta = 0.1$ werden

Algorithmus 9 Heuristische Variablenfixierung mit β–Fixierung

1: *setze* $m = 1$
2: **repeat**
3: *wähle* $(i', j') = argmax_{(i,j)\in\mathscr{A}_*}|R_{ij}|$
4: **if** $(R_{i'j'} < 0)$ **then**
5: *fixiere* $y_{i'j'}$ auf 1
6: **else**
7: *fixiere* $y_{i'j'}$ auf 0
8: *setze* $\mathscr{A}_* = \mathscr{A}_* \setminus (i', j')$ *und* $m = m + 1$
9: **until** $(m \geq min(|\mathscr{A}_*|, \lceil \beta n \rceil))$

beispielsweise in jedem Level des Baums 10% aller Kanten fixiert, die maximale Tiefe des Baums ist also 10 und die maximale Knotenanzahl $2^{10} - 1$. Außerdem wird der Algorithmus zu einem exakten Verfahren für $\beta = 0$ (keine heuristische Fixierung).

2.11 Zusätzliche Ungleichungen

Neben den Kardinalitätsbedingungen werden in unserem Algorithmus weitere zusätzliche Ungleichungen benutzt. Dies sind einerseits gültige Ungleichungen für das Netzwerkentwurfproblem, die auf der Tatsache basieren, dass die Kapazität eines Start- und Zielorte trennenden Schnitts in jeder zulässigen Lösung groß genug sein muss, um den Transportbedarf über den Schnitt zu decken. Weiterhin werden auch Ungleichungen verwendet, die in bestimmten Ästen des Suchbaums Lösungen mit zu hohen Kosten abschneiden.

Um die Struktur der Lagrange-Teilprobleme nicht zu zerstören, gehen diese zusätzlichen Ungleichungen nicht als Restriktionen in die Lagrange-Relaxation ein, sondern werden mit neuen Lagrange-Multiplikatoren in die Zielfunktion der Lagrange-Relaxation aufgenommen.

2.11.1 Schnittungleichungen

Sei die Knotenmenge $\mathscr{N}$ des Graphen $G = (\mathscr{N}, \mathscr{A})$ in zwei nichtleere disjunkte Mengen S und $\bar{S}$ zerlegt. Die Menge der von S nach $\bar{S}$ führenden Kanten (d.h. der Kanten mit Anfangsknoten in S und Endknoten in $\bar{S}$) heißt (S von $\bar{S}$ trennender) *Schnitt* in G und wird mit $(S, \bar{S})$ bezeichnet.

Sei $\mathscr{C}(S, \bar{S}) \subseteq \mathscr{C}$ die Menge der Güter mit Startorten in S und Zielorten in $\bar{S}$. In jeder zulässigen Lösung des Netzwerkentwurfproblems müssen mindestens $r_{(S,\bar{S})} = \sum_{k\in\mathscr{C}(S,\bar{S})} r^k$ Einheiten über den Schnitt $(S, \bar{S})$ fließen. Also muss die Gesamtkapazität der Kanten von $(S, \bar{S})$ in jeder zulässigen Lösung mindestens die Größe $r_{(S,\bar{S})}$ haben:

$$\sum_{(i,j)\in(S,\bar{S})} u_{ij} y_{ij} \geq r_{(S,\bar{S})}. \tag{2.16}$$

Diese als *Schnittungleichung* (cutset inequality) bekannte Ungleichung gilt für alle Teilmengen $S \subset \mathscr{N}$.

Da die Schnittungleichung (2.16) nicht nur für das Netzwerkentwurfproblem selbst, sondern auch für seine LP-Relaxation gültig ist, hätte das Einbeziehen dieser Ungleichung in die

Beschreibung des Problems keinen Einfluss auf die Güte der LP- oder Lagrange-Schranke. Wenn man allerdings die Ganzzahligkeit der y-Variablen ausnutzt, kann man aus Schnittungleichungen gültige Ungleichungen für das Netzwerkentwurfproblem herleiten, die diese Schranken verbessern können. Eine solche auf Schnittungleichungen aufbauende Familie gültiger Ungleichungen wird im Abschnitt 2.11.2 behandelt.

Identifizieren verletzter Schnittungleichungen

Im Gegensatz zu den zulässigen Lösungen der LP-Relaxation, muss die Lösung eines Lagrange-Teilproblems nicht notwendigerweise alle Schnittungleichungen (2.16) erfüllen. Um Schnittungleichungen zu identifizieren, die in der Lösung (x,y) eines Lagrange-Teilproblems verletzt werden, bietet sich die folgende einfache Prozedur an: Für jedes Gut $k \in \mathscr{C}$ bestimmt man die Menge der Knoten $S_k \subseteq \mathscr{N}$, die von dem Startort $o(k)$ aus über die Kanten (i,j) mit $y_{ij} = 1$ erreicht werden können (Abbildung 2.16). Ist der Zielort $d(k) \notin S_k$, hat man damit einen

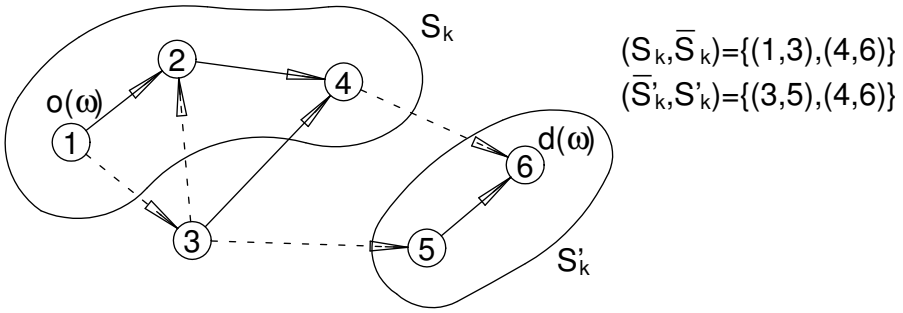


Abbildung 2.16: Knotenmengen S_k und S'_k. Die Kanten (i,j) mit $y_{ij} = 0$ sind gestrichelt dargestellt.

Schnitt $(S_k, \bar{S_k})$ gefunden, dessen Kapazität in der aktuellen Lösung der Lagrange-Relaxation gleich Null ist und in jeder zulässigen Lösung des Netzwerkentwurfproblems groß genug sein muss, um r^k Einheiten des Gutes k durchzulassen. Um eine verletzte Schnittungleichung $\sum_{(i,j)\in(S_k,\bar{S_k})} u_{ij} y_{ij} \geq r_{(S_k,\bar{S_k})}$ zu formulieren, bleibt jetzt nur noch den Gesamttransportbedarf $r_{(S_k,\bar{S_k})}$ der Güter mit Startorten in S_k und Zielorten in $\bar{S_k}$ auszurechnen.

Für ein Gut k, für das es bei der aktuellen Belegung der y-Variablen keinen Weg von dem Start- zum Zielknoten gibt, wird zusätzlich zu $(S_k, \bar{S_k})$ ein weiterer Schnitt $(\bar{S'_k}, S'_k)$ definiert. Die Menge S'_k wird hierbei als die Menge der Knoten festgelegt, von denen es Wege nach $d(k)$ über die Kanten (i,j) mit $y_{ij} = 1$ gibt (Abbildung 2.16).

Weil die Lösung eines Lagrange-Teilproblems (vor allem bei einem schlecht gewählten Multiplikator ω) oftmals viele Schnittungleichungen verletzt und wir uns in erster Linie für Ungleichungen interessieren, die typischerweise in den Lösungen von Teilproblemen verletzt werden, die gute untere Schranken liefern, wird diese Prozedur in der Regel nicht direkt auf die Lösungen einzelner Lagrange-Teilprobleme angewandt, sondern auf Vektoren $\bar{y}^{(l)}$, die sich jeweils aus den optimalen Lösungen $y^{(l)}$ und $y_{best}^{(l-1)}$ des aktuellen Teilproblems und des Teilproblems mit dem bisher größten Wert $z_R(\omega)$ als $\bar{y}_{ij}^{(l)} = \max\{y_{ij}^{(l)}, y_{ij\,best}^{(l-1)}\}\ \forall (i,j) \in \mathscr{A}$ zusammensetzen. Ferner brechen wir die die Suche nach verletzten Schnittungleichungen ab, sobald die ersten Schnitte $(S_k, \bar{S_k})$ und $(\bar{S'_k}, S'_k)$ mit zu wenigen Kanten gefunden werden.

Eine Schwäche der oben beschriebenen Prozedur zum Identifizieren verletzter Schnittungleichungen ist, dass man damit nur solche Schnitte findet, die bei der aktuellen Belegung der Designvariablen *keine* Kanten enthalten. Die Schnitte $(S,\bar{S})$, deren Kapazität größer als Null, aber kleiner als der Transportbedarf $r_{(S,\bar{S})}$ ist, werden übersehen. Um die Suche nach verletzten Schnittungleichungen effektiver zu machen, kann die Information aus der Berechnung oberer Schranken benutzt werden.

Im Laufe der Bundle-Algorithmus wird in regelmäßigen Abständen die Lösung des primalen Teilproblems $PS(\bar{\mathscr{A}})$ mit einer durch y-Lösungen der Lagrange-Teilprobleme festgelegten Kantenmenge $\bar{\mathscr{A}} = \{(i,j) \in \mathscr{A} : y_{ij}^{(l)} = 1 \text{ oder } y_{ij}{}_{best}^{(l-1)} = 1\}$ gestartet. Wir betrachten nun die folgende Situation: Die im Abschnitt 2.7.3 beschriebene Heuristik für das primale Teilproblem schlägt fehl bei dem Versuch für ein Gut k einen Fluss der Stärke r^k zu finden und bricht ab. Wegen der in der Heuristik gewählten Formulierung des MCF beschreibt der Vektor x^k in dieser Situation einen maximalen Fluss des Gutes k in dem Graphen $\bar{G} = (\mathscr{N}, \bar{\mathscr{A}})$ mit den (nach Festlegen anderer Flüsse) noch vorhandenen Kapazitäten $\hat{u}_{ij}$. Verringert man die unbenutzten Kapazitäten $\hat{u}_{ij}$ um die Flusswerte x_{ij}^k, so erhält man ein Flussnetzwerk, in dem jeder von dem Startort $o(k)$ zum Zielort $d(k)$ führende Weg mindestens eine Kante (i,j) mit der Restkapazität $\hat{u}_{ij} = 0$ enthält. Wir definieren nun S_k als die Menge der von dem Startort $o(k)$ über die Kanten $(i,j) \in \bar{\mathscr{A}}$ mit $\hat{u}_{ij} > 0$ erreichbaren Knoten und S'_k als die Menge der Knoten, von denen es Wege zu $d(k)$ über die Kanten $(i,j) \in \bar{\mathscr{A}}$ mit positiven Restkapazitäten $\hat{u}_{ij} > 0$ gibt, und betrachten die Schnitte $(S_k,\bar{S}_k)$ und $(\bar{S}'_k,S'_k)$ in $G = (\mathscr{N},\mathscr{A})$. In der von der Heuristik gelieferten (unzulässigen) Lösung x ist der Transportbedarf von S_k nach $\bar{S}_k$ nicht gedeckt, aber die Kapazitäten der Kanten $(i,j) \in (S_k,\bar{S}_k) \cap \bar{\mathscr{A}}$ sind bereits ausgeschöpft. Daher erscheint es naheliegend, zu prüfen, ob die Gesamtkapazität der Kanten aus $(S_k,\bar{S}_k) \cap \bar{\mathscr{A}}$ genügt, um den Transportbedarf $r_{(S_k,\bar{S}_k)}$ zwischen S_k und $\bar{S}_k$ zu decken. Gilt $\sum_{(i,j)\in(S_k,\bar{S}_k)\cap\bar{\mathscr{A}}} u_{ij} < r_{(S_k,\bar{S}_k)}$, so stellt $\sum_{(i,j)\in(S_k,\bar{S}_k)} u_{ij}y_{ij} \geq r_{(S_k,\bar{S}_k)}$ eine in dem primalen Teilproblem $PS(\bar{\mathscr{A}})$ sowie in den Lösungen $y^{(l)}$ und $y_{best}^{(l-1)}$ entsprechender Lagrange-Teilprobleme verletzte Schnittungleichung dar. Ähnliches gilt auch für den Schnitt $(\bar{S}'_k,S'_k)$.

Wir starten die Suche nach verletzten Schnittungleichungen in jeder M-ten Iteration des Bundle-Algorithmus und speichern diese Ungleichungen ab, um sie später zur Formulierung gültiger Überdeckungsungleichungen oder zum Fixieren von Variablen zu nutzen.

2.11.2 Überdeckungsungleichungen

In jedem Start- und Zielorte trennenden Schnitt $(S,\bar{S})$ in G gibt es Teilmengen der Kanten $C \subseteq (S,\bar{S})$ mit folgender Eigenschaft: Die Gesamtkapazität der Kanten aus $(S,\bar{S}) \setminus C$ ist kleiner als der Transportbedarf von S nach $\bar{S}$. In einer zulässigen Lösung des Netzwerkentwurfproblems muss mindestens eine Kante aus C zum Gütertransport benutzt werden. Das Einbeziehen der für solche Kantenmengen gültigen Ungleichungen $\sum_{(i,j)\in C} y_{ij} \geq 1$ in die Berechnung der unteren Schranke kann zu einer Verbesserung der Schranke und damit auch zum Verkleinern des Branch-and-Bound-Baums beitragen.

Für eine gegebene Schnittungleichung $\sum_{(i,j)\in(S,\bar{S})} u_{ij}y_{ij} \geq r_{(S,\bar{S})}$ sei $Y_{(S,\bar{S})}$ die Menge der sie erfüllenden Belegungen der y-Variablen,

$$Y_{(S,\bar{S})} = \left\{ y \in \{0,1\}^{|(S,\bar{S})|} : \sum_{(i,j)\in(S,\bar{S})} u_{ij}y_{ij} \geq r_{(S,\bar{S})} \right\}.$$

Wir bezeichnen eine Kantenmenge $C \subseteq (S,\bar{S})$ als *Überdeckung* in Bezug auf $(S,\bar{S})$, wenn die Gesamtkapazität der Kanten aus $(S,\bar{S}) \setminus C$ nicht ausreicht, um den Transportbedarf von S nach $\bar{S}$ zu decken, das heißt wenn

$$\sum_{(i,j)\in(S,\bar{S})\setminus C} u_{ij} < r_{(S,\bar{S})}$$

ist. Eine Überdeckung $C \subseteq (S,\bar{S})$ ist *minimal*, wenn das Öffnen jeder beliebigen Kante aus C (zusätzlich zu den Kanten von $(S,\bar{S}) \setminus C$) die Kapazität des Schnitts soweit erhöht, dass der Transportbedarf von S nach $\bar{S}$ erfüllt werden kann, das heißt wenn für alle $(p,q) \in C$ gilt

$$\sum_{(i,j)\in(S,\bar{S})\setminus C} u_{ij} + u_{pq} \geq r_{(S,\bar{S})}.$$

Behauptung 2.5 *Sei $C \subseteq (S,\bar{S})$ eine Überdeckung bezüglich $(S,\bar{S})$, so gilt für jedes $y \in Y_{(S,\bar{S})}$ und somit auch für alle zulässigen Lösungen des Netzwerkdesignproblems die* Überdeckungsungleichung[8]

$$\sum_{(i,j)\in C} y_{ij} \geq 1. \tag{2.17}$$

Ist C minimal, so definiert die Ungleichung (2.17) eine Facette der konvexen Hülle von $Y'_{(S,\bar{S})} = Y_{(S,\bar{S})} \cap \{y : y_{ij} = 1 \; \forall (i,j) \in (S,\bar{S}) \setminus C\}$ und eine $(|C|-1)$-dimensionale Randfläche von conv$(Y_{(S,\bar{S})})$. (vgl. [Nemhauser and Wolsey, 1988], [Klose, 2002])

Die Idee der Überdeckungsungleichung (2.17) ist einfach: Wenn die Gesamtkapazität der Kanten aus $(S,\bar{S}) \setminus C$ nicht ausreicht, um den Transportbedarf über den Schnitt $(S,\bar{S})$ zu decken, muss mindestens eine Kante aus C zum Gütertransport geöffnet werden. Ist die Überdeckung C nicht minimal, so ist die entsprechende Überdeckungsungleichung für die Beschreibung der konvexen Hülle von $Y_{(S,\bar{S})}$ redundant, weil sie offenbar als die Summe der Überdeckungsungleichung $\sum_{(i,j)\in C'} y_{ij} \geq 1$ für eine minimale Überdeckung C' und der Schrankenbedingungen $y_{ij} \geq 0$ für $(i,j) \in C \setminus C'$ ausgedrückt werden kann.

Nun stellt sich die Frage, ob das Einbeziehen einer Überdeckungsungleichung in die Zielfunktion der Lagrange-Relaxation zum Verschärfen der besten mit dieser Relaxation berechenbaren Schranke z_{LM} beitragen kann. Wie im Abschnitt 2.12.2 erklärt wird, trifft dieses nur für solche Ungleichungen zu, die ein Stück des zulässigen Bereichs der zum Lagrange-Multiplikator-Problem dualen Aufgabe (L) abschneiden. Im Falle der Rucksack-Relaxation (2.4.2) stimmt diese Aufgabe mit der LP-Relaxation des Netzwerkentwurfproblems überein (vgl. [Gendron and Crainic, 1994b]). Das heißt dass das Einbeziehen einer zusätzlichen Ungleichung in die Lagrange-Relaxation nur in dem Fall eine Verbesserung der Schranke z_{LM} bewirken kann, wenn diese Ungleichung für die LP-Relaxation nicht redundant ist. Wir zeigen:

[8] Die in dieser Arbeit verwendete Definition der Überdeckung wurde auch von Chouman et al. [Chouman et al., 2003] verwendet. Im Standardfall werden Überdeckungsungleichungen (cover inequalities) für Lösungsmengen binärer Rucksackprobleme $Y = \{y \in \{0,1\}^n : \sum_{i=1}^n a_i y_i \leq b\}$ mit $a \geq 0$ und $b \geq 0$ definiert und haben die Form $\sum_{i\in C} y_i \leq |C| - 1$. Durch Komplementierung der Variablen, d.h. Ersetzen von y_i durch $1 - \bar{y}_i$, kann eine solche Menge Y in die Form $\bar{Y} = \{\bar{y} \in \{0,1\}^n : \sum_{i=1}^n a_i \bar{y}_i \geq \sum_{i=1}^n a_i - b\}$ und die Überdeckungsungleichung $\sum_{i\in C} y_i \leq |C| - 1$ in die Form $\sum_{i\in C} \bar{y}_i \geq 1$ gebracht werden.

Behauptung 2.6 *Sei $(S,\bar{S})$ ein beliebiger Schnitt in G mit $r_{(S,\bar{S})} > 0$. Für die Überdeckung $C = (S,\bar{S})$ stellt (2.17) eine für die LP-Relaxation des Netzwerkentwurfproblems redundante Ungleichung dar.*

Beweis: Sei $k \in \mathscr{C}$ ein Gut mit $o(k) \in S$ und $d(k) \in \bar{S}$. In jeder zulässigen Lösung des Netzwerkentwurfproblems (oder seiner LP-Relaxation) fließen mindestens r^k Einheiten des Gutes k über die Kanten von $C = (S,\bar{S})$, somit ist $\sum_{(i,j)\in C} x_{ij}^k \geq r^k$. Aus der Kapazitäts-Nebenbedingung folgt damit $\sum_{(i,j)\in C} y_{ij} \geq \sum_{(i,j)\in C} \frac{x_{ij}^k}{d_{ij}^k} \geq \sum_{(i,j)\in C} \frac{x_{ij}^k}{r^k} \geq 1$. □

Beispiel

Wir zeigen nun an einem Beispiel, dass **minimale Überdeckungen** die LP-Relaxation (und damit auch die beste mit der Lagrange-Relaxation berechenbare Schranke z_{LM}) verbessern können.

In dem in der Abbildung 2.17, a gezeigten Netzwerk sollen $r^1 = 5$ Einheiten eines Gutes mit möglichst kleinen Kosten von 1 nach 4 transportiert werden. Die Kapazitäten, Transport- und Fixkosten sind an den Kanten in der Form u_{ij}, c_{ij}^1, f_{ij} geschrieben. In einer kostenminimalen Lösung der LP-Relaxation dieses Netzwerkentwurfproblems fließen vier Einheiten über die Kante $(1,4)$ und eine über die Kanten $(1,3)$ und $(3,4)$ (Abbildung 2.17, b). Das Aufheben der Ganzzahligkeitsforderung für Designvariablen in der LP-Relaxation führt dazu, dass die y-Variablen die kleinsten zulässigen Werte $y_{ij} = \max\{x_{ij}^1/u_{ij}, x_{ij}^1/r^1\}$ annehmen. Mit $y_{14} = 1$,

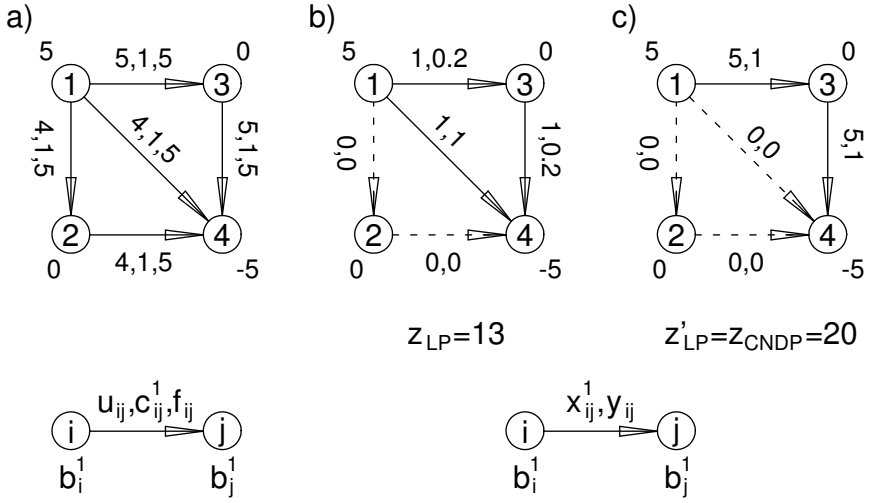


Abbildung 2.17: Ein Netzwerkentwurfproblem (a) und die optimalen Lösungen seiner LP-Relaxation (b) ohne zusätzliche Ungleichungen und (c) mit der Ungleichung $y_{13} + y_{24} \geq 1$.

$y_{13} = y_{34} = 0.2$ und $y_{12} = y_{24} = 0$ ist das keine zulässige Lösung des Netzwerkentwurfproblems. Betrachte nun die Zerlegung der Knotenmenge $\mathscr{N}$ in $S = \{1,2\}$ und $\bar{S} = \{3,4\}$. Die Gesamtkapazität der von S nach $\bar{S}$ führenden Kanten muss in jeder zulässigen Lösung mindestens so groß sein wie der Transportbedarf von 1 nach 4:

$$5y_{13} + 4y_{14} + 4y_{24} \geq 5. \tag{2.18}$$

Mit dieser Schnittungleichung können für $(S,\bar{S})$ zwei minimale Überdeckungen $C_1 = \{(1,3),(1,4)\}$ und $C_2 = \{(1,3),(2,4)\}$ identifiziert werden. Ein gleichzeitiges Sperren aller Kanten einer Überdeckung C_i, $i = 1,2$ verletzt die Forderung (2.18) und ist deshalb unzulässig. Also muss jede zulässige Lösung des Netzwerkentwurfproblems die Überdeckungsungleichungen

$$y_{13} + y_{14} \geq 1 \qquad \text{und} \qquad y_{13} + y_{24} \geq 1$$

erfüllen. Die in der Abbildung 2.17, b gezeigte optimale Lösung der LP-Relaxation des Problems verletzt die Überdeckungsungleichung $y_{13} + y_{24} \geq 1$ und kann durch Hinzufügen dieser Ungleichung zur Problembeschreibung ausgeschlossen werden. Die kostenminimale Lösung der um $y_{13} + y_{24} \geq 1$ erweiterten LP-Relaxation ist in der Abbildung 2.17, c dargestellt. Sie erfüllt mit $y_{13} = y_{34} = 1$ und $y_{12} = y_{14} = y_{24} = 0$ die Ganzzahligkeitsforderung für Designvariablen und ist somit auch für das Netzwerkentwurfproblem optimal.

Wir bemerken noch, dass auch Ungleichungen, die den Wert der besten durch die Lösung des Lagrange-Multiplikator-Problems berechenbaren Schranke z_{LM} nicht ändern, einen Einfluss auf die Konvergenz des Bundle-Verfahrens zu diesem Wert und die tatsächliche Güte der durch dieses Verfahren gelieferten Schranke haben können.

Identifizieren verletzter Überdeckungsungleichungen

Während der Lösung des Lagrange-Multiplikator-Problems durch das Bundle-Verfahren wird in unserem Algorithmus in regelmäßigen Abständen die Suche nach Überdeckungsungleichungen angestoßen, die von den im Bündel gespeicherten y-Lösungen verletzt werden. Solche Ungleichungen werden (gegebenenfalls nach dem Verschärfen durch eine Liftingprozedur) mit einem neuen Lagrange-Multiplikator in die Zielfunktion der Lagrange-Relaxation aufgenommen, sodass ihr Verletzen in den nachfolgenden Iterationen mit höheren Kosten bestraft werden kann.

Seien $\langle \omega^p, z^p, s^p, y^p \rangle$, $p \in I_\beta$ die aktuellen Einträge des Bündels β. Um den Wert des Lagrange-Multiplikators ω für die nächste Iteration festzulegen, wird im Bundle-Verfahren von Frangioni das quadratische Optimierungsproblem $(\Delta_{\beta t})$ gelöst. Die optimale Lösung dieses Problems $\theta^* \in \{\theta : \theta \geq 0, \sum_{p \in I_\beta} \theta_p = 1\}$ bestimmt die Gewichte der Subgradienten s^p bei der Wahl der Suchrichtung d (vgl. Abschnitt 2.6.2). Für eine bekannte Schnittungleichung

$$\sum_{(i,j) \in (S,\bar{S})} u_{ij} y_{ij} \geq r_{(S,\bar{S})} \tag{2.19}$$

sei $B = \{p \in I_\beta : \sum_{(i,j) \in (S,\bar{S})} u_{ij} y^p_{ij} < r_{(S,\bar{S})}\}$ die Menge der Indizes der im Bündel gespeicherten y-Lösungen, die diese Schnittungleichung verletzen, und $\theta^*_B = \sum_{p \in B} \theta^*_p$ das Gesamtgewicht der Subgradienten s^p mit $p \in B$ bei der Festlegung der Suchrichtung d. In dem Fall $\theta^*_B > 0$ werden in unserem Algorithmus die Lösungen y^p mit $p \in B$ zu einem Punkt $\tilde{y} = \sum_{p \in B} \frac{\theta^*_p}{\theta^*_B} y^p$ zusammengefasst und wir suchen eine in $\tilde{y}$ verletzte Überdeckungsungleichung für den Schnitt $(S,\bar{S})$. Weil $\tilde{y}$ als eine Konvexkombination der Punkte y^p mit $p \in B$ die Schnittungleichung (2.19) verletzt, ist die Wahrscheinlichkeit, eine in diesem Punkt verletzte Überdeckungsungleichung zu finden, vergleichsweise hoch. Außerdem hat $\tilde{y}$ wie alle Konvexkombinationen der Lösungen aus dem Bündel β die Eigenschaft, dass jede in diesem Punkt verletzte Überdeckungsungleichung auch in einigen im Bündel gespeicherten Lösungen verletzt wird.

Gesucht ist eine Überdeckung für $(S,\bar{S})$ mit einer in dem Punkt $\tilde{y}$ verletzten Überdeckungsungleichung, das heißt, eine Überdeckung $C \subseteq (S,\bar{S})$ mit $\sum_{(i,j)\in C}\tilde{y}_{ij} < 1$. Mit der Einführung binärer Variablen z_{ij} mit $z_{ij} = 0$ für $(i,j) \in C$ und $z_{ij} = 1$ für $(i,j) \in (S,\bar{S}) \setminus C$ kann die Bestimmung einer solchen Überdeckung C auf die Lösung der Optimierungsaufgabe

$$\begin{aligned} Z = \quad & \min \quad \sum_{(i,j)\in(S,\bar{S})} \tilde{y}_{ij}(1-z_{ij}) \\ & \text{u.d.N.} \sum_{(i,j)\in(S,\bar{S})} u_{ij}z_{ij} < r_{(S,\bar{S})} \\ & \qquad z_{ij} \in \{0,1\} \qquad \forall (i,j) \in (S,\bar{S}). \end{aligned} \tag{2.20}$$

zurückgeführt werden. Eine exakte Lösung dieser Aufgabe liefert entweder eine Überdeckung mit der am stärksten verletzten Überdeckungsungleichung oder (im Falle $Z \geq 1$) einen Beweis dafür, dass $\tilde{y}$ alle Überdeckungsungleichungen für den Schnitt $(S,\bar{S})$ erfüllt.

Die Aufgabe (2.20) hat die Form eines binären Rucksackproblems und ist im Allgemeinen $\mathcal{NP}$-hart. Zu ihrer approximativen Lösung wird in unserer Implementierung eine Heuristik eingesetzt, bei der man versucht, aus der Menge $C = (S,\bar{S})$ möglichst viele Kanten mit großen $\tilde{y}_{ij}$ auszuschließen, um eine Überdeckung mit einem möglichst kleinen Wert $\sum_{(i,j)\in C}\tilde{y}_{ij}$ zu erhalten. Die Heuristik läuft wie folgt ab[9]: Nach der Initialisierung $z_{ij} := 0 \ \forall (i,j) \in (S,\bar{S})$ werden die Kanten aus $(S,\bar{S})$ nach den Werten $\tilde{y}_{ij}$ fallend und in den Gruppen mit gleichen $\tilde{y}_{ij}$ nach Kapazitäten u_{ij} aufsteigend sortiert. Wir betrachten die Kanten in dieser Reihenfolge und setzen für eine Kante (p,q) $z_{pq} := 1$, wenn ihre Kapazität u_{pq} zuzüglich der Gesamtkapazität der Kanten mit den bereits auf eins festgelegten Variablen z_{ij} kleiner als $r_{(S,\bar{S})}$ ist. Am Ende des Verfahrens liefert $C := \{(i,j) \in (S,\bar{S}) : z_{ij} = 0\}$ eine minimale Überdeckung für den Schnitt $(S,\bar{S})$.

Die mit dieser Überdeckung assoziierte Ungleichung $\sum_{(i,j)\in C} y_{ij} \geq 1$ kann mit einer Liftingprozedur verschärft werden. (Selbst wenn diese Ungleichung in dem Punkt $\tilde{y}$ nicht verletzt ist, ist es in manchen Fällen möglich, durch das Lifting eine verletzte Ungleichung zu erhalten.)

Lifting

Das Lifting ist eine oft verwendete Technik zum Verschärfen gültiger Ungleichungen. Um die Grundidee zu erläutern, betrachten wir eine Menge $Y \subseteq \{0,1\}^n$ mit ihren Teilmengen $Y_0 = \{y \in Y : y_1 = 0\}$ und $Y_1 = \{y \in Y : y_1 = 1\}$. Sei

$$\sum_{i=2}^{n} \pi_i y_i \geq \pi_0 \tag{2.21}$$

eine gültige Ungleichung für Y_1. Das Liftingproblem ist, einen Liftingkoeffizienten γ_1 zu finden, sodass

$$\gamma_1 y_1 + \sum_{i=2}^{n} \pi_i y_i \geq \pi_0 + \gamma_1 \tag{2.22}$$

eine gültige Ungleichung für die gesamte Menge Y darstellt.

[9] Eine ähnliche Heuristik wurde in [Chouman et al., 2003] verwendet.

Satz 2.7 (Down-Lifting) *Sei $Y_0 \neq \emptyset$ und (2.21) eine gültige Ungleichung für Y_1. Für alle $\gamma_1 \leq \varepsilon_1 - \pi_0$ mit*

$$\varepsilon_1 = \min\{\textstyle\sum_{i=2}^{n} \pi_i y_i : y \in Y_0\}$$

ist (2.22) eine gültige Ungleichung für Y. Gilt $\gamma_1 = \varepsilon_1 - \pi_0$ und beschreibt (2.21) eine k-dimensionale Randfläche von conv(Y_1), so definiert (2.22) eine $(k+1)$-dimensionale Randfläche der konvexen Hülle von Y.

Beweis: Wir beweisen nur die erste Aussage des Satzes und verweisen für einen vollständigen Beweis auf [Nemhauser and Wolsey, 1988].

Für $y_1 = 1$ sind die Ungleichungen (2.21) und (2.22) offenbar äquivalent. Daher bedeutet die vorausgesetzte Gültigkeit von (2.21) für die Menge Y_1 zugleich die Gültigkeit der Ungleichung (2.22) für alle $y \in Y$ mit $y_1 = 1$. Ferner folgt aus der Definition von ε_1 und γ_1 für alle $y \in Y$ mit $y_1 = 0$

$$\gamma_1 y_1 + \sum_{i=2}^{n} \pi_i y_i = \sum_{i=2}^{n} \pi_i y_i \geq \varepsilon_1 \geq \pi_0 + \gamma_1.$$

Damit ist die Gültigkeit der Ungleichung (2.22) für Y gezeigt. □

Nach der Bestimmung einer Überdeckungsungleichung $\sum_{(i,j)\in C} y_{ij} \geq 1$ für einen Schnitt $(S,\bar{S})$ wird in unserem Algorithmus das Down-Lifting aller Variablen y_{ij} mit $(i,j) \in (S,\bar{S}) \setminus C$ durchgeführt. Wir betrachten die Berechnung eines Liftingkoeffizienten γ_{pq} für eine Variable y_{pq}. Bezeichne

$$L = \{(i,j) \in (S,\bar{S}) \setminus C : y_{ij} \text{ wurde bereits geliftet}\}$$

die Menge der in vorherigen Schritten gelifteten Variablen und sei

$$\textstyle\sum_{(i,j)\in L} \gamma_{ij} y_{ij} + \sum_{(i,j)\in C} y_{ij} \geq 1 + \sum_{(i,j)\in L} \gamma_{ij} \tag{2.23}$$

die dadurch gewonnene gültige Ungleichung für $Y_{(S,\bar{S})}$. Der Liftingkoeffizient für y_{pq} berechnet sich als $\gamma_{pq} = \varepsilon_{pq} - 1 - \sum_{(i,j)\in L} \gamma_{ij}$ aus der Lösung der Optimierungsaufgabe

$$\begin{aligned} \varepsilon_{pq} = \min \quad & \sum_{(i,j)\in L} \gamma_{ij} y_{ij} + \sum_{(i,j)\in C} y_{ij} \\ \text{u.d.N.} \quad & \sum_{(i,j)\in (S,\bar{S})\setminus\{(p,q)\}} u_{ij} y_{ij} \geq r_{(S,\bar{S})} \\ & y_{ij} \in \{0,1\} \qquad \forall (i,j) \in (S,\bar{S}). \end{aligned} \tag{2.24}$$

Hat diese Aufgabe keine zulässige Lösung (d.h. reicht die Kapazität von $(S,\bar{S})$ ohne die Kante (p,q) nicht aus, um den Transportbedarf zu erfüllen), so gilt in jeder zulässigen Lösung des Netzwerkentwurfproblems $y_{pq} = 1$. Das Fixieren der Variablen y_{pq} auf diesen Wert macht ihr Lifting überflüssig.

Weil die Ungleichung (2.23) ebenso wie die ursprüngliche Überdeckungsungleichung

$$\sum_{(i,j)\in C} y_{ij} \geq 1$$

für jedes $y \in Y_{(S,\bar{S})}$ und somit für alle zulässigen Lösungen von (2.24) gültig ist, gilt $\varepsilon_{pq} \geq 1 + \sum_{(i,j)\in L} \gamma_{ij}$. Demzufolge ist der Liftingkoeffizient $\gamma_{pq} = \varepsilon_{pq} - 1 - \sum_{(i,j)\in L} \gamma_{ij}$ nicht negativ.

Das Gleiche gilt natürlich für alle Koeffizienten γ_{ij}. Damit wird klar, dass $\sum_{(i,j)\in L}\gamma_{ij}y_{ij}$ auf der linken Seite der Ungleichung (2.23) für alle y kleiner oder gleich dem Term $\sum_{(i,j)\in L}\gamma_{ij}$ auf ihrer rechten Seite ist und dass diese Art des Liftings in der Tat zum Verschärfen der Überdeckungsungleichung beiträgt. Außerdem sind aufgrund der Ganzzahligkeit von Koeffizienten der Überdeckungsungleichung offenbar auch alle Liftingkoeffizienten ganzzahlig.

Das bei der Berechnung des Koeffizienten γ_{pq} zu lösende Liftingproblem (2.24) hat eine ähnliche Struktur wie das binäre Rucksackproblem und kann mit einer Modifikation des bekannten Verfahrens der dynamischen Programmierung für das Rucksackproblem[10] in der Zeit $O(\varepsilon_{pq}|(S,\bar{S})|)$ gelöst werden. Bezeichne π_{ij} den Koeffizienten der Variablen y_{ij} in der Zielfunktion von (2.24) und sei $a_1,\ldots,a_m$ eine Nummerierung der Kanten von $(S,\bar{S})$ mit $\pi_{ij}\neq 0$. Wir definieren $F_j(\xi)$ als den größten Transportbedarf von S nach $\bar{S}$, der mit Kanten eines Gesamtwerts ξ erfüllt werden kann, wenn man die Kanten $a_1,\ldots,a_j$ benutzen darf,

$$F_j(\xi)=\max\left\{\sum_{i=1}^{j}u_{a_i}y_{a_i}:\ \sum_{i=1}^{j}\pi_{a_i}y_{a_i}=\xi,\ y\in\{0,1\}^m\right\}.$$

Mit der Initialisierung

$$F_0(\xi)=\begin{cases}0 & \text{für } \xi=0\\ -\infty & \text{für } \xi>0\end{cases}$$

können die Werte $F_j(\xi)$ mit $j=1,\ldots,m$ und $\xi=0,\ldots,\sum_{i=1}^{m}\pi_{a_i}$ durch sukzessive Anwendung der Bellmanschen Optimalitätsgleichung

$$F_j(\xi)=\begin{cases}F_{j-1}(\xi) & \text{für } \xi<\pi_{a_j}\\ \max\{F_{j-1}(\xi),F_{j-1}(\xi-\pi_{a_j})+u_{a_j}\} & \text{für } \xi\geq\pi_{a_j}\end{cases}$$

nacheinander berechnet werden. Das kleinste ξ mit

$$F_m(\xi)\geq r_{(S,\bar{S})}-\sum_{(i,j)\in(S,\bar{S})\setminus\{(p,q)\}:\pi_{ij}=0}u_{ij}$$

ergibt den Wert der optimalen Lösung ε_{pq} der Aufgabe (2.24).

Im Allgemeinen hängen die Werte der Liftingkoeffizienten γ_{ij} von der Liftingsequenz (d.h. der Reihenfolge, in der die Variablen geliftet werden) ab. Der Koeffizient γ_{ij} ist maximal, wenn man das Lifting mit y_{ij} beginnt, und wird immer kleiner, je mehr Variablen vor y_{ij} geliftet werden.[11] Um eine gültige Ungleichung mit einer möglichst großen Verletzung in dem Punkt $\tilde{y}$ (Seite 63) zu erhalten, werden in unserer Implementierung Variablen aus $(S,\bar{S})\setminus C$ in der Reihenfolge aufsteigender Werte $\tilde{y}_{ij}$ und bei gleichen $\tilde{y}_{ij}$ in der Reihenfolge aufsteigender Kapazitäten u_{ij} geliftet.

2.11.3 Lokale Schnitte

Neben den bekannten gültigen Ungleichungen für das Netzwerkentwurfproblem werden in unserem Branch-and-Cut-Verfahren auch Ungleichungen generiert, die in einzelnen Ästen des Suchbaums zulässige Lösungen mit zu hohen Kosten abschneiden. Hierzu wird genauso wie in der im Abschnitt 2.9 beschriebenen Methode der Variablenfixierung eine Abschätzung

[10] Siehe die Beschreibung des pseudopolynomiellen exakten Algorithmus für das Rucksackproblem in [Kellerer et al., 2004], Kapitel 6.

[11] Für einen Beweis dieser Aussage siehe [Nemhauser and Wolsey, 1988, Seite 264].

der unteren Schranke beim Fixieren einiger y-Variablen entgegen ihren Werten in der aktuellen Lagrange-Relaxation unternommen.

Betrachte eine mit dem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ assoziierte Rucksack-Relaxation ohne Kardinalitätsbedingungen. Nach der Berechnung der reduzierten Kosten $\hat{g}_{ij}$ kann diese Lagrange-Relaxation als eine Optimierungsaufgabe in y-Variablen

$$z_R(\omega) = \min_{y \in \{0,1\}^{|\mathscr{A}_*|}} \sum_{(i,j) \in \mathscr{A}_*} \hat{g}_{ij} y_{ij} + \sum_{(i,j) \in A_1} \hat{g}_{ij} - \omega^T b \tag{2.25}$$

aufgefasst werden. Bezeichne $\overline{z}_{CNDP}$ die Kosten der besten bekannten zulässigen Lösung des Netzwerkentwurfproblems. Wir zeigen:

Behauptung 2.8 *a) Sei $T_+ \subseteq \{(i,j) \in \mathscr{A}_* : \hat{g}_{ij} > 0\}$ eine Teilmenge der unfixierten Kanten mit positiven reduzierten Kosten, für die gilt:*

$$z_R(\omega) + \hat{g}_{pq} + \hat{g}_{rs} \geq \overline{z}_{CNDP} \quad \forall\, (p,q),(r,s) \in T_+ \text{ mit } (p,q) \neq (r,s). \tag{2.26}$$

Alle zulässigen Lösungen des mit dem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ assoziierten Teilproblems, in denen mehr als eine Designvariable aus T_+ den Wert eins annimmt, haben mindestens die Kosten $\overline{z}_{CNDP}$.

b) Sei $T_- \subseteq \{(i,j) \in \mathscr{A}_* : \hat{g}_{ij} < 0\}$ eine Teilmenge der unfixierten Kanten mit negativen reduzierten Kosten mit der Eigenschaft

$$z_R(\omega) - \hat{g}_{pq} - \hat{g}_{rs} \geq \overline{z}_{CNDP} \quad \forall\, (p,q),(r,s) \in T_- \text{ mit } (p,q) \neq (r,s), \tag{2.27}$$

so hat jede zulässige Lösung des mit dem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ assoziierten Teilproblems, in der mehr als eine Designvariable aus T_- den Wert Null annimmt, mindestens die Kosten $\overline{z}_{CNDP}$.

Beweis: a) Seien (p,q) und (r,s) zwei beliebige Kanten aus T_+ und bezeichne y' die optimale Lösung der Lagrange-Relaxation (2.25) und y'' die kostenminimale Lösung dieser Aufgabe mit $y_{pq} = y_{rs} = 1$.

In y' werden die Werte aller unfixierten y-Variablen abhängig von den Vorzeichen ihrer reduzierten Kosten festgelegt:

$$y'_{ij} = \begin{cases} 1, & \text{falls } \hat{g}_{ij} < 0 \\ 0 & \text{sonst} \end{cases} \qquad \forall (i,j) \in \mathscr{A}_*.$$

Für die Kanten (p,q) und (r,s) gilt deshalb $y'_{pq} = y'_{rs} = 0$. Die Lösung y'' erhält man, indem man für alle Kanten $(i,j) \in \mathscr{A}_* \setminus \{(p,q),(r,s)\}$ $y''_{ij} = y'_{ij}$ und $y''_{pq} = y''_{rs} = 1$ setzt. Folglich lassen sich die Kosten von y'' als

$$z(\omega, y'') = z(\omega, y') + \hat{g}_{pq}(y''_{pq} - y'_{pq}) + \hat{g}_{rs}(y''_{rs} - y'_{rs}) = z_R(\omega) + \hat{g}_{pq} + \hat{g}_{rs}$$

berechnen. Weil $z(\omega, y'')$ eine untere Schranke für die Kosten aller zulässigen Lösungen des Teilproblems $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ mit $y_{pq} = y_{rs} = 1$ darstellt, folgt aus

$$z_R(\omega) + \hat{g}_{pq} + \hat{g}_{rs} \geq \overline{z}_{CNDP} \quad \forall\, (p,q),(r,s) \in T_+ \text{ mit } (p,q) \neq (r,s),$$

dass jede zulässige Lösung des Teilproblems $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ mit $y_{pq} = y_{rs} = 1$ mindestens die Kosten $\overline{z}_{CNDP}$ hat. Weil dies ebenso für jedes andere Kantenpaar aus T_+ zutrifft, kann es an dem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ keine verbessernden Lösungen mit mehr als einer auf eins gesetzten Variablen y_{pq}, $(p,q) \in T_+$ geben.

b) Der zweite Teil der Behauptung folgt analog. □

Folgerung 2.9 *Sind T_+ und T_- Kantenmengen aus der Behauptung 2.8, so kann der zulässige Bereich des Teilproblems $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ durch Hinzufügen der Ungleichung*

$$\sum_{(i,j)\in T_+} y_{ij} \leq 1 \tag{2.28}$$

bzw. der Ungleichung

$$\sum_{(i,j)\in T_-} y_{ij} \geq |T_-| - 1 \tag{2.29}$$

eingeschränkt werden, ohne hierbei verbessernde Lösungen zu verlieren.

Bezeichne $D = \overline{z}_{CNDP} - z_R(\omega)$ die Differenz zwischen der besten bekannten oberen Schranke und dem Wert der aktuellen Rucksack-Relaxation. Wir gehen davon aus, dass $D > 0$ ist (andernfalls kann der aktuelle Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ sicher keine Lösungen mit kleineren Kosten als $\overline{z}_{CNDP}$ enthalten und wird ohne weitere Untersuchung aus dem Suchbaum entfernt). Ferner wird angenommen, dass alle Designvariablen y_{ij} mit $|\hat{g}_{ij}| \geq D$ bereits durch das im Abschnitt 2.9 geschilderte Verfahren auf die richtigen Werte fixiert worden sind.

Um Ungleichungen zu generieren, die möglichst viele uninteressante Lösungen abschneiden, wird nun nach größtmöglichen Mengen T_+ und T_- gesucht. Das bei der Wahl der Menge T_+ verwendete Verfahren ist im Algorithmus 10 dargestellt. Man startet mit der Menge $T_+ := \{(i,j) \in \mathscr{A}_* : \hat{g}_{ij} \geq \frac{1}{2}D\}$ und bestimmt die Menge der Kanten, deren Hinzufügen zu T_+ keine Verletzung der Bedingung (2.26) bewirkt. Danach wählt man per Zufall eine dieser Kanten aus und fügt sie der Menge T_+ hinzu. Die Wahl der Menge T_- erfolgt analog: Man

Algorithmus 10 Lokale Schnitte: Zusammensetzen der Menge T_+

```
1: setze T_+ := {(i,j) ∈ A_* : ĝ_ij ≥ ½D}
2: if T_+ ≠ ∅ then
3:    setze (p,q) := argmin_{(i,j)∈T_+} ĝ_ij
4:    setze W := {(i,j) ∈ A_* \ T_+ : ĝ_ij + ĝ_pq ≥ D}
5:    if W ≠ ∅ then
6:       wähle (r,s) ∈_R W                    {zufällige Wahl}
7:       setze T_+ := T_+ ∪ {(r,s)}
```

beginnt mit der Menge $T_- := \{(i,j) \in \mathscr{A}_* : -\hat{g}_{ij} \geq \frac{1}{2}D\}$ und sucht eine Kante, mit der diese Menge ergänzt werden kann, ohne eine Verletzung der Bedingung (2.27) hervorzurufen. Enthält T_+ bzw. T_- am Ende dieser Prozedur mehr als eine Kante, nutzen wir diese Menge, um eine für *verbessernde Lösungen* gültige Ungleichung (2.28) bzw. (2.29) zu generieren.

Um möglichst viele Lösungen mit hohen Kosten abzuschneiden, wird dieses Verfahren in jeder Iteration des Bundle-Algorithmus ausgeführt. Die hierbei gewonnenen Ungleichungen werden gespeichert, um später zum Verschärfen der unteren Schranke oder zum Variablenfixieren benutzt zu werden.

2.11.4 Einfluss der Ungleichungen auf die Berechnung unterer Schranken

Während der Lösung des Lagrange-Multiplikator-Problems durch das Bundle-Verfahren wird in regelmäßigen Abständen die Suche nach gültigen Ungleichungen gestartet, die in den Lösungen einzelner Lagrange-Teilprobleme verletzt werden. Sei Y_β die Menge der im Bündel β eingetragenen Lösungen y^p mit einem von Null verschiedenen Gewicht $\theta_p^* > 0$ in der optimalen Lösung von $(\Delta_{\beta t})$. Wir prüfen, ob die Punkte aus Y_β alle bekannten für verbessernde Lösungen gültigen Ungleichungen nach Abschnitt 2.11.3 erfüllen, und versuchen aus bekannten Schnittungleichungen geliftete Überdeckungsungleichungen herzuleiten, die in einigen Punkten aus Y_β verletzt werden.

Identifiziert man solche in den Punkten aus Y_β verletzten gültigen Ungleichungen, die noch keine Lagrange-Multiplikatoren haben, wird der Zielfunktion der Lagrange-Relaxation für jede solche Ungleichung $\sum_{(i,j)\in\mathscr{A}} a_{ij}^k y_{ij} \geq q_k$ ein Term $\upsilon_k(q_k - \sum_{(i,j)\in\mathscr{A}} a_{ij}^k y_{ij})$ mit einem neuen Lagrange-Multiplikator $\upsilon_k \geq 0$ hinzugefügt. Das Lagrange-Multiplikator-Problem bekommt neue Variablen und damit vielleicht auch einen höheren Maximalwert z_{LM}. Um die im Bündel gespeicherte Information der neuen Lagrange-Funktion anzupassen, wird für jeden Eintrag $\langle(\omega^p,\upsilon^p),z^p,(s^p,t^p),y^p\rangle$ und jeden neuen Lagrange-Multiplikator υ_k die mit diesem Lagrange-Multiplikator assoziierte Komponente des Subgradienten $t_k^p = q_k - \sum_{(i,j)\in\mathscr{A}} a_{ij}^k y_{ij}^p$ bestimmt. Der Eintrag $\langle(\omega^p,\upsilon^p),z^p,(s^p,t^p),y^p\rangle$ wird zu $\langle(\omega^p,\upsilon^p,0,\ldots,0),z^p,(s^p,t^p,t_{k_1}^p,\ldots,t_{k_m}^p),y^p\rangle$ und das Bundle-Verfahren wird fortgesetzt.

Nach Einbeziehen zusätzlicher Ungleichungen in die Zielfunktion der Rucksack-Relaxation lässt sich diese als

$$\begin{aligned} z_R(\omega,\upsilon) \;= \min\; & \sum_{(i,j)\in\mathscr{A}} \left[\sum_{k\in\mathscr{C}}(c_{ij}^k + \omega_i^k - \omega_j^k)x_{ij}^k + \left(f_{ij} - \sum_{k\in K}\upsilon_k a_{ij}^k\right) y_{ij}\right] \\ & - \sum_{k\in\mathscr{C}}\sum_{i\in\mathscr{N}} \omega_i^k b_i^k + \sum_{k\in K}\upsilon_k q_k \end{aligned}$$

unter den Nebenbedingungen des Netzwerkentwurfproblems formulieren. Die reduzierten Kosten auf der Kante (i,j) berechnen sich als $\hat{g}_{ij} = g_{ij}(\omega) + f_{ij} - \sum_{k\in K}\upsilon_k a_{ij}^k$. Hierbei bezeichnet $g_{ij}(\omega)$ den durch die Lösung des kontinuierlichen Rucksackproblems (R_{ij}) errechneten Wert der kostenminimalen Belegung der x-Variablen für $y_{ij} = 1$.

Zum Verschärfen der unteren Schranke wird in unserem Algorithmus in jedem Knoten $(\mathscr{A}_0,\mathscr{A}_1,\mathscr{A}_*)$ nach Beendigung des Bundle-Verfahrens eine Anpassung der mit zusätzlichen Ungleichungen assoziierten Lagrange-Multiplikatoren vorgenommen. Seien $(\bar{\omega},\bar{\upsilon})$ die besten im Laufe des Bundle-Verfahrens gefundenen Werte der Lagrange-Multiplikatoren und $(\bar{x},\bar{y})$ die kostenminimale Lösung des entsprechenden Lagrange-Teilproblems. Wir beziehen die Überdeckungsungleichungen und die lokal gültigen Ungleichungen aus dem Abschnitt 2.11.3, die in $\bar{y}$ verletzt werden und noch keine Lagrange-Multiplikatoren haben, mit einem neuen mit Null initialisierten Lagrange-Multiplikator in die Zielfunktion der Lagrange-Relaxation $\mathrm{LR}(\bar{\omega},\bar{\upsilon})$ ein. Wir versuchen anschließend durch Variieren einzelner mit zusätzlichen Ungleichungen assoziierter Lagrange-Multiplikatoren υ_k eine Erhöhung der unteren Schranke zu erreichen, ohne dabei den Bereich im Raum (ω,υ) zu verlassen, in dem $(\bar{x},\bar{y})$ eine optimale Lösung der Lagrange-Relaxation $\mathrm{LR}(\omega,\upsilon)$ darstellt [12].

[12] Eine ähnliche Methode zum Verschärfen der unteren Schranke wurde von Balas und Christofides in [Balas and Christofides, 1981] verwendet.

Hierfür wird in jedem Schritt ein Lagrange-Multiplikator υ_k mit einer von Null verschiedenen Komponente des Subgradienten $t_k = q_k - \sum_{(i,j)\in\mathscr{A}} a_{ij}^k \bar{y}_{ij}$ gewählt. Eine Modifikation dieses Multiplikators durch $\upsilon_k := \upsilon_k + \delta_k$ erhöht die Kosten von $(\bar{x},\bar{y})$ um $\delta_k t_k$ und verändert die reduzierten Kosten jeder Kante $(i,j) \in \mathscr{A}$ entsprechend $\hat{g}_{ij} := \hat{g}_{ij} - \delta_k a_{ij}^k$. Wählt man δ_k so, dass

(a) $\hat{g}_{ij} - \delta_k a_{ij}^k \geq 0$ für alle $(i,j) \in \mathscr{A}_*$ mit $\bar{y}_{ij} = 0$ und

(b) $\hat{g}_{ij} - \delta_k a_{ij}^k \leq 0$ für alle $(i,j) \in \mathscr{A}_*$ mit $\bar{y}_{ij} = 1$ gilt,

so bleibt $(\bar{x},\bar{y})$ bei dieser Änderung des Lagrange-Multiplikators υ_k optimal. Die Kosten dieser Lösung stellen in der neuen Lagrange-Relaxation eine untere Schranke für die Kosten aller für den Knoten $(\mathscr{A}_0,\mathscr{A}_1,\mathscr{A}_*)$ zulässigen Lösungen des Netzwerkentwurfs dar. Um den Anstieg der unteren Schranke $\delta_k t_k$ zu maximieren, wird in dem Fall $t_k > 0$ der größte Wert $\delta_k \in [0,\infty[$ gewählt, der die Bedingungen (a) und (b) erfüllt, und in dem Fall $t_k < 0$ der kleinste diese Bedingungen erfüllende Wert $\delta_k \in [-\upsilon_k, 0]$. Wir setzen $\upsilon_k := \upsilon_k + \delta_k$, bestimmen die neuen reduzierten Kosten und fahren mit dem nächsten Lagrange-Multiplikator fort, bis alle mit zusätzlichen Ungleichungen assoziierten Lagrange-Multiplikatoren υ_k mit von Null verschiedenen Komponenten des Subgradienten behandelt worden sind.

Bei einer Verzweigung des Branch-and-Bound-Baums werden die Werte der Lagrange-Multiplikatoren (ω,υ), die im Vaterknoten für den größten Wert der unteren Schranke $z_R(\omega,\upsilon)$ sorgten, an die Nachfolgerknoten weitergegeben, um später zum Initialisieren des Bundle-Verfahrens benutzt zu werden. Die zusätzlichen Ungleichungen, deren Lagrange-Multiplikatoren υ_k den Wert Null haben, werden hierbei aus der Zielfunktion der Lagrange-Relaxation entfernt, um den Aufwand des Bundle-Algorithmus in den Nachfolgerknoten zu reduzieren.

2.12 Relax-and-cut-Algorithmus

Eine zurzeit sehr oft verwendete Lösungsmethode für (gemischt) ganzzahlige lineare Optimierungsprobleme kombiniert ein auf LP-Relaxation basierendes Branch-and-Bound-Verfahren mit einem Schnittebenenalgorithmus. Bei diesem so genannten Branch-and-cut-Ansatz wird in jedem Knoten des Suchbaums vor einer Verzweigung zunächst eine Verschärfung der aktuellen LP-Relaxation durch Hinzufügen neuer Nebenbedingungen vorgenommen. In diesem Abschnitt beschreiben wir den Branch-and-cut-Algorithmus, der auf Lagrange-Relaxationen als untere Schranken basiert und in der Literatur als Relax-and-cut-Algorithmus bezeichnet wird.

2.12.1 Schnittebenenalgorithmus

Das ganzzahlige lineare Optimierungsproblem

$$(G) \quad z_G = \min\{c^T x : x \in S\} \text{ mit } S = M \cap Z^n,\ M = \{x \in R_+^n : Ax \geq b\}$$

mit $A \in Z^{m\times n}$ und $b \in Z^m$ kann auf die Lösung eines, auf der konvexen Hülle von S definierten, linearen Programms

$$(G') \qquad \min\{c^T x : x \in conv(S)\}$$

zurückgeführt werden. Weil $conv(S)$ ein S einschließendes konvexes Polyeder ist, dessen Ecken Elemente aus S und zulässige Lösungen von (G) darstellen, ist eine optimale Basislösung von (G') auch für (G) optimal.

Demzufolge könnte eine optimale Lösung des ganzzahligen Optimierungsproblems (G) zum Beispiel mit der Simplexmethode berechnet werden, wenn eine Beschreibung von $conv(S)$ durch Ungleichungen bekannt wäre. Dieses ist jedoch im Allgemeinen nicht der Fall. Selbst wenn man die Struktur der benötigten Ungleichungen kennt, ist es meist ein aussichtsloses Unterfangen, eine komplette polyedrische Beschreibung von $conv(S)$ finden zu wollen, weil eine solche Beschreibung in der Regel eine extrem große (mit n exponentiell wachsende) Anzahl Ungleichungen hat.

Statt unmittelbar eine Beschreibung von $conv(S)$ zu suchen, wird bei der Schnittebenenmethode eine Folge konvexer Polyeder $Q_1, Q_2, \ldots$ mit $M = Q_1 \supset Q_2 \supset \ldots \supseteq conv(S)$ konstruiert. In jeder Iteration wird ein Stück des bisherigen zulässigen Bereichs Q_i durch Hinzufügen einer oder mehrerer für $conv(S)$ gültiger Ungleichungen weggeschnitten, bis eine optimale Lösung des vorliegenden linearen Optimierungsproblems

$$(L_i) \qquad \min\{c^T x : x \in Q_i\}$$

ganzzahlig und damit auch für (G) optimal ist.

Der generelle Ablauf eines Schnittebenenverfahrens ist im Algorithmus 11 dargestellt (siehe auch [Marchand et al., 2002]). Man startet mit der Lösung der LP-Relaxation von (G)

$$(L_1) \qquad \min\{c^T x : x \in Q_1 = M\}.$$

Wenn (L_1) keine optimale Lösung besitzt, weil der zulässige Bereich M leer ist oder weil die Zielfunktion $c^T x$ auf M nach unten unbeschränkt ist, hat auch die ganzzahlige Aufgabe (G) keine optimale Lösung [Klose, 2002], das Verfahren terminiert. Andernfalls bezeichne x^1 die optimale Lösung von (L_1).

Algorithmus 11 Schnittebenenverfahren

Gegeben: eine Menge für $conv(S)$ gültiger Ungleichungen
$\alpha^{kT} x \geq \beta^k$, $k \in \mathcal{K}$

1: initialisiere $Q_1 := \{x \in R^n_+ : Ax \geq b\}$
2: bestimme eine optimale Lösung x^1 von $\min\{c^T x : x \in Q_1\}$
3: **if** $Q_1 \neq \emptyset$ **and** $-\infty < \min\{c^T x : x \in Q_1\}$ **then**
4: $\quad i := 1$
5: $\quad$ **while** $Q_i \neq \emptyset$ **and** $x^i \notin Z^n$ **and** $\exists k \in \mathcal{K}$ mit $\alpha^{kT} x^i < \beta^k$ **do**
6: $\quad\quad$ wähle $\mathcal{K}^i \subseteq \mathcal{K}$ mit $\mathcal{K}^i \neq \emptyset$ und $\alpha^{kT} x^i < \beta^k$ für alle $k \in \mathcal{K}^i$
7: $\quad\quad$ $Q_{i+1} := Q_i \cap \{x \in R^m_+ : \alpha^{kT} x \geq \beta^k$ für alle $k \in \mathcal{K}^i\}$
8: $\quad\quad$ bestimme eine optimale Lösung x^{i+1} von $\min\{c^T x : x \in Q_{i+1}\}$
9: $\quad\quad$ $i := i + 1$

Solange die optimale Lösung x^i der aktuellen linearen Aufgabe (L_i) nichtganzzahlig ist, sucht man in jeder Iteration aus einer gegebenen Menge

$$\alpha^{kT} x \geq \beta^k, \quad k \in \mathcal{K}$$

gültiger Ungleichungen für $conv(S)$ eine oder mehrere aus, die in x^i verletzt sind (Zeile 6). Diese Ungleichungen werden der Beschreibung des zulässigen Bereiches hinzugefügt (Zeile 7), sodass der nichtganzzahlige Punkt x^i aus dem neuen zulässigen Bereich Q_{i+1} ausgeschlossen wird. Danach bestimmt man eine kostenminimale Lösung x^{i+1} der neuen linearen Optimierungsaufgabe (L_{i+1}) und geht zur nächsten Iteration über. Findet man keine Ungleichung, die den aktuellen nichtganzzahligen Punkt x^i von $conv(S)$ trennt, terminiert das Verfahren. Der Wert $c^T x^i$ liefert eine untere Schranke für die Kosten der optimalen Lösung von (G). Wenn die optimale Lösung der aktuellen linearen Aufgabe (L_i) ganzzahlig und somit auch für (G) optimal ist, terminiert das Verfahren. Hat (L_i) keine zulässige Lösung, bricht der Algorithmus ab, weil in diesem Fall auch die ganzzahlige Aufgabe (G) keine zulässige Lösung besitzt.

2.12.2 Schnittebenen und Lagrange-Relaxation

In dem oben skizzierten Schnittebenenverfahren wird die LP-Relaxation eines ganzzahligen Optimierungsproblems (G) durch Hinzufügen zusätzlicher Nebenbedingungen verschärft, sodass man eine bessere untere Schranke für den optimalen Zielfunktionswert von (G) oder sogar die optimale Lösung dieses Problems erhält. In dieser Arbeit wird eine ähnliche Technik zum Verschärfen der Lagrange-Relaxation angewendet.

Betrachtet wird ein ganzzahliges Optimierungsproblem

$$
\begin{aligned}
(P) \qquad z_P = \ & \min \quad c^T x \\
& \text{u.d.N. } Ax \geq b && (2.30) \\
& \qquad x \in X = \{x \in Z^n : Dx \geq q\}. && (2.31)
\end{aligned}
$$

Wir nehmen an, dass (2.31) eher einfache Nebenbedingungen darstellt, während die Einschränkung (2.30) die Aufgabe sehr kompliziert, sodass die Lagrange-Relaxation von (P) bezüglich der Nebenbedingungen (2.30)

$$z_{LR}(\omega) = \min\{c^T x + \omega^T (b - Ax) : x \in X\} \tag{2.32}$$

für jedes $\omega \geq 0$ eine leicht zu lösende Aufgabe darstellt. Weiter sei für den zulässigen Bereich von (P) eine Menge gültiger Ungleichungen

$$\alpha^{kT} x \geq \beta^k, \quad k \in \mathscr{K}$$

gegeben. Diese für die Beschreibung von (P) redundanten Ungleichungen können benutzt werden, um die Dualitätslücke zwischen z_P und dem optimalen Zielfunktionswert z_{LM} des Lagrange-Multiplikator-Problems

$$z_{LM} = \max\{z_{LR}(\omega) : \omega \geq 0\} \tag{2.33}$$

zu verkleinern.

Mit diesem Ziel werden während der Lösung des Lagrange-Multiplikator-Problems durch ein Subgradienten- oder Bundle-Verfahren die Ungleichungen $\alpha^{kT} x \geq \beta^k$, $k \in \mathscr{K}$ identifiziert, die in den Lösungen einzelner Lagrange-Teilprobleme verletzt werden. Solche Ungleichungen werden mit einem neuen Multiplikator v_k in die Zielfunktion der Lagrange-Relaxation

aufgenommen, sodass ihr Verletzen in den späteren Iterationen mit zusätzlichen Kosten $v_k(\beta^k - \alpha^{kT}x)$ bestraft und ihr Einhalten belohnt wird. Das Lagrange-Multiplikator-Problem (2.33) bekommt neue Variablen $v_k \geq 0$ und damit vielleicht auch einen neuen (größeren) Maximalwert z_{LM}.

Es stellt sich nun die Frage, unter welchen Bedingungen das Einbeziehen einer neuen Ungleichung in die Lagrange-Relaxation eine Verschärfung der Schranke z_{LM} bewirkt. Das Lagrange-Multiplikator-Problem (2.33) besitzt den gleichen optimalen Zielfunktionswert wie die Minimierungsaufgabe

$$(L) \qquad z_L = \min\{c^T x : Ax \geq b, x \in conv(X)\}.$$

Führt man eine weitere für den zulässigen Bereich von (P) gültige Ungleichung $\alpha^T x \geq \beta$ in das Modell ein, um sie anschließend in die Zielfunktion der Lagrange-Relaxation aufzunehmen, ändert sich auch die Aufgabe (L). Sie erhält eine neue Nebenbedingung $\alpha^T x \geq \beta$ und wird zu

$$(L') \qquad z'_L = \min\{c^T x : Ax \geq b, \alpha^T x \geq \beta, x \in conv(X)\}.$$

Die zulässigen Bereiche beider Optimierungsaufgaben (L) und (L') sind in der Abbildung 2.18 schraffiert dargestellt. Das Einbeziehen der Ungleichung $\alpha^T x \geq \beta$ in die Zielfunktion des Lagrange-Teilproblems (2.32) sorgt immer dann für ein größeres Optimum des Lagrange-Multiplikator-Problems, wenn die Aufgabe (L') einen größeren optimalen Zielfunktionswert als (L) besitzt. Dies ist genau dann der Fall, wenn alle optimalen Lösungen von (L) in dem Halbraum mit $\alpha^T x < \beta$ liegen und durch die Einschränkung $\alpha^T x \geq \beta$ aus dem zulässigen Bereich von (L') ausgeschlossen werden.

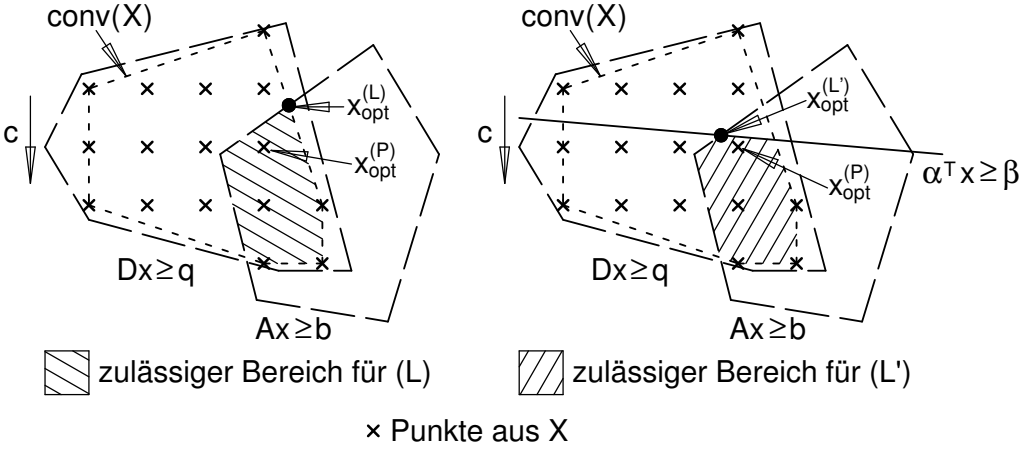


Abbildung 2.18: Änderung der zum Largange-Multiplikator-Problem dualen Aufgabe (L) beim Einbeziehen einer neuen Ungleichung $\alpha^T x \geq \beta$ in die Kostenfunktion der Lagrange-Relaxation

Sei $x^* \in X$ die optimale Lösung eines Lagrange-Teilproblems (2.32) und bezeichne Q den zulässigen Bereich von (L). Weil jeder Punkt aus $Q \cap X$ eine zulässige Lösung von (P) darstellt, kann eine für (P) gültige Ungleichung nur in solchen Punkten aus X verletzt sein, die sich außerhalb der Menge Q befinden. Das heißt wenn man eine für den zulässigen Bereich

von (P) gültige Ungleichung $\alpha^T x \geq \beta$ findet, die in x^* verletzt ist, gilt $x^* \notin Q$ und es ist nicht sicher, ob die Ungleichung $\alpha^T x \geq \beta$ zusammen mit x^* auch einen Teil des zulässigen Bereichs von (L) abschneidet (vgl. [Guignard, 1998, Ralphs and Galati, 2005]).

In der Abbildung 2.19 sind zwei Schnittebenen $\alpha^{kT} x \geq \beta^k$, $k = 1,2$ dargestellt, die den Punkt $x^* \in X$ von den zulässigen Lösungen des Optimierungsproblems (P) trennen. Während die erste Ungleichung $\alpha^{1T} x \geq \beta^1$ eine optimale Lösung von (L) abschneidet und so zu einer Verbesserung der Schranke z_{LM} beiträgt, verläuft der zweite Schnitt außerhalb des zulässigen Bereichs von (L), sodass die Ungleichung $\alpha^{2T} x \geq \beta^2$ keinen Einfluss auf die Lösung der Minimierungsaufgabe (L) und die Qualität der Schranke z_{LM} hat.

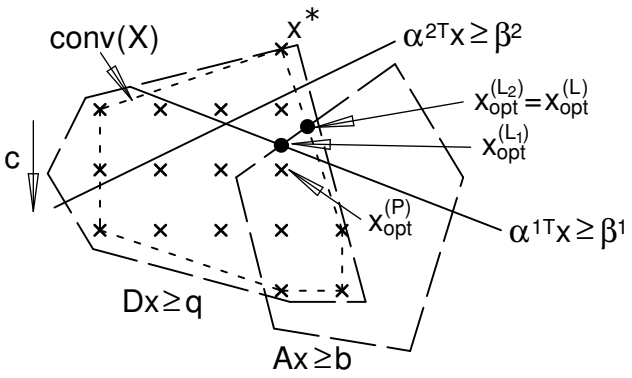


Abbildung 2.19: Eine Ungleichung, die die Lösung x^* eines Lagrange-Teilproblems von den zulässigen Lösungen der Aufgabe (P) trennt, muss nicht zwangsläufig einen Teil des zulässigen Bereichs von (L) abschneiden.

Wie man an diesem Beispiel sieht, führt das Einbeziehen einer in der Lösung des aktuellen Lagrange-Teilproblems verletzten gültigen Ungleichung in die Kosten der Lagrange-Relaxation nicht immer zu einer Verringerung der Dualitätslücke $z_P - z_{LM}$. Jedoch konnte diese Vorgehensweise bei einer Reihe ganzzahliger Optimierungsprobleme für eine deutliche Verbesserung der berechenbaren Schranke sorgen. In Kombination mit einem Branch-and-Bound-Verfahren wurde diese Technik bereits erfolgreich auf das Steiner-Baum-Problem [Lucena, 1992], das Cliquen-Problem mit Kantengewichten [Hunting et al., 2001], die Tourenplanung [Martinhon et al., 2000] und einige weitere schwierige ganzzahlige Optimierungsaufgaben angewandt.

2.12.3 Relax-and-cut-Verfahren

Dieser Abschnitt behandelt die in unserer Implementierung gewählten Strategien für den Aufbau und die Abarbeitung des Branch-and-Bound-Baums. Der Ablauf unseres Relax-and-cut-Verfahrens ist im Algorithmus 12 gezeigt. Es folgt dem allgemeinen Schema aus dem Abschnitt 2.3, ergänzt um eine Heuristik zum schnelleren Auffinden guter zulässiger Lösungen, heuristische Variablenfixierung, problemspezifische Verzweigungsstrategie und Kriterien zum Abschneiden uninteressanter Knoten.

Als *Suchstrategie* wurde in unserem Branch-and-Bound-Verfahren die Bestensuche implementiert. Als nächster zu untersuchender Knoten wird hierbei ein Knoten mit der kleinsten

Algorithmus 12 Relax-and-cut-Verfahren für das Netzwerkentwurfproblem

initialisiere $H := \{(\emptyset, \emptyset, \mathscr{A})\}$, $\overline{z}_{CNDP} := \infty$, $\underline{z}_{(\emptyset,\emptyset,\mathscr{A})} := -\infty$
while $(H \neq \emptyset)$ **and** $(\min_{(\mathscr{A}'_0,\mathscr{A}'_1,\mathscr{A}'_*)\in H} \underline{z}_{(\mathscr{A}'_0,\mathscr{A}'_1,\mathscr{A}'_*)} < \overline{z}_{CNDP})$ **do**
  wähle $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*) := \operatorname{argmin}_{(\mathscr{A}'_0,\mathscr{A}'_1,\mathscr{A}'_*)\in H} \underline{z}_{(\mathscr{A}'_0,\mathscr{A}'_1,\mathscr{A}'_*)}$ {Bestensuche}
  setze $H := H \setminus \{(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)\}$
  fixiere Variablen anhand zusätzlicher Ungleichungen
  if $(\mathscr{A}_* = \emptyset)$ **then**
    starte das exakte Verfahren für $PS(\mathscr{A}_1)$
  else
    starte die Heuristik (Algorithmus 7) für $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$
    if (die Heuristik findet keine Lösung) **then**
      starte das exakte Verfahren für $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$
      if (das Problem $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ hat keine zulässige Lösung) **or**
        $(\sum_{k\in\mathscr{C}} \sum_{(i,j)\in\mathscr{A}_1\cup\mathscr{A}_*} c^k_{ij} x^k_{ij} + \sum_{(i,j)\in\mathscr{A}_1} f_{ij} \geq \overline{z}_{CNDP})$ **then**
        continue {Cut off}
    berechne mit Bundle-Verfahren eine neue untere Schranke $\underline{z}_{(\mathscr{A}_0,\mathscr{A}_1,\mathscr{A}_*)}$
    [starte α- oder β-Fixierung]
    fixiere Variablen anhand zusätzlicher Ungleichungen
    if $(\underline{z}_{(\mathscr{A}_0,\mathscr{A}_1,\mathscr{A}_*)} \geq \overline{z}_{CNDP})$ **or** (eine zusätzliche Ungleichung wird
      bei jeder Belegung der unfixierten Variablen verletzt) **then**
      continue {Cut off}
    if (das Problem $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ wurde noch nicht exakt gelöst) **then**
      starte das exakte Verfahren für $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$
      if (das Problem $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ hat keine zulässige Lösung) **or**
        $(\sum_{k\in\mathscr{C}} \sum_{(i,j)\in\mathscr{A}_1\cup\mathscr{A}_*} c^k_{ij} x^k_{ij} + \sum_{(i,j)\in\mathscr{A}_1} f_{ij} \geq \overline{z}_{CNDP})$ **then**
        continue {Cut off}
    $E :=$ ⟨Menge der in der Lösung von $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ benutzten Kanten⟩
    wähle $(i,j) := \operatorname{argmin}_{(i',j')\in E\cap\mathscr{A}_*} |\hat{g}_{i'j'}|$
    generiere neue Knoten $W_0 := (\mathscr{A}_0 \cup \{(i,j)\}, \mathscr{A}_1, \mathscr{A}_* \setminus \{(i,j)\})$ und
    $W_1 := (\mathscr{A}_0, \mathscr{A}_1 \cup \{(i,j)\}, \mathscr{A}_* \setminus \{(i,j)\})$
    setze $H := H \cup \{W_0, W_1\}$ und $\underline{z}_{W_0} := \underline{z}_{W_1} := \underline{z}_{(\mathscr{A}_0,\mathscr{A}_1,\mathscr{A}_*)}$

unteren Schranke ausgewählt (Zeile 3). Die Bestensuche zeichnet ein im Vergleich zu anderen Suchstrategien kleinerer Rechenaufwand aus, weil man bei dieser Strategie nur Knoten untersucht, die nicht durch eine Verbesserung der oberen Schranke abgeschnitten werden können und untersucht werden müssen.

Die *zulässigen Lösungen* werden in unserem Algorithmus nicht nur an den Blättern, sondern an jedem Knoten des Suchbaums generiert. Die Untersuchung eines Knotens $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ mit einer nichtleeren Menge $\mathscr{A}_*$ fängt mit der Suche nach einer zulässigen Lösung an. Zu diesem Zweck lösen wir das primale Teilproblem $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ mit der im Abschnitt 2.7.3 beschriebenen Heuristik (Zeile 9). Liefert die Heuristik keine Lösung, lösen wir $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ nochmal mit einem exakten Verfahren (Zeile 11), um entweder eine zulässige Lösung zu finden, oder nachzuweisen, dass der Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ keine zulässigen Lösungen enthält und nicht weiter untersucht werden muss. Bei einer exakten Lösung von $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ erhält man neben einer zulässigen Lösung auch eine untere Schranke für den Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$. Ein Vergleich dieser Schranke mit den Kosten der besten bekannten Lösung (Zeile 12) trägt dazu bei, die für die Suche nach verbessernden Lösungen uninteressanten Knoten zu erkennen.

Nach der Überprüfung der Lösbarkeit des mit dem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ assoziierten Teilproblems starten wir das Bundle-Verfahren, um eine *untere Schranke* für die Kosten seiner optimalen Lösung zu finden (Zeile 16). Um eine kontinuierliche Verbesserung der unteren Schranke bei der Bewegung von der Wurzel des Suchbaums zu seinen Blättern zu sichern, wird das Bundle-Verfahren an dem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ mit den Werten der Lagrange-Multiplikatoren initialisiert, die in seinem Vaterknoten für den besten Wert der Lagrange-Schranke sorgten. Zum Verbessern der oberen Schranke wird in jeder M-ten Iteration des Bundle-Algorithmus die Lösung eines primalen Teilproblems $PS(\overline{\mathscr{A}})$ mit einer von den y-Lösungen der Lagrange-Relaxation abhängigen Menge $\overline{\mathscr{A}}$ angestoßen (s. Abschnitt 2.7.1). Außerdem werden im Laufe des Bundle-Verfahrens Algorithmen zur Variablenfixierung, Einschränkung des Kardinalitätsintervalls und zum Identifizieren gültiger Ungleichungen ausgeführt. Die in einzelnen Lagrange-Teilproblemen verletzten gültigen Ungleichungen werden mit neuen Lagrange-Multiplikatoren in die Zielfunktion der Lagrange-Relaxation einbezogen, in der Hoffnung, die untere Schranke zu verschärfen.

Liefert die Lösung der Rucksack-Relaxation in einer Iteration des Bundle-Algorithmus eine untere Schranke $\underline{z}_{(\mathscr{A}_0,\mathscr{A}_1,\mathscr{A}_*)}$, die die Kosten der besten bekannten zulässigen Lösung $\overline{z}_{CNDP}$ übersteigt, wird das Bundle-Verfahren abgebrochen und der Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ eliminiert, weil dieser Knoten keine zulässigen Lösungen mit kleineren Kosten als $\overline{z}_{CNDP}$ enthält. Tritt diese Abbruchbedingung nie ein, so hält das Bundle-Verfahren an, wenn

- alle Designvariablen fixiert worden sind,
- das Abbruchkriterium (2.14) erfüllt ist oder
- ein Iterationslimit L erreicht ist.

Nach der Berechnung der unteren Schranke wird in der heuristischen Version des Algorithmus eine der im Abschnitt 2.9 beschriebenen Heuristiken zur Variablenfixierung aufgerufen (Zeile 17). Anschließend wird überprüft, ob man aus den am Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ gespeicherten zusätzlichen Ungleichungen die richtigen Werte für einige y-Variablen folgern kann (Zeile 18) und ob es für jede dieser Ungleichungen eine Belegung der unfixierten y-Variablen gibt, die sie erfüllt (Zeile 19). Findet man eine für verbessernde Lösungen gültige Ungleichung, die am

Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ nicht erfüllt werden kann, bricht man die Untersuchung des Knotens ab.

Andernfalls lösen wir das primale Teilproblem $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ mit einem exakten Verfahren (sofern das nicht bereits bei der Suche nach einer zulässigen Lösung geschah) und teilen den zulässigen Bereich von $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ zur weiteren Untersuchung auf, indem wir eine *Verzweigungsvariable* y_{ij} mit $(i, j) \in \mathscr{A}_*$ wählen und zwei Nachfolgerknoten von $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ erzeugen, einen mit $y_{ij} = 1$ (die Kante (i, j) ist offen) und einen mit $y_{ij} = 0$ (die Kante (i, j) ist gesperrt). Die neuen Knoten werden der Menge H der noch nicht untersuchten Knoten hinzugefügt und die Bearbeitung des Knotens $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ wird abgeschlossen.

Bei der Auswahl einer Verzweigungsvariablen nutzen wir eine in [Holmberg and Yuan, 2000] vorgeschlagene Strategie, bei der man für die Verzweigung stets eine Kante $(i, j) \in \mathscr{A}_*$ wählt, die

(a) in der Lösung des primalen Teilproblems $PS(\mathscr{A}_1 \cup \mathscr{A}_*)$ zum Transport der Güter benutzt wird und

(b) unter allen die Voraussetzung (a) erfüllenden Kanten den kleinsten Betrag der reduzierten Kosten $|\hat{g}_{ij}|$ in einer Rucksack-Relaxation besitzt.

Die Verzweigung auf einer in der optimalen Lösung des primalen Teilproblems für den Transport der Güter benutzten Kante macht diese Lösung in einem der beiden Nachfolgerknoten unzulässig und ermöglicht so das Auffinden alternativer primaler Lösungen. Gehören alle in der Lösung des primalen Teilproblems benutzten Kanten der Menge $\mathscr{A}_1$ an, könnte keine Verzweigung in dem Knoten $(\mathscr{A}_0, \mathscr{A}_1, \mathscr{A}_*)$ zu einer besseren primalen Lösung führen. Die Wahl einer Kante mit betragsmäßig kleinen reduzierten Kosten basiert auf einer in vielen Branch-and-Bound-Algorithmen benutzten Idee, solche Variablen für die Verzweigung zu wählen, deren Werte in der Lösung der aktuellen Relaxation unsicher zu sein scheinen. In der Lösung der Rucksack-Relaxation (2.4.2) bestimmt das Vorzeichen der reduzierten Kosten $\hat{g}_{ij}$ den optimalen Wert einer Designvariablen y_{ij}. Ist $|\hat{g}_{ij}|$ groß, so ist der Wert von y_{ij} fast sicher und die Wahrscheinlichkeit, dass diese Variable durch eines der im Abschnitt 2.9 beschriebenen Verfahren fixiert wird, relativ hoch. Dagegen erscheint der Wert einer Designvariablen mit einem kleinen Betrag der reduzierten Kosten eher unsicher, weswegen die Untersuchung der beiden Fälle $y_{ij} = 0$ und $y_{ij} = 1$ für eine solche Variable y_{ij} angemessen erscheint.

Das Verfahren bricht ab, wenn der Branch-and-Bound-Baum vollständig abgearbeitet ist (d.h. wenn die Menge H leer ist), oder wenn man für jeden noch nicht untersuchten Knoten eine untere Schranke aus seinem Vorgänger kennt, die größer bzw. gleich den Kosten der besten bis dahin bekannten zulässigen Lösung $\overline{z}_{CNDP}$ ist.

2.13 Systemaufbau

Zur approximativen Lösung des Lagrange-Multiplikator-Problems wird in dieser Arbeit der im Abschnitt 2.6.2 beschriebene Bundle-Algorithmus von Antonio Frangioni eingesetzt. Um dieses Verfahren in unseren Relax-and-cut-Algorithmus zu integrieren, wurden Spezialisierungen der von Frangioni eingeführten Klassen implementiert. Die Abbildung 2.20 zeigt einige an der Berechnung der unteren Schranken beteiligte Klassen. Hierbei sind die von Antonio Frangioni eingeführten Klassen fett eingerahmt.

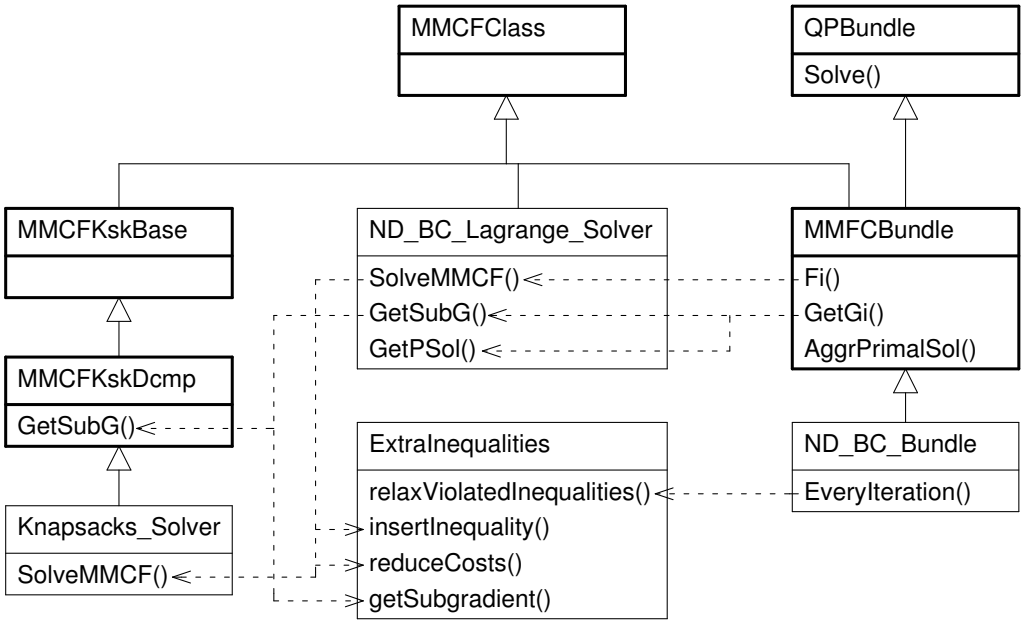


Abbildung 2.20: Klassendiagramm

Die Klasse `QPBundle` realisiert den im Abschnitt 2.6.2 beschriebenen Bundle-Algorithmus, bei dem der nächste Vektor ω in jeder Iteration durch die Lösung der quadratischen Optimierungsprobleme $(\Delta_{\beta t})$ und $(\Pi_{\beta t})$ festgelegt wird. Die Klasse `MMCFBundle` ist eine Spezialisierung des Bundle-Verfahrens auf die Lösung von Lagrange-Multiplikator-Problemen. Sie bestimmt die Interaktion zwischen dem Bundle-Algorithmus und der für die Lösung von Lagrange-Teilproblemen zuständigen Klasse. Eine wesentliche Funktion der Klasse `MMCFBundle` ist das Speichern der mit Subgradienten des Bündels β assoziierten Lösungen der Lagrange-Relaxation. Wir nutzen diese Funktionalität von `MMCFBundle`, um die y-Lösungen zu speichern.

Für unser Branch-and-Cut-Verfahren wurde eine von `MMCFBundle` abgeleitete Klasse `ND_BC_Bundle` eingeführt. Neben der Abwicklung des Bundle-Algorithmus hat diese Klasse die Aufgabe, das Einbeziehen zusätzlicher Ungleichungen in die Lagrange-Funktion zu leiten. Während der Lösung des Lagrange-Multiplikator-Problems startet `ND_BC_Bundle` in regelmäßigen Abständen die Suche nach gültigen Ungleichungen, die in den im Bündel gespeicherten Lösungen $\mathbf{y}$ verletzt werden. Für solche Ungleichungen werden neue Lagrange-Multiplikatoren (mit dem Initialwert 0) eingeführt, die Subgradienten aus β werden um neue Komponenten erweitert und das Bundle-Verfahren wird fortgesetzt. Eine andere Funktion der Klasse `ND_BC_Bundle` ist das Entfernen ungültiger Subgradienten aus dem Bündel β nach einer durch Variablenfixierung verursachten Änderung der Lagrange-Funktion.

Für die Lösung der Lagrange-Relaxation (2.4.2) wird bei Antonio Frangioni die Klasse `MMCFKskDcmp` verwendet. In unserer Implementierung wird zur Lösung der Rucksack-Relaxation eine von `MMCFKskDcmp` abgeleitete Klasse `Knapsacks_Solver` eingesetzt. Sie übernimmt die Datenstruktur und die meisten Methoden wie zum Beispiel die Berechnung von Subgradienten aus der Klasse `MMCFKskDcmp` und löst die Rucksackprobleme nach dem Schema des Algorithmus 2.

Die Berechnung unterer Schranken wird in unserem Branch-and-Cut-Verfahren mit einer Reihe weiterer Operationen, wie Variablenfixierung, Berechnung zulässiger Lösungen und Generieren zusätzlicher Ungleichungen, kombiniert. Zudem wird auch die Lagrange-Relaxation durch Kardinalitätsbedingungen und das Einbeziehen zusätzlicher Ungleichungen in die Zielfunktion modifiziert. Für die Lösung dieser modifizierten Rucksack-Relaxation und zum Integrieren der oben erwähnten Operationen in die Lagrange-Heuristik wurde die Klasse `ND_BC_Lagrange_Solver` eingeführt. Ihre Definition als eine Unterklasse von `MMCFClass` sichert die Kompatibilität mit `MMCFBundle`.

Die Klasse `ExtraInequalities` speichert die aus den Vorgängerknoten bekannten und im Laufe der Bundle-Optimierung generierten gültigen Ungleichungen für den aktuellen Knoten des Branch-and-Bound-Baums. Neben der Verwaltung zusätzlicher Ungleichungen hat `ExtraInequalities` die Aufgabe, die Kostenkoeffizienten der Rucksack-Relaxation entsprechend den Werten der mit zusätzlichen Ungleichungen assoziierten Lagrange-Multiplikatoren zu variieren und die mit diesen Ungleichungen assoziierten Komponenten des Subgradienten zu bestimmen.

2.14 Zusammenfassung

In diesem Abschnitt möchten wir die entwickelten Ideen und Algorithmen zusammenfassen. Die Abbildung 2.21 bietet einen Überblick über das Optimierungssystem für Netzwerkentwurf. Die Methoden und Algorithmen wurden als Komponenten entworfen, die miteinander kombiniert werden können. Die Branch-and-bound Algorithmen benötigen untere und obere Schranken, sowie Strategien für Branching, Baumsuche und Variablenfixierung.

Algorithmen: Branch-and-bound (2.3, Seite 19) und Relax-and-cut (2.12, Seite 70) bilden die beiden Hauptkomponenten des Systems. Die zusätzlichen Überdeckungsungleichungen (2.11.2, Seite 60) und lokalen Schnitte (2.11.3, Seite 66) können im Relax-and-cut-Algorithmus benutzt werden.

Untere Schranken: Zwei Lagrange Relaxationen stehen wahlweise zur Verfügung (2.4.2, Seite 21). Wir verwenden standardmäßig die Rucksack-Relaxation, weil sie deutlich schneller als die Kürzeste-Wege Relaxation ist.

Lagrange-Dual: Die Berechnung der unteren Schranken kann alternativ mit dem Subgradientenverfahren (2.6.1, Seite 30) oder der Bundle-Methode (2.6.2, Seite 32) erfolgen.

Obere Schranken: Zulässige Lösungen werden mittels Column Generation (2.7.2, Seite 43) oder mit der primalen Heuristik (2.7.3, Seite 44) berechnet.

Branching: Zwei Branchingstrategien stehen alternativ zur Auswahl: Variablendichotomie (2.8.1, Seite 46) und Branching mit Kardinalitätsbedingungen (2.8.2, Seite 47).

Baumsuche: Sowohl die Tiefensuche als auch die Bestensuche können eingesetzt werden.

Variablenfixierung: Exakte Strategien zur Variablenfixierung (2.9, Seite 50) sind: Rucksack-Fixierung, Kürzeste-Wege Fixierung (*shortest path*), kombinierte Strategie, sowie die Variablenfixierung mit Kardinalitätsbedingungen (CIT: *cardinality interval tightening*,

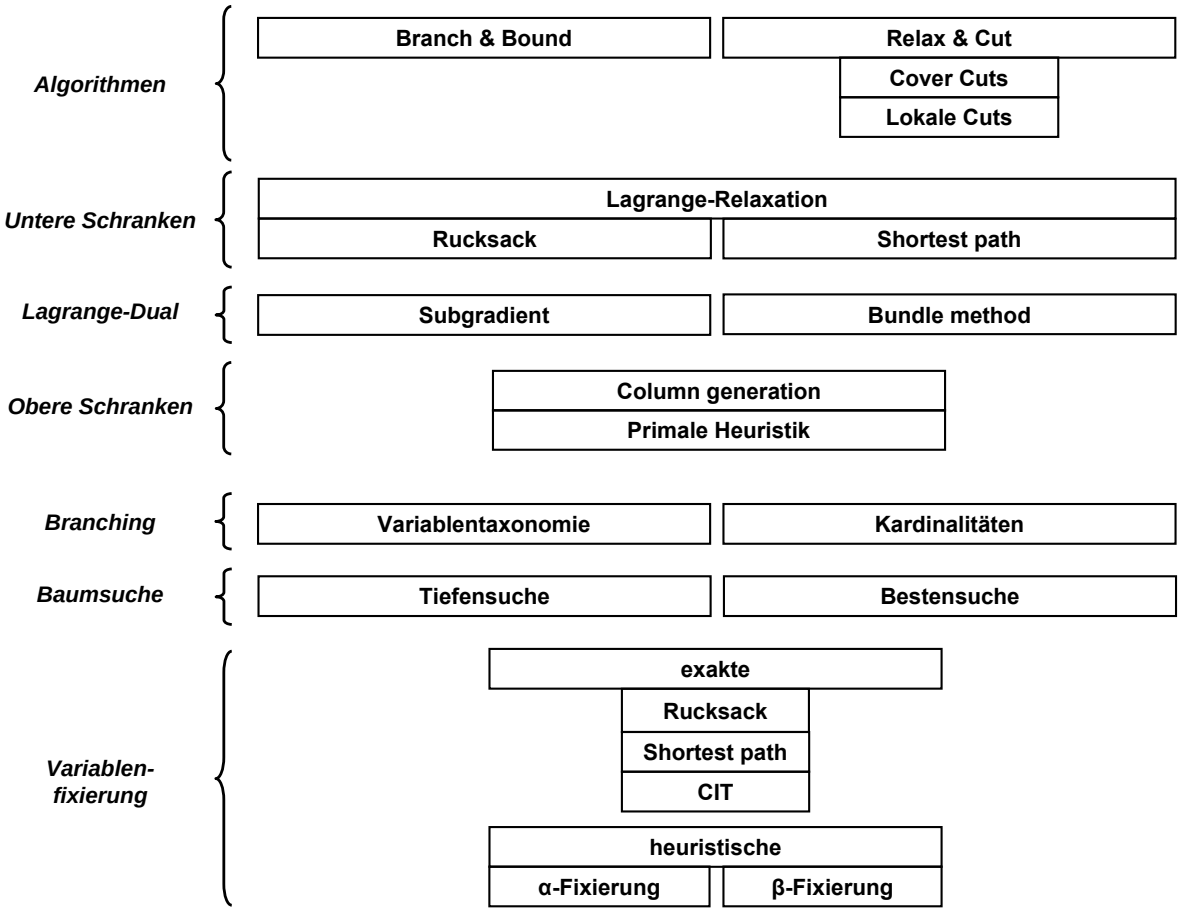


Abbildung 2.21: Überblick über die implementierten Algorithmen

2.9.4, Seite 54). Die beiden heuristischen Strategien - α- und β- Fixierung - verwandeln den jeweiligen Algorithmus in einen heuristischen Algorithmus für Netzwerkentwurf (2.10, Seite 56).

Jede Komponente verfügt weiterhin über mehrere Parameter, die durch den Benutzer eingestellt werden können, wobei jeweils auch Standardparameter vorgegeben werden.

Mehrere Konfigurationen des Systems sind nun möglich. Im Prinzip können die Komponenten beliebig miteinander kombiniert werden. Die in der Abbildung 2.21 nebeneinander stehenden Komponenten können jeweils alternativ eingesetzt werden; die übereinander stehenden können miteinander kombiniert werden.

Zwei Konfigurationen möchten wir gesondert hervorheben. Die erste ist in der Abbildung 2.22 dargestellt und wird im Folgenden als der **NDBB-Solver** bezeichnet.

Der **NDBB-Solver** besteht aus dem Branch-and-bound Algorithmus, der auf der Rucksack-Relaxation basiert (ohne zusätzliche Ungleichungen). Die unteren Schranken werden durch die Subgradientenmethode berechnet, die oberen sowohl exakt als auch heuristisch gesucht. Das Branching wird nach der Variablendichotomie durchgeführt. Die Rucksack-Variablenfixierung wird noch um die CIT-Fixierung ergänzt. Der NDBB-Solver wurde in Zusammenarbeit mit Meinolf Sellmann und Achim Koberstein entwickelt [Koberstein, 2002], [Sellmann, 2002].

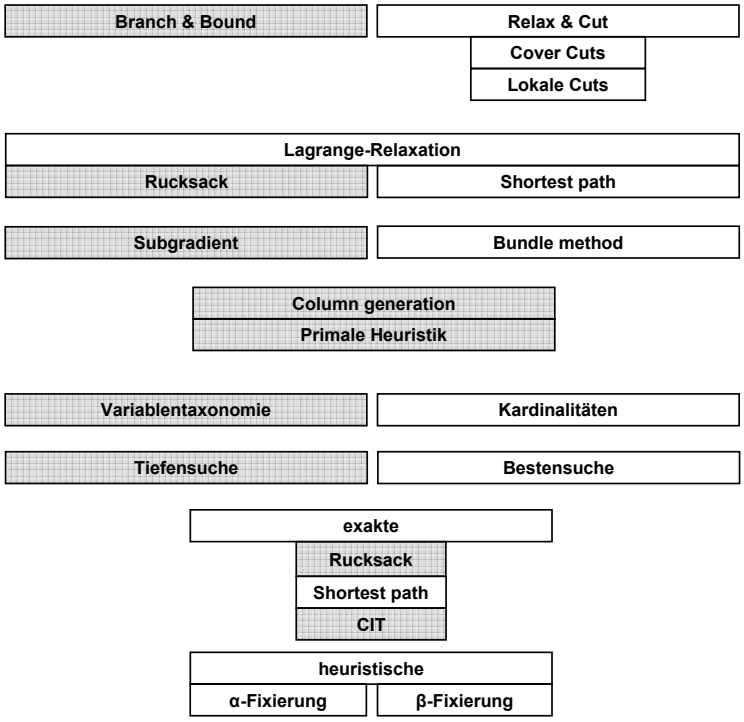


Abbildung 2.22: NDBB Solver: Network Design Branch-and-bound Solver

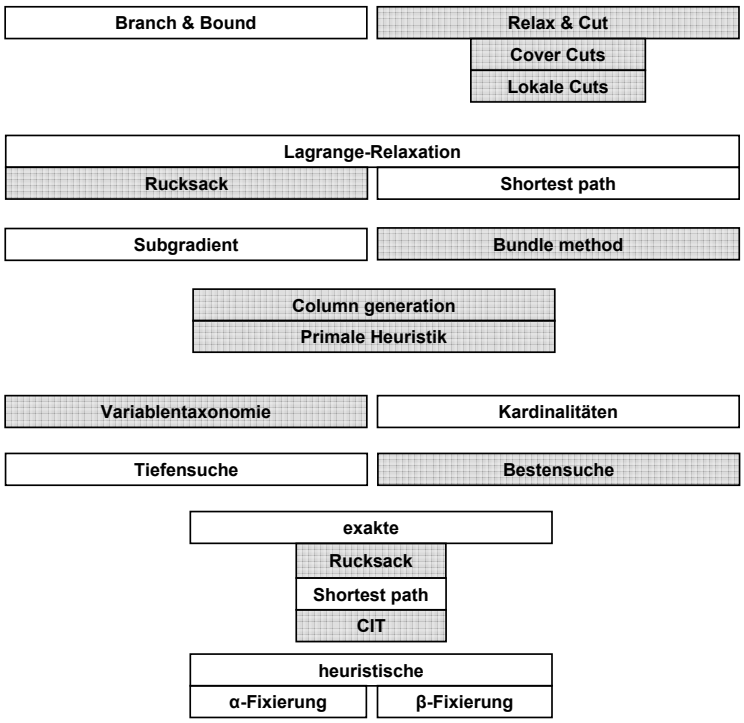


Abbildung 2.23: NDBC Solver: Network Design Relax-and-cut Solver

Die Ergebnisse des NDBB-Solvers wurden u.a. in der folgenden Publikation veröffentlicht:

- Meinolf Sellmann, Georg Kliewer, and Achim Koberstein. Lagrangian cardinality cuts and variable fixing for capacitated network design. In Proceedings of the 10th Annual European Symposium on Algorithms (ESA-2002), Springer, LNCS 2461, pages 845-858, Rome, Italy, 2002.

Der **NDBC-Solver** (Abbildung 2.23) implementiert den Relax-and-cut-Algorithmus, indem er zur Rucksack-Relaxation Überdeckungsungleichungen (*cover cuts*) und die lokalen Schnitte hinzufügt. Die Bundle-Methode wird zur Lagrange-Optimierung eingesetzt. Die anderen Komponenten entsprechen dem **NDBB-Solver**. Statt der Tiefensuche wird aber die Bestensuche benutzt. Der NDBC-Solver wurde in Zusammenarbeit mit Larissa Timajev entwickelt [Timajev, 2004] und u.a. in der folgenden Arbeit vorgestellt:

- Georg Kliewer, and Larissa Timajev. Relax-and-cut for capacitated network design. In Proceedings of the 13th Annual European Symposium on Algorithms (ESA-2005), 2005.

Im Kapitel 4.1 werden wir die Leistungsfähigkeit des Optimierungssystems bewerten. Es werden Vergleiche mit anderen Systemen vorgenommen sowie die Performance einzelner Komponenten untersucht.

— 3 –

Flottenzuweisung: Integration der Planungsphasen

Dieses Kapitel beginnt wir mit einer Motivation zur Integration unterschiedlicher Planungsschritte in der Flugplanung. Wir gehen genauer auf die Aufgaben der Marktmodellierung, der Flottenzuweisung und des Revenue Managements ein. Anschließend stellen wir drei Integrationsstrategien vor, die die Flottenzuweisung mit anderen Planungsphasen verbinden und auf diese Weise die Qualität des Planungsprozesses erhöhen.

3.1 Motivation

Der Prozess der Flugplanung findet, wie bereits im Kapitel 2.1 beschrieben, auf vielen Ebenen statt. Das Flugnetz, die Flugzeuge, das Personal und die Ticketverkäufe werden zeitlich parallel geplant und weisen hohe Abhängigkeiten untereinander auf. Diese Tatsache führt dazu, dass keine der Planungsaufgaben für sich alleine gelöst werden kann. Interaktion zwischen den Aufgaben und den zuständigen Planungsabteilungen ist notwendig, um zu einer guten Gesamtplanung zu gelangen. Eine vollständige Integration wird aber alleine schon wegen der enormen Anzahl der zu planenden Freiheitsgrade und Parameter kaum möglich sein. Das Potential der kooperierenden Planung wird in vielen Fällen aber noch nicht vollständig ausgeschöpft.

In dieser Arbeit beschäftigen wir uns mit der Integration der Planungsphase der Flottenzuweisung mit anderen benachbarten Planungsaufgaben. In den nächsten Abschnitten wird die Kopplung der Marktmodellierung und der Flottenzuweisung, sowie des Revenue Management und der Flottenzuweisung untersucht.

Verwandte Arbeiten zur Integration der Planungsphasen sind u.a.:
Flottenzuweisung und Aircraft Rotation ([Barnhart et al., 1998], [Grothklags, 2003]),
Flottenzuweisung und Schedule Design ([Lohatepanont, 2002]),
Flottenzuweisung und Crew Scheduling ([Shenoi, 1996]),

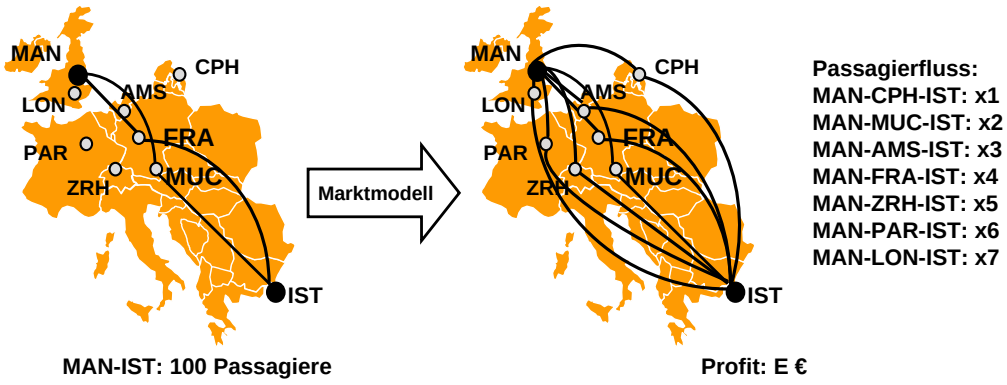


Abbildung 3.1: Beispiel: Ein Markt wird modelliert

Aircraft Rotation und Crew Pairing ([Klabjan et al., 2002], [Cohn and Barnhart, 2003]), Flottenzuweisung und Pricing/Revenue Management ([Jacobs et al., 2000a]).

3.2 Marktmodellierung

Das Marktmodell wird in der lang- bis mittelfristigen Planung eingesetzt (siehe Abbildung 2.1 auf Seite 8). Dort ist der Flugplan noch nicht endgültig festgelegt und der Planer benötigt in dieser Enwicklungsphase eine Entscheidungsunterstützung. Die Aufgabe der Marktmodellierung besteht darin, für ein gegebenes Flugnetzwerk die erwartete Anzahl der Passagiere auf jeder Flugstrecke vorauszusagen. Damit kann die Profitabilität des Flugplans abgeschätzt werden. Auf dieser Grundlage werden eine Analyse und Bewertung von Flugplanungsszenarien durchgeführt. Die nachfolgende Planung der Ressourcen greift immer wieder auf das Marktmodell zurück, um Veränderungen zu bewerten.

Marktmodellierung	
Eingabe	Flugplan (Flüge mit zugewiesener Flotte) Märkte (Städtepaare und Nachfragen)
Parameter	Regeln für den Aufbau der Reiserouten Parameter für die Verteilung der Marktnachfrage Definition der Kosten Definition der Erträge (Revenues)
Ausgabe	Passagierfluss Profite = Erträge - Kosten des Flugplans

Das Marktmodell soll auch voraussagen, wie sich die Passagiere auf die eigenen Flüge und Flüge der anderen Fluggesellschaften aufteilen werden.

3.2.1 Discrete-Choice-Modelle

Modelle des Passagierverhaltens (*passenger choice models*) basieren auf Präferenzen der Passagiere und erlauben Voraussagen, für welche der angebotenen Reisealternativen ein Passagier sich entscheiden wird.

Lösungsverfahren

Das Problem der Marktmodellierung wird von vielen Fluggesellschaften mit Hilfe von Verhaltensmodellen gelöst.

Zunächst wird die Nachfrage für alle Städtepaare (Märkte) im Flugnetzwerk geschätzt. Historische Daten werden dabei in die Prognose einbezogen. Im zweiten Schritt werden alle relevanten Reiseverbindungen (*itineraries*) für diese Märkte aufgebaut. Die Modellierung des Reiseverhaltens der Passagiere benötigt Informationen über die Präferenzen der Passagiere. Die Aufteilung der Passagiere auf die alternativen Reiseverbindungen erfolgt nach Prinzipien der *Discrete-Choice-Theorie*. Das Ergebnis liefert die geschätzte Anzahl der Passagiere auf jeder der zuvor generierten Reiseverbindungen. An dieser Stelle wird die Sitzkapazität auf den Flugstrecken noch nicht berücksichtigt (*unconstrained demand*). Die berechnete Anzahl der Passagiere übersteigt evtl. die Anzahl der verfügbaren Plätze. In diesem Fall müssen einige Passagiere einen anderen Reiseweg wählen. Im Modell wird vorausgesagt, wie sich abgewiesene Passagiere verhalten (*spill and recapture*). Im letzten Schritt werden die voraussichtlichen Erträge und die durch den Flugplan verursachten Kosten berechnet. Als Ergebnis erhält man die erwartete Profitabilität des Flugplans.

Algorithmus 13 Marktmodellierung

1: **Aufbau der Reiserouten (connection builder)**
2: **while** $(Maerkte \neq \emptyset)$ **do**
3: Berechne zulässige Pfade zwischen O und D: Menge I_{OD}
4: **Verteilung der Marktnachfrage (market share)**
5: **while** $(Maerkte \neq \emptyset)$ **do**
6: Berechne die Aufteilung der Passagiere im Markt OD auf die Pfade $p \in I_{OD}$
7: Anwendung des Multinomial Logit Modells
8: **Berücksichtigung der Kapazitäten (spill and recapture)**
9: **while** $(Fluege \neq \emptyset)$ **do**
10: Berechne für den Flug f die Anzahl der abgewiesenen Passagiere (spill)
11: Verteile diese Passagiere auf alternative Reiserouten (recapture)
12: **Kosten- und Ertragsrechnung**
13: Berechne für jeden Flug die entstehenden fixen und variablen Kosten
14: Berechne für jeden Flug den erzielbaren Ertrag
15: **return** Passagierfluss und die Profitabilitätsbewertung des Flugplans

Der Algorithmus 13 beschreibt die Vorgehensweise der Marktmodellierung. Als Ergebnis wird der Passagierfluss berechnet, der als Grundlage für die Profitabilitätsbewertung des Flugplans und gleichzeitig als Ausgangspunkt für die Einplanung der Flugzeugtypen im Netzwerk (Flottenzuweisung) dient.

Grundlagen der Discrete-Choice-Theorie

Die nachfolgende Darstellung der Discrete-Choice-Theorie lehnt sich an das Buch von Ben-Akiva und Lerman [Ben-Akiva and Bierlaire, 1985]. Die Grundlagen werden auch im Buchkapitel in [Ben-Akiva and Bierlaire, 1999] vorgestellt. In [Koppelman and Sethi, 2000] werden diverse Logit Modelle behandelt und deren Stärken und Schwächen miteinander verglichen.

Der Entscheidungsprozess eines Passagiers wird von vier Komponenten beschrieben: Entscheider, Alternativen, Attributen und Entscheidungsregel. Das Szenario ist wie folgt: Der Entscheider hat aus einer Menge der Alternativen eine auszuwählen. Die Entscheidung wird nach der Entscheidungsregel vorgenommen, die die Attribute der Alternativen auswertet. Jeder Entscheider besitzt eine Nutzenfunktion, die von den Attributen abhängt. Da dem Entscheider nicht unbedingt vollständige Informationen vorliegen, beeinflusst eine Zufallskomponente die ansonsten deterministische Nutzenfunktion.

Eingabedaten und Parameter

C_n	Auswahlmenge des Entscheiders n
$i \in C_n$	Alternative i aus der Auswahlmenge n
V_{in}	deterministischer Anteil der Nutzenfunktion des Entscheiders n bei der Alternative i
β_k	Gewichtsparameter für Attribut k der Alternative i
x_{ink}	Attribut k für die Kombination (i,n)
ε_{in}	Zufallsterm
U_{in}	Nutzenfunktion des Entscheiders n bei der Alternative i

$$V_{in} = \sum_k \beta_k x_{ink}$$

$$U_{in} = V_{in} + \varepsilon_{in}$$

$$P(i \mid C_n) = P(U_{in} \geq U_{jn} \forall j \in C_n) = P(U_{in} = max_{j \in C_n} U_{jn})$$

Der deterministische Nutzen V_{in} ist eine Summe der gewichteten Attribute einer Alternative. In der Nutzenfunktion U_{in} des Entscheiders n kommt noch zusätzlich eine Komponente ε_{in} vor, die die Zufälligkeit der Entscheidung abbilden soll. Der Passagier entscheidet sich damit für die wertvollste Alternative $U_{in} = max_{j \in C_n} U_{jn}$.

Die Annahme über Verteilung der Zufallskomponente ε_{in} definiert zwei große Familien der Discrete-Choice-Modelle. Die Logit-Familie benutzt Gumbel-verteilte Variablen, normalverteilte Variablen führen zu der Familie der Probit Modelle. Beide Modelle haben Vorteile und Nachteile, auf die wir im späteren Verlauf des Kapitels eingehen werden.

Logit und verwandte Modelle

Bei der folgenden Annahme über die Zufallskomponente ε_{in}

$$F(\varepsilon) = e^{-e^{-\mu(\varepsilon - \eta)}}, \mu > 0(scale), \eta(location)$$

ist die Wahrscheinlichkeit, dass ein Entscheider die Alternative i aus der Menge C_n auswählt, relativ einfach zu berechnen:

$$P(i \mid C_n) = e^{\mu V_{in}} / \sum_{j \in C_n} e^{\mu V_{jn}}$$

Vorteile der Logit-Modelle sind ihre einfache Implementierbarkeit und vorliegende effiziente Schätzalgorithmen. Nachteil ist: explizite Annahme über Unabhängigkeit der Zufallskomponenten. Um dieses Nachteil auszumerzen, sind mehrere Erweiterung des Logit-Modells untersucht worden, z.B. die Probit- und Dogit-Modelle.

Probit-Modelle können die gegenseitigen Abhängigkeiten der Alternativen abbilden. Nachteilig sind die aufwendigen Schätzalgorithmen, die keinen Praxiseinsatz in großen Systemen erlauben. Dogit-Modelle sind in der Diplomarbeit von Müller-Bungart untersucht worden und verhalten sich besser als Logit-Modelle ([Müller-Bungart, 2003]).

Marktmodell der Lufthansa Systems

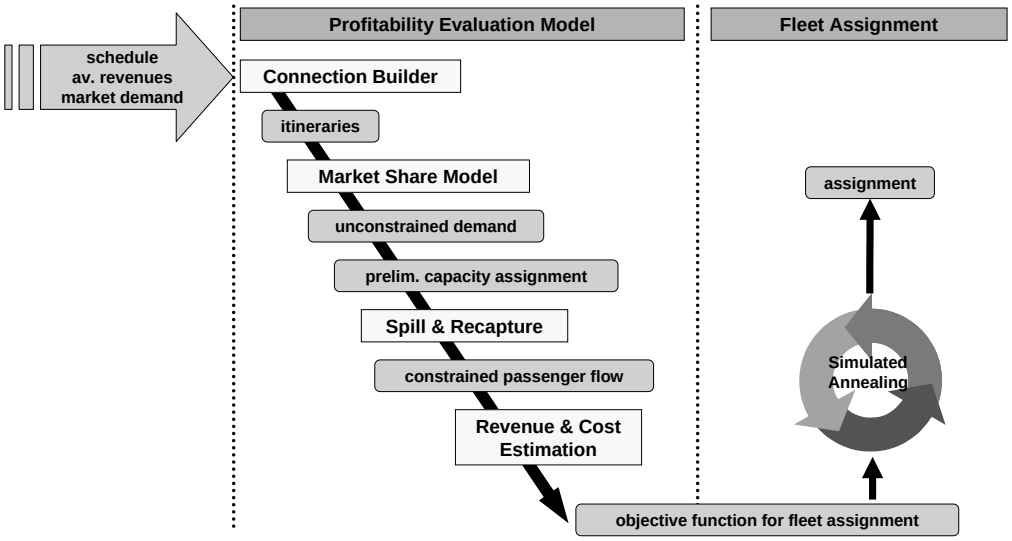


Abbildung 3.2: Die Vorgehensweise des Marktmodellierung

Das Logit-Marktmodell der Lufthansa Systems ist Bestandteil des Produkts NetLine/Plan und wird von mehreren Fluggesellschaften eingesetzt[1].

Uns stand für diese Arbeit eine Version von NetLine/Plan zur Verfügung [LufthansaSystems, 2005]. Die Experimente wurden auf Realdaten von Fluggesellschaften durchgeführt. In einigen Experimenten war eine Anpassung oder Selektion der Daten notwendig, um die wesentlichen Effekte besser darstellen zu können.

Auf die detaillierten Angaben der Parameter des Marktmodells müssen wir in dieser Arbeit verzichten, weil diese nicht veröffentlicht werden dürfen. Es gibt z.B. eine Vielzahl von Parametern, die die Regeln festlegen, nach denen die zulässigen Reiseverbindungen gebildet werden. Die Parameter des Logit-Modells müssen so kalibriert werden, dass die Voraussagen eine möglichst gute Qualität haben. Die Kosten- und Ertragsparameter sind spezifisch für jede Fluggesellschaft und stellen das vielleicht am stärksten gehütete Betriebsgeheimnis dar.

Die Dissertation von Radicke [Radicke, 1994] stellt Lösungsverfahren für das Fleet Assignment Problem vor, die auf Informationen aus dem Marktmodell der Lufthansa Systems beruhen. Die Bewertung des Marktmodells wird im Fleet Assignment Algorithmus simuliert.

In einem internen Dokument von Lufthansa Systems [Lefeld and Pölt, 1995] werden die sechs Bestandteile des Marktmodells detailliert beschrieben: Connection-Builder, das Logit-Modell, das Spill-and-Recapture Modul, das Kosten-Modul und die Profitabilitätsbewertung.

[1] NetLine/Plan unterstützt die mittel- und langfristige Streckennetzplanung von Fluggesellschaften. Auf der Basis der Airline-eigenen Flugpläne sowie der Flugpläne der Mitbewerber werden Prognosen für Passagierströme, Erträge und Kosten erstellt. Damit können geplante Änderungen des Flugplans analysiert und Flugpläne optimiert werden. Die Möglichkeit, interaktiv neue Verbindungen, Zeit-, Equipment- und Code-Share Änderungen zu analysieren und evaluieren, zählt zu den besonderen Stärken von NetLine/Plan. Zusatzmodule für die automatisierte Optimierung erlauben es, komplexe Szenarien zu evaluieren. Dies ermöglicht es der planenden Airline, die Umsatz- und Ertragspotentiale ihres Netzwerks voll zu erschließen. Quelle: www.lhsystems.com

In [Sieber, 1995] wird die Spill-and-Recapture Komponente genauer vorgestellt. In der technischen Dokumentation [LufthansaSystems, 1997] werden Eingabedaten, Parameter und das Verhalten des Marktmodells beschrieben. Im Projektbericht [2] [PARALOR, 1997] wird das Marktmodell und dessen Parallelisierung im Rahmen des BMBF-Projekts beschrieben.

In zwei Arbeiten [Müller-Bungart, 2003], [Scheidler, 2003] werden die im Marktmodell benutzten Logit-Modelle untersucht. Beide stellen fest, dass einige Annahmen, die das multinomiale Logit-Modell trifft, in der Praxis oft nicht zutreffen. Als Abhilfe werden neue Modelle vorgestellt. Müller-Bungart untersucht das sog. Dogit-Modell und stellt fest, dass die IIA-Annahme (Unabhängigkeit von irrelevanten Alternativen) in vielen Märkten nicht haltbar ist und die Passagiere sich nicht immer an ihrem Nutzen orientieren (Loyalität). Das Dogit-Modell behandelt explizit die Loyalität der Passagiere und geht nicht von der IIA-Eigenschaft der Märkte aus. Scheidler erweitert das Logit-Modell ebenfalls so, dass die IIA-Eigenschaft nicht gegeben sein muss.

3.2.2 Weitere Methoden der Marktmodellierung

In seiner Dissertation[3] gibt Kniker einen guten Überblick über die Probleme und Lösungsverfahren im Bereich Marktmodellierung (*airline passenger itinerary selection process*)[Kniker, 1998].

Im Übersichtsartikel [Etschmaier and Mathaisel, 1984] wird im Abschnitt *Schedule Evaluation Models* der Stand der Technik bis 1984 beleuchtet.

Der aktuellere Beitrag von Viswanathan (Sabre)[Viswanathan, 1999] gibt einen Überblick über Techniken in diesem Bereich an.

In der Dissertation von Ramming [Ramming, 2002] werden Modelle für die Wahl von Reiserouten im Straßenverkehr diskutiert.

In der Literatur findet man weitere Verfahren für die Vorhersage des Passagierverhaltens:

- **QSI-Modelle** (*quality of service index*) teilen die Menge der Passagiere auf die Alternativen rein nach ihrer Attraktivität auf. Berücksichtigt werden dabei z.B. die Dauer der Verbindung, Anzahl der Umsteigeverbindungen etc.

- **Zeitreihenmodelle** (*time series models*) basieren auf historischen Daten und versuchen, Trends in der Entwicklung zu erkennen und damit eine Zukunftsprognose zu machen.

- **Regressionsmodelle** versuchen, Abhängigkeiten zwischen relevanten Faktoren zu finden und durch Abbildung von weiteren Faktoren eine Prognose zu machen.

- **Simulationsmodelle** werden z.B. in den Arbeiten zu PODS von Boeing [Parker, 2003], in den Dissertationen von Farkas[Farkas, 1996] und Carrier[Carrier, 2003] behandelt. Die Simulation generiert Buchungsanfragen von Passagieren und wird in erster Linie für die Evaluation von Methoden des Revenue Managements benutzt. PODS kann aber auch zum Bewerten von Flugplänen verwendet werden.

- **Analytische Methoden für User-Equilibrium.** In den Arbeiten von Soumis wird im Gegensatz zu einem Systemoptimum nach einem User-Equilibrium gesucht. Die Passagiere entscheiden sich für eine Reiseverbindung abhängig von ihrer Attraktivität. Diese

[2] Seiten 88 - 92.

[3] Kapitel 3.3, Seiten 56 - 60

Entscheidungen werden in einem Markov-Prozess berechnet. Nichtlineare Bedingungen bringen den Passagierfluss ins Gleichgewicht mit den angebotenen Kapazitäten. (Siehe [Soumis and Nagurney, 1993], [Dumas and Soumis, 2004])

- **Kommerzielle Systeme**: Lufthansa Systems entwickelte das bereits beschriebene NetLine/Plan System. Sabre AirFlite Profit Manager[Sabre, 2004] und United Airlines Profitability Forecasting Model[Usman, 2002], [Sivakumar, 2003] benutzen ebenfalls ein Logit-Modell für die Marktmodellierung. O&D FAM von Sabre[Jacobs et al., 2000b] bewertet Flugpläne mit einem vereinfachten O&D (origin-destination) Revenue Management Modell.

3.2.3 Zusammenfassung

Marktmodelle werden in der Flugplanung eingesetzt, um Flugplanszenarien zu bewerten. Wir haben einige Klassen von Marktmodellen vorgestellt, u.a. Discrete-Choice-Modelle. In den folgenden Kapiteln werden wir untersuchen, wie die Aufgabe der Marktmodellierung mit der Planungsphase der Flottenzuweisung integriert werden kann.

3.3 Revenue Management

In diesem Abschnitt geben wir einen kurzen Überblick über die Verfahren im Bereich Revenue Management. Die Beschreibung ist an die Darstellung im Buch von Talluri und van Ryzin angelehnt [Talluri and van Ryzin, 2005].

Die Aufgaben des Revenue Managements (oder Ertragsmanagements) beinhalten die Aufteilung der Plätze auf einem Flug in unterschiedliche Kategorien, die Festlegung der Ticketpreise und die Steuerung der Ticketverkäufe. Diese Aufgaben werden während der ganzen Buchungsperiode wahrgenommen, d.h. zwischen der Veröffentlichung des Flugplans für eine Periode (Sommer-/Winterflugplan) und dem jeweiligen Flug. Das Ziel des Revenue Management (RM) kann so beschrieben werden:

Verkaufe das richtige Flugticket zum richtigen Zeitpunkt an die richtige Person zum richtigen Preis.

Wesentliche Merkmale eines RM-Systems sind:

- Differenzierung der Tickets nach Verkaufs- und Umbuchungsbedingungen (Business, Economy, Discount) mit bis zu 10 unterschiedlichen Preisklassen (*product differentiation*).
- Während des Buchungszeitraums werden die Preise dynamisch an die Nachfrage angepasst (*dynamic pricing*).
- Steuerung der Verfügbarkeit der einzelnen Preisklassen unter Berücksichtigung der Buchungsprognosen (*inventory control*).

Im Bild 3.3 ist der Aufbau eines typischen Revenue Management Systems dargestellt. Die Abbildung 3.4 veranschaulicht den Zusammenhang zwischen Flugplanung, Pricing und Revenue Management.

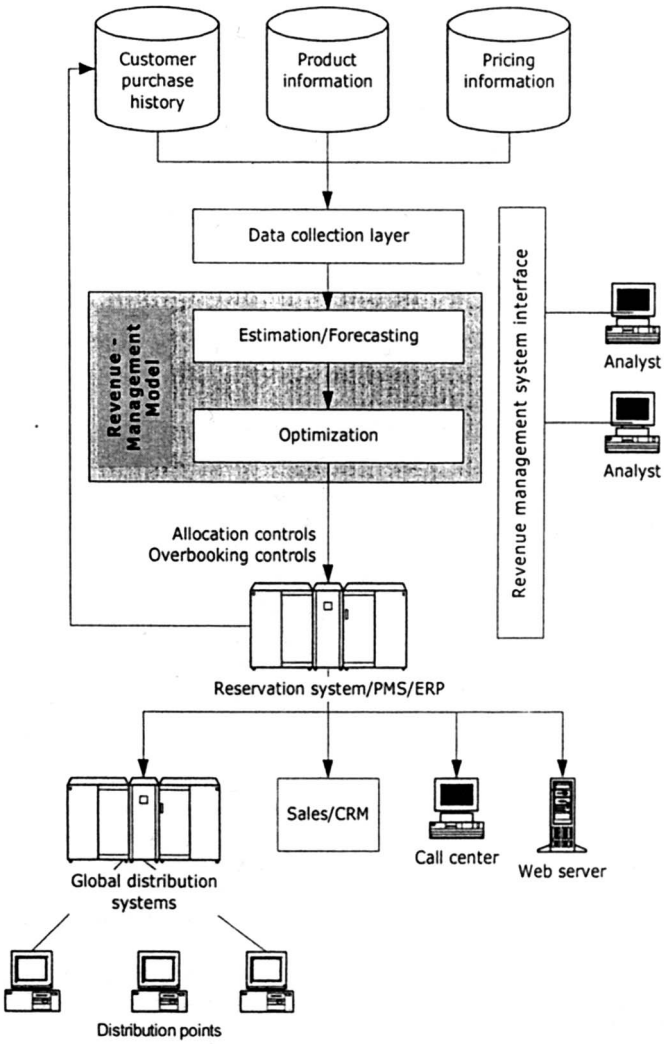


Abbildung 3.3: Ein Revenue Management System (Quelle: [Talluri and van Ryzin, 2005])

Relevante historische Daten werden zunächst gesammelt und für die Prognosen aufbereitet. Die Passagiernachfrage wird modelliert und daraus werden Buchungsprognosen erstellt, die laufend angepasst und verfeinert werden müssen.

Bei der Optimierung geht es darum, eine optimale Steuerung des Ticketverkaufs zu organisieren. Aktuelle Preise und Regeln für die Annahme oder Abweisung von Buchungsanfragen werden aufgestellt.

Das Reservierungssystem empfängt über sämtliche Verkaufskanäle (Reisebüros, Call-Center, Internet) Anfragen und bedient sie nach der aktuell gültigen Steuerungsstrategie.

Die wichtigsten Bestandteile eines RM-Systems sind die Prognosemodelle für Buchungen und die Optimierungsverfahren zur Kapazitätssteuerung.

Prognosemodelle müssen der Tatsache Rechnung tragen, dass unterschiedliche Kundengruppen unterschiedliches Buchungsverhalten aufweisen. Im Bild 3.5 sind die zeitlichen Verläufe beispielhaft dargestellt. Business-Kunden buchen typischerweise erst ein Paar Tage vor Abflug und möchten vor allem flexible Umbuchungsmöglichkeiten erhalten. Economy- und Discount-

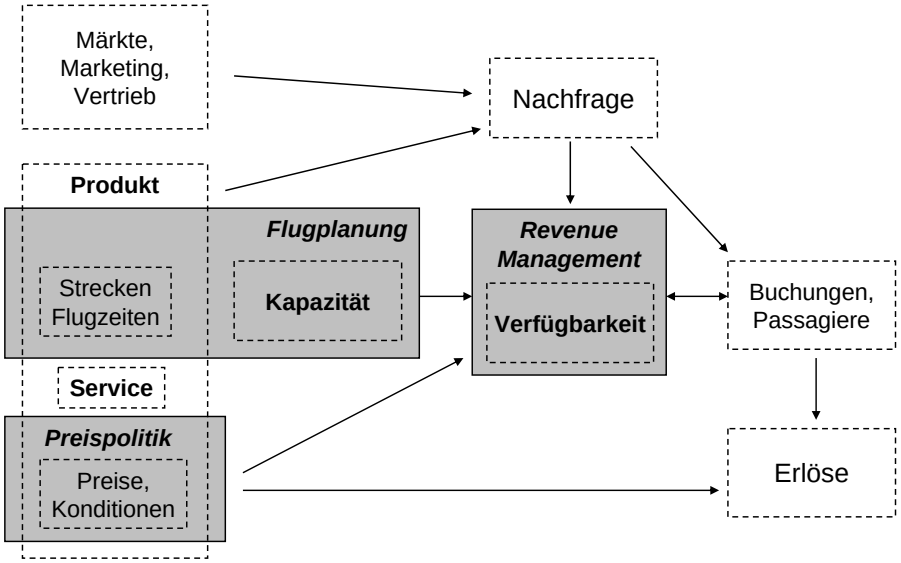


Abbildung 3.4: Der Zusammenhang von Flugplanung, Pricing und Revenue Management (Quelle: [Sterzenbach and Conrady, 2003, Pölt, 2001])

Kunden buchen lange vor Abflug und nehmen diverse Restriktionen (z.B. Wochenende-Regel) in Kauf, vorausgesetzt der Ticketpreis bleibt niedrig.

3.3.1 Steuerungsstrategien

Die grundsätzliche Frage des Revenue Managements lautet deswegen:

Soll eine Economy- oder Discount-Buchungsanfrage jetzt akzeptiert werden oder lohnt es sich, den betreffenden Platz für eine spätere Business-Anfrage zu reservieren?

Im einfachen Fall von nur zwei Klassen mit Preisen p_1 und p_2 mit $p1 > p2$ und geschätzten Bedarfen D_1 und D_2 (Wahrscheinlichkeitsfunktionen) kann der Platz x an die Anfrage zum Preis p_2 vergeben werden, falls gilt:

$$p_2 \geq p_1 \cdot P(D_1 \geq x)$$

Der Preis p_2 übersteigt also den geschätzten erzielbaren Preis für diesen Sitz in der Klasse 1. Da die Funktion $P(D_1 \geq x)$ fallend in x ist, existiert ein optimaler Wert y_1^* mit:

$$p_2 = p_1 \cdot P(D_1 \geq y_1^*)$$

Bei einer kontinuierlichen Wahrscheinlichkeitsfunktion $F_1(x)$ kann dieser optimale Wert berechnet werden (Littlewood's Regel [Littlewood, 1972]):

$$y_1^* = F^{-1}(1 - p2/p1)$$

Entsprechend dieser Regel kann eine Buchungsgrenze von y_1^* Plätzen definiert werden, die die Klasse-1 vor Klasse-2 Buchungen schützt.

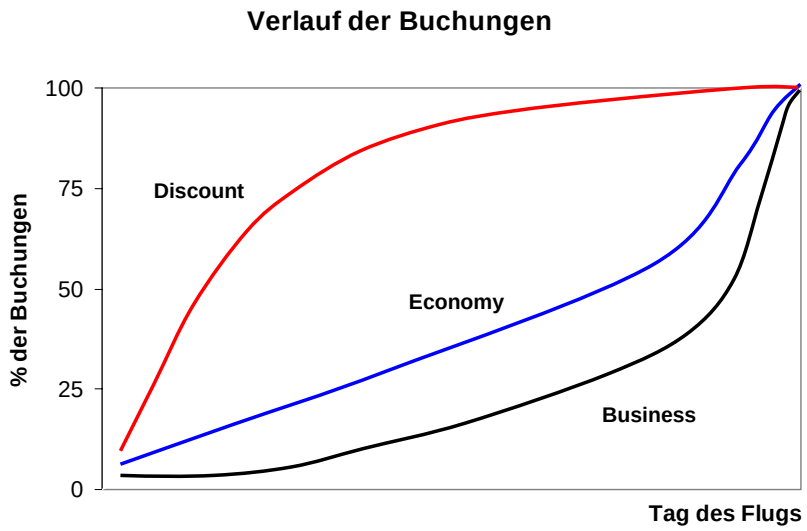


Abbildung 3.5: Buchungsverhalten unterschiedlicher Kundengruppen

Eine Verallgemeinerung dieser Regel auf n Klassen wurde von Belobaba in [Belobaba, 1989] vorgeschlagen. Die heuristische Strategie **EMSR-b** (*expected marginal seat revenue - version b*) findet eine sehr breite Verwendung in RM-Systemen vieler Fluggesellschaften.

Die Klassenanfragen mit Preisen $p_1 > p_2 > ... > p_n$ mit Bedarfsfunktionen $F_j(x)$ erreichen in der Reihenfolge $n, ..., 2, 1$ das RM-System.

In der Runde $j+1$ betrachten wir die Anfrage mit dem Preis p_{j+1} und suchen nach einer Buchungsgrenze y_j für die Klassen $j, j-1, ..., 2, 1$. Der Gesamtbedarf dieser Klassen ist $S_j = \sum_{k=1}^{j} D_k$. Der gewichtete Durchschnittspreis der Klassen $1, ..., j$ ist

$$\bar{p_j} = \frac{\sum_{k=1}^{j} p_k \cdot E[D_k]}{\sum_{k=1}^{j} E[D_k]}$$

Die Buchungsgrenze wird festgelegt auf

$$P(S_j > y_j) = \frac{p_{j+1}}{\bar{p_j}}$$

In jeder Runde kann so bestimmt werden, wie viele Buchungsanfragen in der jeweiligen Preisklasse akzeptiert werden können.

Wie bereits im Abschnitt 3.5.1 beschrieben wurde, können Ressourcen, wie z.B. Plätze auf einem Flug, nicht unabhängig von anderen Flügen optimal zugeteilt werden. In diesem Fall muss die Kontrolle auf der Netzwerkebene ausgeführt werden (*O&D-RM: origin-destination revenue management*).

Von den existierenden Arten der Netzwerkkontrolle möchten wir in diesem Abschnitt auf die *bid-price* Steuerung erwähnen.

Ein bid-price definiert eine Preisschwelle für jede Ressource im Netzwerk. Bei einer Buchungsanfrage für eine Reiseverbindung (die evtl. aus mehreren Flügen besteht) wird deren Preis mit der Summe der bid-prices der einzelnen Flüge verglichen und die Anfrage akzeptiert, falls dieser Preis höher ist.

Bid-price Kontrollstrategien sind nicht immer optimal, liefern aber eine gute Approximation der optimalen Kontrolle [Talluri and van Ryzin, 1998].

Bid-prices können mit unterschiedlichen Methoden berechnet werden:

- Globale Approximationsmethoden:
  - deterministisches lineares Modell
  - probabilistisches oder randomisiertes lineares Modell
- Dekompositionsmethoden:
  - *OD factors* Methode
  - *prorated EMSR*
  - *DAVN: displacement-adjusted virtual nesting*
  - Dynamische Programmierung
  - *iterative DAVN*
  - *iterative prorated EMSR*

Im Abschnitt 3.6 gehen wir genauer auf das deterministische lineare Modell ein.

Das Ergebnis dieser Methoden ist ein bid-price für jeden Flug im Netzwerk. Die Berechnung des Passagierflusses erfolgt als nächster Schritt. Die Buchungsanfragen können z.B. simuliert werden. In der vereinfachten Version können die Passagierzahlen in den einzelnen Preisklassen gleich den Erwartungswerten angenommen werden. Aus dem so berechneten Passagierfluss kann der Gesamtertrag im Flugnetz berechnet werden.

Diese Ertragsschätzung wird an die Flottenzuweisung weitergegeben, damit die Optimierung darauf reagieren und eine bessere Lösung berechnen kann.

3.4 Flottenzuweisung

Bei der Flottenzuweisung (Fleet Assignment) wird für jeden Flug der Typ des zugehörigen Flugzeugs bestimmt. Dieser bestimmt unter anderem die Zahl der Passagiere, die auf dieser Strecke befördert werden können. Dabei ist die Anzahl der Flugzeuge je Flugzeugtyp beschränkt. Der Gewinn der Flottenzuweisung soll maximiert werden. Die einzelnen Gewinne für jede mögliche Zuweisung eines Flugzeugtyps zu einem Flug wurden vor der Berechnung der Flottenzuweisung durch die Marktmodellierung geschätzt.

3.4.1 Modellierung

Das Problem wird als ein Mehr-Güter-Fluss-Problem im Flugnetzwerk modelliert. Die unterschiedlichen Flotten entsprechen dabei den zu transportierenden Gütern. Es wird ein *time-space*-Netzwerk definiert, in dem für jede mögliche Flugbewegung auf einem Flughafen ein Knoten und für jede Flugverbindung eine „Flug-Kante“ existiert. Außerdem gibt es in diesem Netzwerk noch „Bodenkanten“, die zwei aufeinander folgende Flugereignisse auf einem Flughafen verbinden. Ein Beispiel mit drei Flughäfen ist im Bild 3.7 dargestellt.

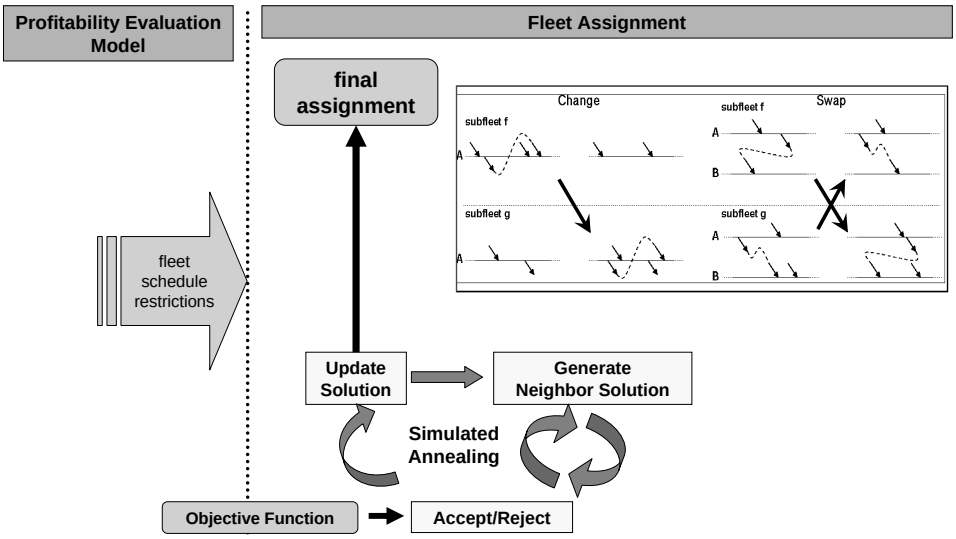


Abbildung 3.6: Der Algorithmus für die Flottenzuweisung

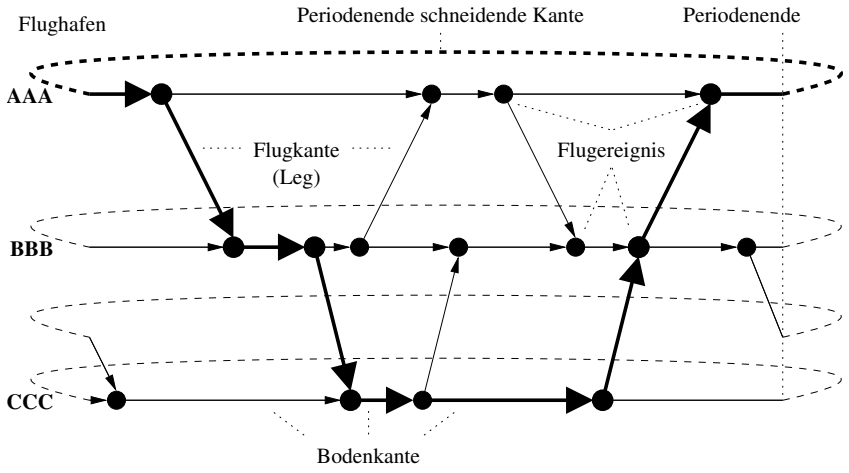


Abbildung 3.7: Time-Space Netzwerk für das Fleet Assignment Problem

Eingabedaten und Parameter

$\mathscr{N}^{\mathrm{TS}}$	Knoten des Time-Space Netzwerks
$\mathscr{A}$	Menge der Flughäfen
$\mathscr{E}_{\mathrm{F}}$	Flugkanten des Time-Space Netzwerks
$\mathscr{E}_{\mathrm{G,f}}$	Bodenkanten des Time-Space Netzwerks
$\mathscr{E}_{\mathrm{F0}}, \mathscr{E}_{\mathrm{G0,f}}$	Flugkanten und Bodenkanten, die über die Null-Uhr-Linie gehen
$\mathscr{F}$	Menge der Flugzeugtypen (subfleets)
$size_f$	Anzahl der verfügbaren Flugzeuge in der subfleet f
$v \in \mathscr{N}^{\mathrm{TS}}$	$v = (\mathrm{time}, a, f)$ ein Flugereignis zum Zeitpunkt time, Flughafen a, subfleet $f \in \mathscr{F}$
$v^-, v^+ \in \mathscr{N}^{\mathrm{TS}}$	Vorgänger- und Nachfolgeflugereignis von v
p_{lf}	Profit für ein Passagier auf dem Flug $l \in \mathscr{E}_{\mathrm{F}}$ mit der subfleet $f \in \mathscr{F}$

Entscheidungsvariablen

y_{lf}	Die Flugkante $l \in \mathscr{E}_{\mathrm{F}}$ wird vom Flugzeugtyp $f \in \mathscr{F}$ bedient
$z_{v,v+}$	Anzahl der Flugzeuge auf einer Bodenkante zwischen v und v^+

Das mathematische Modell für das Flottenzuweisungsproblem mit Hilfe des Time-Space Netzwerks sieht wie folgt aus ([Hane et al., 1995]):

$$\text{Maximize} \quad \sum_{l \in \mathscr{E}_{\mathrm{F}}} \sum_{f \in \mathscr{F}} p_{lf} y_{lf}$$

$$\forall l \in \mathscr{E}_{\mathrm{F}}: \quad \sum_{f \in \mathscr{F}} y_{lf} = 1 \qquad \text{(FA1)}$$

$$\forall v \in \mathscr{N}^{\mathrm{TS}}: \quad \sum_{arr(l,f)=v} y_{lf} - \sum_{dep(l,f)=v} y_{lf} + z_{v-,v} - z_{v,v+} = 0 \qquad \text{(FA2)}$$

$$\forall f \in \mathscr{F}: \quad \sum_{l_f \in \mathscr{E}_{\mathrm{F0}}} y_{lf} + \sum_{(v,v+) \in \mathscr{E}_{\mathrm{G0,f}}} z_{v,v+} \leq size_f \qquad \text{(FA3)}$$

$$\forall l \in \mathscr{E}_{\mathrm{F}}, \forall f \in \mathscr{F}: \quad y_{lf} \in \{0,1\} \qquad \text{(FA4)}$$

$$\forall (v,v+) \in \mathscr{E}_{\mathrm{G,f}}: \quad z_{v,v+} \geq 0 \qquad \text{(FA5)}$$

Die Gleichungen (FA 1) garantieren, dass jede Flugkante von genau einer subfleet geflogen wird. Die Gleichungen (FA 2) stellen sicher, dass die Flusseigenschaft im Graphen erhalten bleibt. Die Bedingung (FA 3) schränkt die Anzahl der einsetzbaren Flugzeuge ein.

3.4.2 Literaturübersicht

Für die Bestimmung einer Flottenzuweisung wurden in der Literatur mehrere exakte und heuristische Methoden entwickelt. Bei den exakten Ansätzen wird eine optimale Lösung mittels Branch-and-Bound berechnet.

Die **ersten Arbeiten** im Bereich Flottenzuweisung von [Ferguson and Dantzig, 1956] erschienen bereits 1956. Sie entwickelten mathematische Modelle für die Zuweisung von Flugzeugen zu Flügen, erfassten die wichtigsten Problemeigenschaften und begründeten damit jahrzehntelange Forschung auf diesem Gebiet. Frühe Ansätze für das Fleet Assignment Problem (FAP) arbeiteten mit heuristischen Varianten des Frank-Wolfe Algorithmus und nutzten die Netzwerk-Struktur des Problems aus [Soumis et al., 1980]. Im Übersichtsartikel von [Etschmaier and Mathaisel, 1984] werden die frühen Arbeiten beschrieben.

Abara formulierte das FAP als ein **Multicommodity Flussproblem** mit zusätzlichen Constraints [Abara, 1989]. Das von ihm vorgestellte Modell wurde das *connection-network* Modell genannt.

In [Daskin and Panayotopoulos, 1989] wird ein auf Lagrange-Relaxation basierter Ansatz für das FAP vorgestellt, bei dem Routen, die von einem Hub ausgehen, mehrere Zwischenflughäfen besuchen und zum Hub zurückkehren, Flugzeugtypen zugewiesen werden.

Das *time-space network* Modell aus [Hane et al., 1995] wurde zum **Standardmodell für das Fleet Assignment Problem**. Der vorgestellte Ansatz wird bei mehreren großen amerikanischen Fluggesellschaften eingesetzt. Die betrachteten Probleme basieren auf einer Tagesplanung, die dann über die ganze Woche unverändert realisiert wird, mit kleineren Änderungen am Wochenende. Im Rahmen des Branch-and-Bound Algorithmus werden die

LP-Probleme in jedem Knoten mit einem Interior-Point Algorithmus oder einem Steepest-Edge Simplex-Algorithmus gelöst. Einen sehr wichtigen Bestandteil bilden problemspezifische Reduktionsverfahren, wie z.B. die Inselbildung oder die Wahl der Branching-Variablen.

In [Berge and Hopperstad, 1993] werden verschiedene heuristische Techniken für das *time-space network* Modell entwickelt. Außerdem werden Strategien vorgestellt, wie man auf Schwankungen der Passagiernachfrage kurzfristig reagieren kann.

In [Desaulniers and Desrosiers, 1997] wird die Flottenzuweisung auf Wochenbasis durchgeführt.

Der Einsatz der exakten mathematischen Optimierung und insbesondere von **kommerziellen Lösern** wie z.B. CPLEX führte zu einem durchschlagenden Erfolg bei vielen Airlines. Die Ergebnisse von Abara [Abara, 1989] wurden in Zusammenarbeit mit American Airlines erreicht. [Hane et al., 1995] entwickelten ihre Modelle und Lösungsverfahren für Delta Air Lines. [Subramanian et al., 1994] berichten über ihre Erfahrungen mit den großen FAP Modellen bei Delta Air Lines.

Die Arbeiten bei USAir in diesem Bereich werden in [Rushmeier and Kontogiorgis, 1997] dargestellt.

Die **Komplexität** des FAP wurde in der Arbeit von [Gu et al., 1994] untersucht. Das FAP ist bereits für drei Flugzeugtypen NP-vollständig.

Ansätze, die neben dem Finden einer guten Lösung einen Beweis der Optimalität durchführen, stoßen bei größeren Probleminstanzen oder bei zusätzlichen Nebenbedingungen oft schnell an ihre Grenzen. Eine optimale Lösung anzustreben ist in der Praxis angesichts langer Berechnungszeiten oft nicht sinnvoll, insbesondere wenn es sich – wie bei der Flottenzuweisung – bei den zugrunde liegenden Daten nur um Schätzungen handelt. Deswegen wird in der Praxis auch ein potentiell exakter Ansatz nach Berechnung einer guten Lösung abgebrochen, um akzeptable Rechenzeiten zu erzielen. Heuristische Ansätze verzichten generell darauf, den Beweis der Optimalität zu führen. Ihnen geht es vielmehr darum, in kurzer Zeit möglichst gute zulässige Lösungen zu liefern.

In diesem Bereich sind zunächst Verfahren zu nennen, die eine FAP Lösung geringfügig verändern, um sie an die neuen äußeren Bedingungen anzupassen. Von den bereits erwähnten Arbeiten stellen z.B. [Berge and Hopperstad, 1993] Heuristiken vor, die mehrfach ein Minimum Cost Flow Problem lösen oder DELPRO-Methode, die die Zuweisung von zwei Flugsequenzen tauscht. Neuere Arbeiten, die ebenfalls fertige FAP Lösungen anpassen, sind z.B. [Klincewicz and Rosenwein, 1995], [Yan and Young, 1996], [Talluri, 1996], [Jarrah et al., 2000], [Sharma et al., 2000].

In der Dissertation von Radicke [Radicke, 1994] wurden heuristische Algorithmen für das FAP entwickelt, die Kreise im Graphen suchen und die Zuweisung der Flugzeugtypen auf diesen Kreisen verändern. Zunächst wird der Algorithmus für zwei Flugzeugtypen vorgestellt, der dann auf mehrere erweitert wird. Die Diplomarbeit von [Nitschke, 1997] untersucht mehrere exakte Verfahren für das Problem der Flottenzuweisung.

In Paderborn wurden im Rahmen von mehreren Projekten heuristische Verfahren für das FAP entwickelt. In der Diplomarbeit von [Grothklags, 2000] sind Simulated Annealing Algorithmen vorgestellt worden, die zur Zeit im industriellen Einsatz bei Kunden der Lufthansa Systems sind. Weitere Publikationen über das Paderborner FAP System sind: [Götz et al., 1999, Götz et al., 2004]. Eine genauere Beschreibung des Systems ist im Abschnitt 3.4.3 zu finden.

In [Grothklags, 2003] wird ein MIP-basierter Ansatz vorgestellt, der akzeptable Laufzeiten auch bei größeren FAP Datensätze liefert. Außerdem wird auch ein exakter Ansatz erarbeitet,

der eine Kombination aus dem *time-space network* Modell mit dem *connection-network* Modell darstellt, um die Phasen des Fleet Assignment und Aircraft Rotation zu verbinden. In der nach Fleet Assignment folgenden Phase des Aircraft Routing werden Flugzeugrotationen gebildet. Diese müssen ausreichend viele Wartungsereignisse beinhalten. Eine weitere wichtige Besonderheit besteht darin, dass die Mindestpausen zwischen zwei aufeinander folgenden Flügen (Bodenzeiten oder *ground times*) nicht unabhängig von den Eigenschaften der beiden Flüge berechnet werden können. Ein mathematisches Modell, das diese Anforderungen erfüllt, wurde in [Grothklags, 2003] entwickelt.

Sosnowska präsentiert in [Sosnowska, 1999], [Sosnowska and Rolim, 2000] Simulated Annealing und GRASP-Algorithmen, die auf kleineren Flugplänen getestet werden.

In den letzten Jahren wurden in der Literatur Erweiterungen des Grundproblems untersucht:

In der Arbeit von Brian Rexing [Rexing, 1997] werden die im klassischen FAP Modell festen Abflugzeiten der Flüge als variabel modelliert. Es werden alternative Abflugzeiten definiert. Dieser neue Freiheitsgrad erlaubt es, durch relativ geringe Änderungen im Flugplan die Zielfunktion im FAP zu verbessern.

Die amerikanische Firma SABRE Inc., ein Anbieter für Softwarelösungen für Fluggesellschaften, entwickelte zusammen mit der Georgia Institute of Technology in Atlanta, USA ein sog. O&D Fleet Assignment System [Jacobs et al., 2000b], [Jacobs and Günther, 2000], [Jacobs et al., 2000a]. Das mathematische Modell für das FAP wird periodisch mit approximierten Werten für die erzielbaren Profits auf Flügen im Netzwerk aktualisiert. Das System berücksichtigt somit durch die implizite Behandlung des Passagierflusses die im Flugnetz auftretenden Netzwerkeffekte. Es werden allerdings keine Ergebnisse über die Laufzeiten oder die Qualität der berechneten Lösungen veröffentlicht.

In der Dissertation von Manoj Lohatepanont [Lohatepanont, 2002] werden ebenfalls mehrere Erweiterungen des Fleet Assignment Modells vorgestellt. Zum einen wird ein Itinerary Based Fleet Assignment Modell (IFAM) entwickelt. Dieses Modell berücksichtigt durch Hinzunahme von Passagierflüssen Netzwerkeffekte im Flugnetz. Das IFAM System wird in einer Simulationsumgebung betrieben und zur Lösung von Tagesproblemen einiger nordamerikanischer Fluggesellschaften eingesetzt. Die Leistungsfähigkeit erreicht nicht den gewünschten Grad, sodass aufwendige Analysen zur Clusterung des Flugnetzes durchgeführt werden müssen. Europäische Flugplanung erfolgt typischerweise auf Wochenbasis bzw. sogar für einen Zeitraum von 4 - 6 Wochen in der taktischen Planungsphase. Daraus resultierende Modelle sind entsprechend größer und komplexer. Ein weiteres Modell in der Arbeit ist ein integriertes Modell für Netzwerkentwurf und Itinerary Based Fleet Assignment. Die Laufzeiten werden mit 20 Stunden auf einem parallelen System mit 6 Prozessoren für ein Problem mit 2000 Flügen angegeben. Es besteht also ein großer Bedarf an der Reduktion der Laufzeiten für den Einsatz in der Praxis.

3.4.3 Ein heuristischer Algorithmus für das Problem der Flottenzuweisung

Der Simulated Annealing Algorithmus startet mit einer (beliebigen) initialen Lösung. Zuerst erfolgt eine Berechnung der Starttemperatur, durch die bei klassischem SA die Unabhängigkeit des Verfahrens von der Wahl der Initiallösung sichergestellt werden soll. Von diesem Punkt des Lösungsraumes aus betrachtet und bewertet der Algorithmus eine benach-

Algorithmus 14 Simulated Annealing Algorithmus

1: S = initiale Lösung
2: T = Starttemperatur
3: **repeat**
4: **repeat**
5: S_{new} = Nachbarlösung von S
6: $\Delta = f(S_{new})$ - $f(S)$
7: **if** ($\Delta < 0$ oder $r < e^{-\frac{\Delta}{T}}$) **then**
8: $S = S_{new}$
9: **until** Gleichgewichtszustand erreicht
10: **until** keine Verbesserungen mehr erreichbar

barte Lösung. Hat diese eine *bessere* Bewertung, so wird sie zur neuen aktuellen Lösung. Eine *schlechtere* Lösung kann auch zur neuen aktuellen werden, dies geschieht mit einer gewissen Wahrscheinlichkeit. Diese Wahrscheinlichkeit ist durch das Akzeptanzkriterium gegeben und hängt bei den meisten SA-Varianten von dem Maß der Verschlechterung und von der aktuellen Temperatur (Steuerungsparameter) ab.

Dieser Schritt wird iteriert, wobei die Temperatur abgesenkt wird, sodass ein Übergang zur schlechteren Lösung immer unwahrscheinlicher wird. Am Ende wird die beste gefundene Lösung zum Ergebnis der Optimierung.

Der sequentielle Algorithmus für die Flottenzuweisung wurde von Sven Grothklags in [Grothklags, 2000] entwickelt. Eine Software-Bibliothek für sequentielles und paralleles Simulated Annealing wurde in der Diplomarbeit [Kliewer and Unruh, 1998] entwickelt und in vielen Anwendungen, unter anderem in der Flottenzuweisung, eingesetzt.

Die Anwendung des Simulated Annealing Algorithmus in der Flottenzuweisung basiert in der Definition der geeigneten Struktur der Nachbarschaft für den vorliegenden Lösungsraum. Die Komplexität und die Zahl der unterschiedlichen Restriktionen erschwert die Suche nach zulässigen Lösungen, die sich nur geringfügig von der aktuellen Lösung unterscheiden. Für das Problem wurden zwei Nachbarschaften definiert, die in der Abbildung 3.8 schematisch dargestellt sind.

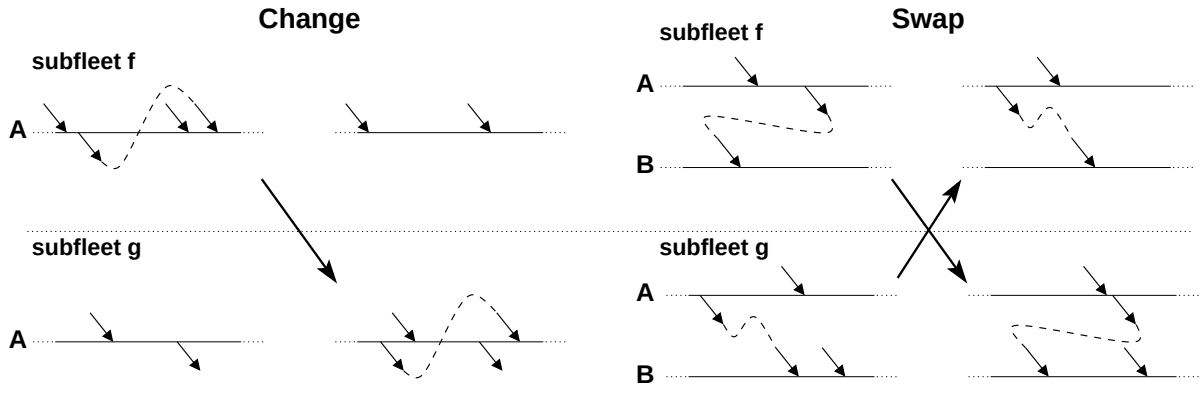


Abbildung 3.8: Nachbarschaftsdefinition für den Simulated Annealing Algorithmus

Operationen change und swap Diese zwei Operationen sind in der Abbildung 3.8 zu finden. Die Aufgabe besteht darin, eine neue Lösung zu finden, die sowohl die Balanciertheit des Flugplans sicherstellt als auch keine zusätzlichen Flugzeuge benötigt. Gesucht werden Sequenzen von Flügen (Legsequenzen) $(l_0, ..., l_k)$, die einem Flugzeugtypen zugewiesen sind und der Ankuftsflughafen eines Fluges dem Startflughafen des nächsten entspricht. Die *change*-Operation verändert die Zuweisung bei einer Sequenz, die am Flughafen A beginnt und auch dort endet. Die Zuweisung dieser Sequenz wird in der Abbildung von f zu g verändert. Die *swap*-Operation sucht nach zwei Sequenzen, die beide auf dem Flughafen A starten und auf dem Flughafen B enden. Die Zuweisung dieser zwei Legsequenzen wird anschließend getauscht.

Die Effizienz der Operationen change und swap bilden den Schlüssel für die Anwendbarkeit des Simulated Annealing Algorithmus in der Flottenzuweisung. Mehrere hundert Operationen pro Sekunde werden ausgeführt, sodass das Verfahren schnell konvergieren kann. Die Qualität der gefundenen Lösungen kann angegeben werden, indem ein Vergleich mit einer oberen Schranke durchgeführt wird. Diese oberen Schranken werden von einem Branch-and-cut Algorithmus von CPLEX geliefert.

3.4.4 Effizienz der Verfahren

Waren die auf Simulated Annealing basierenden heuristischen Methoden den exakten in der Laufzeit weit überlegen, so schrumpfte dieser Vorteil in den letzten Jahren. Fortschritt im Bereich der exakten Methoden zum Lösen gemischt-ganzzahliger linearer Probleme beschleunigte die Branch-and-cut Algorithmen von CPLEX enorm. Zusammen mit intensiven Preprocessing Techniken machte es dieser Fortschritt möglich, auch große Probleme aus der Praxis der Flugplanung zu lösen. Die wichtigsten Techniken sind u.a.:

- Durch eine Zusammenfassung der Knoten des Time-Space Netzwerks erreicht man eine signifikante Reduktion der Größe des mathematischen Modells.
- Die Definition einer Wartefunktion auf den Flughäfen und die daraus resultierenden Inseln spiegeln die besondere Struktur des Flugpläne wider. Durch getrennte Behandlung der Inseln im mathematischen Modell reduziert sich die Modellgröße noch einmal beträchtlich.
- Die SOS-Branching Entscheidung (special ordered sets) ergibt besser ausbalancierte Branch-and-bound Bäume und somit insgesamt kürzere Laufzeiten.
- Die Auswahl der Branching Variable in einem Knoten kann nach sog. Prioritäten getroffen werden. Durch eine nach der Zielfunktion ausgerichtete Definition der Prioritäten wird erreicht, dass in den höheren Branch-and-bound Ebenen die wichtigen Variablen zuerst betrachtet werden.

Einen Laufzeitvergleich der drei vorgestellten Verfahren sehen wir im Bild 3.9. Auf der x-Achse ist die Problemgröße in der Anzahl der Flüge und der Flugzeugtypen angegeben. Die Qualität der Lösungen ist in der Tabelle 3.1 gegenübergestellt worden.

In unseren weiteren Untersuchungen benutzen wir den Simulated Annealing Algorithmus für die Flottenzuweisung.

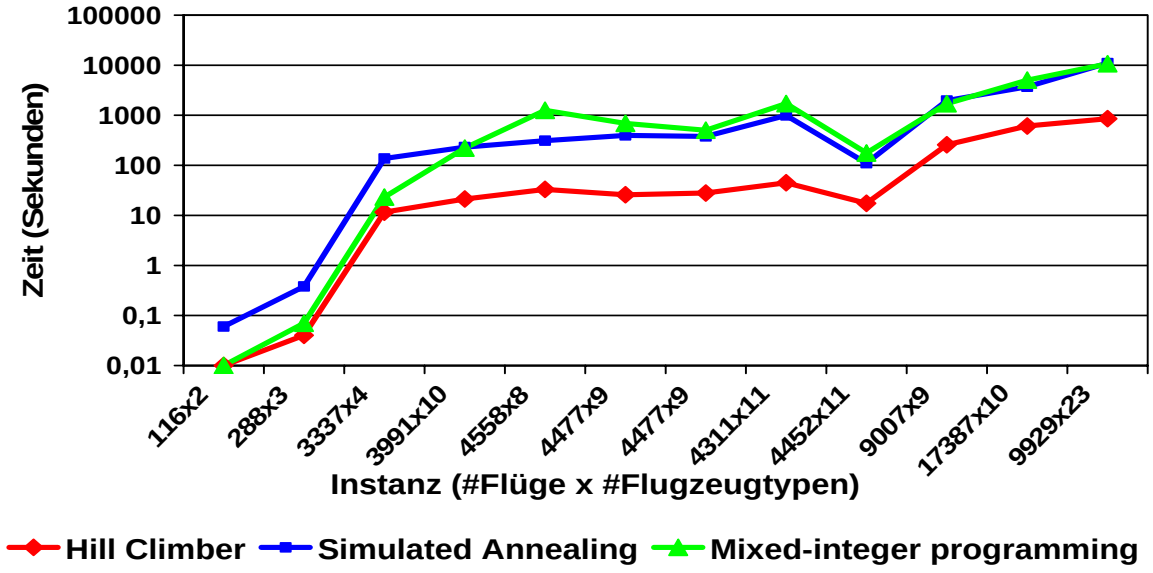


Abbildung 3.9: Vergleich der SA- und MIP-basierter Verfahren für das FAP

Algorithmus	Hill Climber(HC)	Simulated Annealing (SA)	Mixed-Integer (MIP)
⌀ Lösungsqualität	98,5%	99,7%	ca. 99,9%

Tabelle 3.1: Vergleich der Lösungsqualitäten für das Fleet Assingnment Problem

3.5 Marktmodellierung und Flottenzuweisung

Das im Abschnitt 3.4 beschriebene Modell für die Flottenzuweisung (*Fleet Assignment*) benötigt neben den Modellrestriktionen die eigentliche Zielfunktion, die es zu optimieren gilt. Wir erläutern im Folgenden wie diese Zielfunktion vom Marktmodell aufgebaut und berechnet wird.

3.5.1 Zielfunktion und Netzwerkeffekte

Zunächst existiert im Planungssystem eine zulässige Startlösung, die z.B. die Lösung aus der letzten Planungsperiode darstellt. Diese Lösung beinhaltet insbesondere eine gültige Flottenzuweisung für jeden Flug (*Leg*). Diese Zuweisung definiert die zur Verfügung stehenden Kapazitäten im Flugnetzwerk. Das Marktmodell kann eine Profitabilitätsbewertung des Flugplans vornehmen (siehe Abschnitt 3.2). Der dazu berechnete Passagierfluss definiert im Wesentlichen die voraussichtlichen Erträge, die eingesetzten Flugzeugtypen (Flottenzuweisung) bestimmen die anfallenden Kosten.

Die Planungsphase der Flottenzuweisung bekommt einen neuen Freiheitsgrad: Für jede Flugstrecke (*Leg*) stehen mehrere Flugzeugtypen zur Auswahl. Die Alternativen unterscheiden sich im wesentlichen durch unterschiedliche Kosten und Kapazitäten (Anzahl der Plätze). Die Kosten und Erträge müssen vom Marktmodell zur Verfügung gestellt werden.

Bemerkung: Die Berechnung der Zielfunktion für die Flottenzuweisung erfolgt *rein lokal*. Das bedeutet, dass bei einer Änderung der Kapazität auf einem Flug die Berechnung des Gewinns ausschließlich auf diesem Flug vorgenommen wird. Es wird lokal entschieden, welche Passagiere einen Platz auf dem Flug bekommen und welche abgewiesen werden (*local spill*). Insbesondere werden auftretende Netzwerkeffekte nicht berücksichtigt.

Das folgende Beispiel soll die Vorgehensweise des Marktmodells bei der Aufstellung dieser Daten verdeutlichen.

In der Abbildung 3.10 ist ein Netzwerk mit zwei kapazitätsbeschränkten Flügen dargestellt. Es sind drei Passagierströme vorhanden: LHR-BOS mit 200 Passagieren und einem Ertrag von 150 €, TXL-BOS mit 100 Passagieren und 200 € als Ertragswert, sowie 50 PAD-ATL Passagiere mit einem Ertragswert von 100 €. Die Flottenzuweisung ist bei der Anfangslösung wie folgt: FRA-JFK mit dem Typ A bei einer Kapazität von 80 Plätzen, JFK-BOS wird vom Typ C mit 150 Plätzen bedient. Das Marktmodell berechnet den Passagierfluss wie im oberen Teil der Abbildung 3.10 dargestellt: 120 – 30 – 50 Passagiere mit einem Gesamtprofit von 29.000 € (die Flugkosten werden in diesem Beispiel außer Acht gelassen).

Für die erste Alternative wird auf dem Flug FRA-JFK der Typ B mit 50 Plätzen eingesetzt. Im ersten Schritt werden die zwei auf dem Flug vorhandenen Passagierströme miteinander verglichen. Die Passagiere TXL-BOS bekommen wegen der höheren Erträge (200 € gegen 100 €) den Vorzug vor den PAD-ATL Passagieren. Es werden 50 TXL-BOS und keine PAD-ATL Passagiere transportiert. Somit erhält der Flug FRA-JFK einen Gewinnwert von 10.000 €. Das restliche Netzwerk wird nicht betrachtet und demzufolge bleiben die Passagierströme unverändert.

		Typ	Kapazität	Gewinn
FRA	JFK	**A**	**80**	8.000 €
FRA	JFK	B	50	10.000 €
JFK	BOS	**C**	**150**	21.000 €
JFK	BOS	D	200	35.000 €

Tabelle 3.2: Beispiel für die Berechnung der Zielfunktion in der Flottenzuweisung

Bei der zweiten Alternative setzt man den Typ D auf dem Flug JFK-BOS mit 200 Plätzen ein. In diesem Fall wird nach der Greedy-Strategie den höherwertigen Passagieren TXL-BOS Vorrang vor den LHR-BOS Passagieren gewährt. Es werden jeweils 100 Passagiere transportiert. Der Gesamtgewinn auf dem Flug JFK-BOS beträgt 35.000 €.

In der Tabelle 3.2 ist als Ergebnis dieser Berechnung die Zielfunktion für die Flottenzuweisung dargestellt. Die Aufgabe der Flottenzuweisung besteht darin, unter Berücksichtigung der Restriktionen die gewinnmaximale Zuordnung der Typen zu Flügen zu finden (siehe auch Abschnitt 3.4).

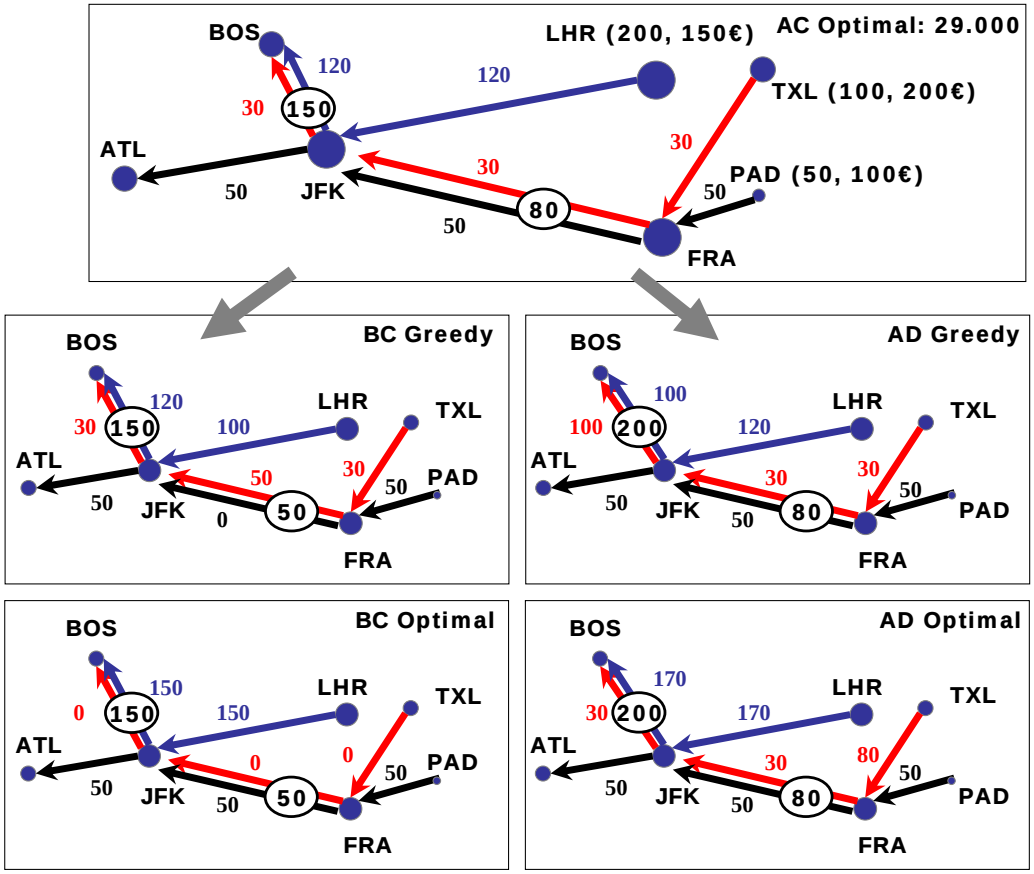


Abbildung 3.10: Beispiel für Netzwerkeffekte

Die beiden unteren Bilder in der Abbildung 3.10 zeigen eine optimale Aufteilung der Passagierströme aus der globalen Netzwerksicht. Die lokale Sicht bei der Generierung der Ziel-

funktion für die Flottenzuweisung erzeugt insbesondere keine gültigen Passagierflüsse sondern lediglich lokale Schätzungen der Gewinne.

Fazit: Die Ungenauigkeit der Zielfunktion und die nicht berücksichtigten Netzwerkeffekte stellen eine Hürde dar und erlauben keine zufriedenstellende Behandlung der beiden Planungsphasen der Marktmodellierung und der Flottenzuweisung. In den nächsten Abschnitten stellen wir mehrere Möglichkeiten dar, diese beiden Planungsphasen aufeinander abzustimmen und insgesamt zu einer besseren Lösung zu kommen.

3.5.2 Beschreibung der ersten Integrationsstrategie

Als erste Möglichkeit zur Abstimmung der Planungsphasen der Marktmodellierung und der Flottenzuweisung wird eine periodische Kommunikation zwischen den Planungsschritten untersucht.

Wie im vorhergehenden Abschnitt 3.5.1 beschrieben, erfolgt zunächst eine initiale Berechnung, deren Ergebnis die Zielfunktion für die Flottenzuweisung darstellt.

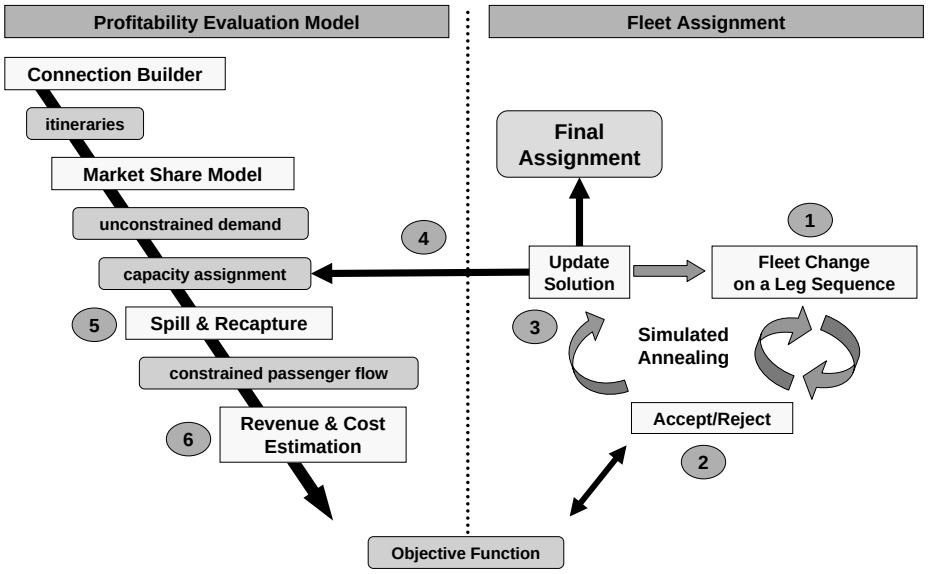


Abbildung 3.11: Kopplung der Marktmodellierung und der Flottenzuweisung

Die Abbildung 3.11 und der Algorithmus 15 stellen die von uns untersuchte Kopplung zwischen dem Maktmodell und der Flottenzuweisung dar.

Nachdem zunächst eine initiale Stufe der Marktmodellierung durchgeführt wurde (Algorithmus 15:1-5), beginnt die Optimierung der Flottenzuweisung durch den Simulated Annealing Algorithmus. Die Kommunikation findet jeweils nach einer abgeschlossenen Temperaturstufe des Simulated Annealing Algorithmus statt. Die empfangene Lösung wird im Marktmodell neu bewertet, indem die neuen Kapazitäten auf den Flügen bei der Berechnung des Passagierflusses berücksichtigt werden. Da die Lösungen aus der Flottenzuweisung keine Flugzeiten verändern, sind eine erneute Berechnung der Reiserouten und die Aufteilung der Passagiere nicht notwendig. Die Kosten- und Ertragsrechnung mit der empfangenen Lösung bildet

Algorithmus 15 Erste Integrationsstrategie

```
1: Aufbau der Reiserouten (connection builder)
2: Verteilung der Marktnachfrage (market share)
3: Berücksichtigung der Kapazitäten (spill and recapture)
4: Kosten- und Ertragsrechnung
5: Zielfunktion für die Flottenzuweisung
6: repeat
7:    repeat
8:       Änderung der Zuweisung bei einer Legfolge (Schritt 1)
9:       Neue Lösung akzeptieren oder verwerfen (Schritt 2)
10:      Übergang durchführen (Schritt 3)
11:   until Temperaturstufe beendet
12:   Lösung an das Marktmodell schicken (Schritt 4)
13:   Berücksichtigung der Kapazitäten der empfangenen Lösung (Schritt 5)
14:   Neue Kosten- und Ertragsrechnung (Schritt 6)
15:   Geänderte Zielfunktion an die Flottenzuweisung schicken
16: until Anzahl der Kommunikationsrunden
17: Beste Lösung als Ergebnis ausgeben
```

erneut die Grundlage für die Zielfunktion der Flottenzuweisung. Diese wird an das Fleet Assignment System geschickt und auf die nächste Kommunikationsrunde gewartet.

Die Häufigkeit der Kommunikation und die Anzahl der Kommunikationsrunden ist veränderbar:

- *Beginn der Kommunikation:* Nach Erreichen einer vorgegebenen Akzeptanzrate im Simulated Annealing Algorithmus. Dadurch kann z.B. vorgegeben werden, dass in der Anfangsphase der Optimierung, in der die Lösung noch sehr stark verändert wird, keine Kommunikation stattfinden soll. Der Standardwert für die Akzeptanzrate liegt bei 20%.

- *Häufigkeit der Kommunikation:* Anzahl der Temperaturstufen zwischen zwei Kommunikationsphasen. Das Kommunikationsmuster kann auch dynamisch verändert werden. In der Anfangsphase kann z.B. nach jeder 10. Temperaturstufe kommuniziert werden, später wird immer häufiger kommuniziert, bis am Ende der Optimierung nach jeder Stufe die Lösung verschickt wird.

- *Schwelle für die Kommunikation:* Die Kommunikation soll erst stattfinden, wenn eine vorgegebene Anzahl von Flügen eine andere Flotte zugewiesen bekommen hat. Dadurch wird erreicht, dass eine zeitintensive Kommunikation nicht für ganz wenige Veränderungen in der Flottenzuweisung angestoßen wird.

- *Anzahl der Kommunikationsrunden:* Die Anzahl kann vom Systembenutzer vorgegeben werden. Damit kann erreicht werden, dass für bestimmte Analysen eine schnelle Antwort geliefert wird und nicht unbedingt die vollständige Konvergenz des Verfahrens abgewartet werden muss.

3.5.3 Analyse des Verfahrens

Als Haupvorteil dieser Kopplung kann die bessere Berücksichtigung der Netzwerkeffekte genannt werden. Diese Effekte werden vom Marktmodell berücksichtigt und explizit modelliert. Durch die angepasste Zielfunktion kann das Fleet Assignment System die Auswirkung der Flottenänderung im Netzwerk erkennen und durch weitere Optimierung darauf reagieren. Die Ergebnisse der Experimente mit dem neuen System werden im Abschnitt 4.2 präsentiert.

Die Nachteile des Verfahrens liegen hauptsächlich in der zeitintensiven Kommunikation zwischen dem Marktmodell und dem Fleet Assignment System. Neben der Übermittlung der neuen Flottenzuweisung und anschließend der neuen Zielfunktion muss im Marktmodell selbst die entsprechende Neuberechnung durchgeführt werden. Typischerweise werden ca. 10-20% der Flüge mit einer anderen Flotte geflogen und dementsprechend müssen Passagiere auf sämtlichen Reiserouten, die diese Flüge benutzen, neu bewertet werden. Die im nächsten Abschnitt vorgestellte Vorgehensweise versucht, einen Kompromiss zwischen der Genauigkeit der Voraussagen über den Passagierfluss und der Berechnungsgeschwindigkeit zu finden. Wir möchten nun auf diese zweite Integrationsstrategie genauer eingehen.

3.6 Modellierung des Passagierflusses

Die zweite Strategie zur Kopplung der Marktmodellierung und der Flottenzuweisung führt eine neue Komponente ein, die zwischen den beiden Planungsphasen eingesetzt wird. Der Passagierfluss im Netzwerk wird dabei als ein lineares Mehrgüterflussproblem aufgefasst. Die Lösung dieses Problems liefert Informationen, die die Optimierung in der Flottenzuweisung steuern. Wir beschreiben zunächst das Modell des Passagierflusses außerhalb des Marktmodells und gehen danach auf die besonderen Integrationsaspekte ein.

3.6.1 Modell des Passagierflusses

Die Verteilung der Passagiere auf die Flüge des Flugnetzwerks kann mit einem Netzwerkfluss modelliert werden. Das *Passenger Flow Modell* (**PFM**) bildet den Passagierfluss im Netzwerk als ein Mehrgüterflussproblem (*multicommodity flow*) ab. Die Güter entsprechen den Passagieren in den untersuchten Märkten (Reiseverbindungen zwischen zwei Städten). Die Kanten des Netzwerks werden von den Flügen gebildet. Die Kapazitäten sind durch die Kapazität der eingesetzten Flugzeuge definiert. Gesucht ist ein gültiger Fluss im Netzwerk mit dem maximalen Gewinn als Zielfunktion. An dieser Stelle verweisen wir auf einen wichtigen Unterschied zu Modellen des Passagierverhaltens hin: die Passagiere entscheiden sich nicht für die für sie günstigste Verbindung. Es wird vielmehr angenommen, dass die Fluggesellschaft durch Steuerung der Buchungen die Passagiere auf die Verbindungen bringen kann, die ihren Gewinn maximieren.

Modelldefinition

Bei der Definition der Transportgüter können unterschiedliche Detaillierungsgrade gewählt werden. Das Städtepaar definiert immer eine *commodity*. Des Weiteren können zusätzlich berücksichtigt werden: Reiseroute (*itinerary*), Klasse (Economy, Business, First), Buchungsklasse (*fare*), Punkt des Ticketverkaufs (*point of sale*).

Wir definieren an dieser Stelle das *Passenger Flow Modell* für die gröbste Detaillierungsstufe: auf ODI Basis (*origin, destination, itinerary*).

Eingabedaten und Parameter

$\mathscr{A}$	Menge der Flughäfen
$\mathscr{E}$	Flugkanten des Netzwerks
$OD \subseteq \mathscr{A}^2$	Menge aller Passagier-Märkte
d_{od}	Geschätzte Anzahl der Passagiere im Markt od
P^{od}	Mögliche Reiserouten für Passagiere aus dem Markt od
$(\delta_{lp} = 1) \Leftrightarrow$	Pfad p benutzt die Kante $l \in \mathscr{E}$
f^{od}	Ertrag pro Passagier aus dem Markt od
c_l	Kapazität der Kante l

Entscheidungsvariablen

x_p^{od} Anzahl der Passagiere aus dem Markt od auf dem Pfad $p \in P^{od}$

$$
\begin{array}{lrcll}
\text{Maximize} & \sum_{od} \sum_{p \in P^{od}} f_p^{od} x_p^{od} & & & \\
\forall od \in OD: & \sum_{p \in P^{od}} x_p^{od} & \leq & d_{od} & (\text{PFM1}) \\
\forall l \in \mathscr{E}: & \sum_{od} \sum_{p \in P^{od}} x_p^{od} \delta_{lp} & \leq & c_l & (\text{PFM2}) \\
\forall od \in OD, \forall \text{p} \in P^{od}: & x_p^{od} & \geq & 0 & (\text{PFM3})
\end{array}
$$

Die Zielfunktion maximiert den voraussichtlichen Gewinn der Fluggesellschaft. Die Restriktionen (PFM1) stellen sicher, dass auf allen möglichen Reiserouten, die im Markt od liegen (die Städte o und d verbinden), nicht mehr Passagiere transportiert werden als geschätzt wurde. Die Ungleichungen (PFM2) erzwingen, dass die Kapazität auf den Flügen nicht überschritten wird. Die Entscheidungsvariablen x_p^{od} werden nicht als ganzzahlig vorausgesetzt, da es sich bei den Passagierzahlen um Schätzungen handelt. Wegen einer inhärenten Ungenauigkeit der Prognosen der Passagierzahlen wird an dieser Stelle auf die Ganzzahligkeit verzichtet und mit fraktionalen Flüssen gearbeitet.

Literatur

Lineare Passagierflussmodelle werden in der Literatur für unterschiedliche Zwecke benutzt [Glover et al., 1982], [Phillips et al., 1991], [Farkas, 1996]. Neben einer Modellierung des Passagierflusses im Netzwerk werden ähnliche Modelle im Bereich des Ertragsmanagements (*Revenue Management*) dazu benutzt, Buchungsanfragen zu verarbeiten und über die Verfügbarkeit von Plätzen auf Flügen zu entscheiden [Boyd, 2002], [Boer et al., 2002], [Williamson, 1988], [Williamson, 1992](Seiten 56 - 59)

3.6.2 Beschreibung der zweiten Integrationsstrategie

Bei dieser Variante modellieren wir den Passagierfluss mit Hilfe des *Passenger Flow Modells* (*PFM*) und koppeln diese neue Komponente an das Fleet Assignment System.

Die ersten Schritte im Algorithmus 16 entsprechen der initialen Marktmodellierung, zusammen mit der ersten Vorgabe der Zielfunktion für die Flottenzuweisung. Anstatt an das

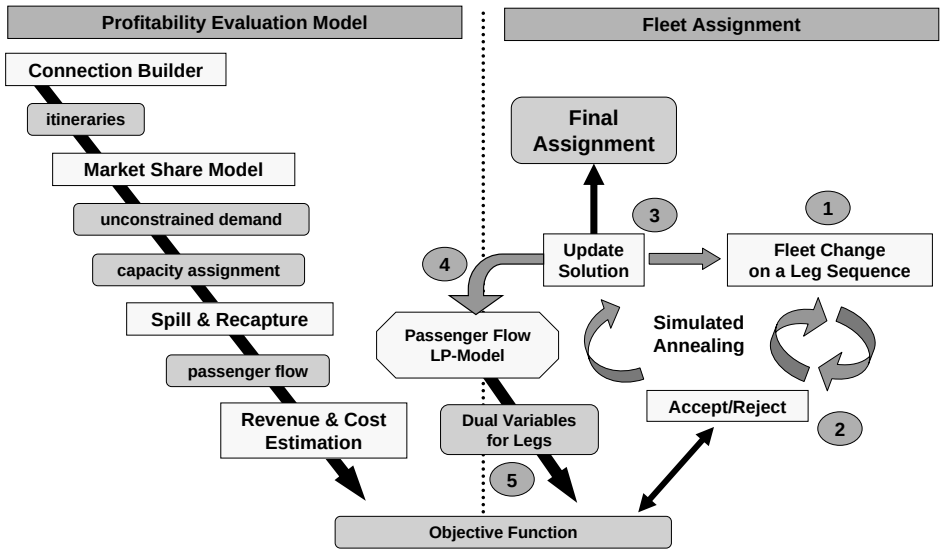


Abbildung 3.12: Passenger Flow Modell als Zwischenkomponente

Marktmodell werden in den Kommunikationsrunden die Lösungen der Flottenzuweisung an die neue Komponente geschickt.

Das Passenger Flow Modell berechnet neben einem Passagierfluss zusätzliche Informationen und schickt diese als neue Zielfunktion an die Flottenzuweisung.

Algorithmus 16 Zweite Integrationsstrategie

1: Aufbau der Reiserouten (*connection builder*)
2: Verteilung der Marktnachfrage (*market share*)
3: Berücksichtigung der Kapazitäten (*spill and recapture*)
4: Kosten- und Ertragsrechnung
5: Zielfunktion für die Flottenzuweisung
6: **repeat**
7: **repeat**
8: Änderung der Zuweisung bei einer Legfolge (Schritt 1)
9: Neue Lösung akzeptieren oder verwerfen (Schritt 2)
10: Übergang durchführen (Schritt 3)
11: **until** Temperaturstufe beendet
12: **Flottenzuweisung an das PFM schicken** (Schritt 4)
13: (Inkrementelle) Lösung des Passenger Flow Modells
14: Berechnung der dualen Variablen des Modells
15: **Geänderte Zielfunktion an die Flottenzuweisung schicken (Schritt 5)**
16: **until** Anzahl der Kommunikationsrunden
17: **Beste Lösung als Ergebnis ausgeben**

Integration: Die Kernidee besteht nun darin, für jede Netzwerkkante die dualen Variablen zu berechnen und diese der Flottenzuweisung zur Verfügung zu stellen. Die Nebenbe-

dingungen (PFM 2) begrenzen den Fluss auf den Kanten. Der Wert einer dualen Variable gibt für die Kante an, wie groß die Änderung der Zielfunktion bei einer Änderung der Kapazität um genau eine Einheit ist. Im Bereich Revenue Management werden die dualen Werte häufig als bid-prices benutzt.

$$\forall l \in \mathscr{E} : \sum_{od} \sum_{p \in P^{od}} x_p^{od} \delta_{lp} \leq c_l \quad \text{(PFM2)}$$

Kanten, deren Kapazität vom Passagierfluss nicht komplett ausgeschöpft wurde, haben einen dualen Wert von Null. Der Simulated Annealing Algorithmus für die Flottenzuweisung bekommt die dualen Werte aller Kanten im Netzwerk und erhält dadurch die Information, welche Kante einen Engpass darstellt. Die Zielfunktion wird entsprechend angepasst. Konkret wird der duale Wert bei den Koeffizienten der Flugzeugtypen aufaddiert, die eine höhere Kapazität besitzen als die aktuelle Zuweisung. Bei Flugzeugtypen mit geringerer Kapazität wird der Zielfunktionskoeffizient um den dualen Wert reduziert.

3.6.3 Analyse des Verfahrens

Die Änderung der Zielfunktion im Simulated Annealing Algorithmus soll bewirken, dass die Typzuweisung besser dem geschätzten Passagierfluss entspricht. Wie der Algorithmus auf die Anpassung der Zielfunktion reagiert, wird im Ergebnisteil dieser Arbeit (4.2) untersucht.

Der Vorteil der zweiten Integrationsstrategie im Vergleich zu der ersten ist die schnellere Berechnung des Flusses durch das *Passenger Flow Modell* (PFM). Außerdem steuert das PFM das Lösungsverfahren der Flottenzuweisung. Dadurch werden insbesondere die Netzwerkeffekte besser berücksichtigt. Die Kommunikationsstruktur lässt sich ebenfalls in Bezug auf Beginn, Häufigkeit und Anzahl der Kommunikationsrunden steuern.

3.7 Kopplung von Revenue Management und Flottenzuweisung

Die dritte Integrationsstrategie verbindet Systeme für Revenue Management (RM) und Flottenzuweisung. Die Motivation kommt aus der Tatsache, dass in einer Planungsphase, die zwischen drei und einem Monat vor Start des Flugplans liegt, bereits einige Buchungen im RM-System vorliegen und dadurch eine genauere Abschätzung der Gewinne möglich wird. Die kurzfristige Änderungsplanung der Flottenzuweisung kann entscheidend von dieser besseren Gewinnschätzung profitieren.

Wir vergleichen in diesem Abschnitt die Qualität der Prognosen in den jeweiligen Planungsschritten. Danach beschreiben wir die Kopplung der beiden Systeme.

3.7.1 Passagierprognosen

Das im Abschnitt 3.2 vorgestellte Marktmodell berechnet ca. ein halbes Jahr vor dem Start der nächsten Flugplanperiode Prognosen zu den Passagierzahlen im Netzwerk. Diese Prognosen werden zur Schätzung der erzielbaren Erträge herangezogen. Neben der Ertragsrechnung berücksichtigt das Marktmodell auch die Kosten für die Ausführung des Flugplans,

die hauptsächlich vom eingesetzten Flugzeugtyp abhängig sind. Die Zielsetzung der Flottenzuweisung besteht in der Maximierung der Gewinnfunktion unter Einhaltung sämtlicher operationeller Restriktionen.

Zur Schätzung der Erträge geht das Verfahren von Reiserouten (*itineraries*) aus, die keine Unterscheidung nach Klassen (First, Business, Economy) beinhaltet. Vielmehr wird ein Durchschnittsertrag für alle drei Klassen berechnet. Die Nachfrage-Prognosen werden lediglich als Mittelwerte verwendet, der Prognosefehler bleibt in diesem Fall unberücksichtigt.

Im O&D-Revenue-Management werden Steuer-Parameter zur Bestimmung der Verfügbarkeit von Flügen und zur Entscheidung von Buchungsanfragen berechnet. Im Gegensatz zur lokalen Steuerung berücksichtigt die O&D-Steuerung Verdrängungseffekte innerhalb des Flugnetzes (Netzwerkeffekte). Die Nachfrage-Prognose berücksichtigt nicht nur die Reiserouten, sondern auch die Buchungsklassen, die eine genauere Betrachtung als die Klassen (First, Business, Economy) erlaubt. Weiterhin werden auch Prognose-Fehler erfasst und in die Schätzung eingearbeitet.

Auch durch eine spätere Bereitstellung der Prognosen wird die Qualität dieser deutlich gesteigert. Mit einer viel detaillierteren und genaueren Ertragsschätzung für ein Flugnetzwerk kann die Lösung des Problems der Flottenzuweisung besser angegangen werden. Die Kapazität im Netzwerk wird besser an die erwartete Nachfrage angepasst. Auf der anderen Seite werden durch die kurzfristigere Planung keine umfangreichen Änderungen möglich sein. Diese hätten Auswirkungen auf weitere Planungsschritte (z.B. Crewplanung) und können relativ kurz vor Start des Flugplans nicht mehr realisiert werden.

3.7.2 Beschreibung der dritten Integrationsstrategie

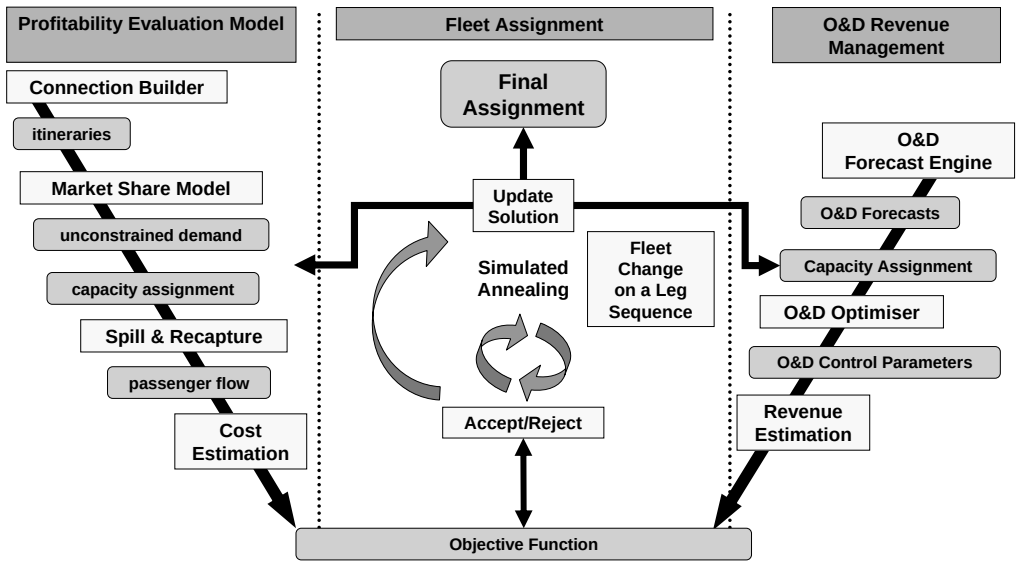


Abbildung 3.13: Einbeziehung von genaueren Profitprognosen im taktischen Fleet Assignment

Wie auch in den beiden bereits beschriebenen Integrationsstrategien erfolgt zunächst die initiale Bewertung des Ausgangsflugplans durch das Marktmodell. Die Kommunikation des

Systems für die Flottenzuweisung erfolgt in diesem Fall mit dem O&D Revenue Management System (siehe Abbildung 3.13).

Zunächst lädt das Revenue Management System Prognosen (einschließlich Prognose-Fehler) aus der Forecast Datenbank. Der O&D Optimizer berechnet daraus mit Hilfe der Methode *iterative prorated EMSR* die Steuerparameter (bid-prices) und übermittelt sie an die Gesamt-Revenue-Schätzung. Im letzten Schritt wird unter Berücksichtigung der aktuellen Flottenzuweisung eine Ertragsschätzung ermittelt. Diese Information wird dem Flottenzuweisungssystem zur Verfügung gestellt. Die Kostenberechnung erfolgt weiterhin auf der Grundlage der vom Marktmodell ermittelten Daten.

Die beschriebene dritte Integrationsstrategie wurde in einer Vorstudie gemeinsam mit Lufthansa Systems erarbeitet. Die ersten beiden Integrationsstrategien und die konzeptuellen Ergebnisse dieser Studie wurden in folgenden Arbeiten veröffentlicht:

- Weber K.; Sun J.; Sun Z.; Kliewer G.; Grothklags S.; Jung N. Systems integration for revenue-creating control processes. Journal of Revenue and Pricing Management, July 2003, vol. 2, no. 2, pp. 120-137(18) Henry Stewart Publications

- Georg Kliewer, Sven Grothklags, and Klaus Weber. Improving revenue by system integration and co-operative optimization. In Proceedings of the 43rd Annual Symposium of the Airline Group of the International Federation of Operational Research Societies (AGIFORS), Honolulu, Hawaii, USA, 2002.

- Georg Kliewer, Sven Grothklags, and Klaus Weber. Improving revenue by system integration and co-operative optimization. In Proceedings of the Reservations and Yield Management Study Group meeting, AGIFORS, Berlin, Germany, 2002.

3.8 Zusammenfassung

In diesem Kapitel haben wir uns mit der Integration der Planungsaufgabe der Flottenzuweisung beschäftigt. Das Verbesserungspotential wurde untersucht und die zeitlich benachbarten Planungsphasen beschrieben. In der Mittelfristplanung erfolgt die Integration mit der Aufgabe der Marktmodellierung. Dadurch werden die im Flugnetz auftretenden Netzwerkeffekte besser berücksichtigt. In der Kurzfristplanung werden wesentlich genauere Passagierprognosen und Ertragsschätzungen aus dem Revenue Management benutzt, um die Kapazitäten durch veränderte Flottenzuweisung besser auf den Passagierfluss anzupassen.

Insgesamt wurden im Rahmen dieser Arbeit drei Integrationsstrategien entwickelt, die zum Teil in der industriellen Praxis zum Einsatz kommen. Einige Vorschläge wurden von unseren Partnern noch nicht in den Planungssystemen umgesetzt, sodass weiteres Verbesserungspotential der Flugplanungsprozesse besteht.

— 4 —

Experimentelle Ergebnisse

4.1 Netzwerkentwurf

Im Rahmen dieser Arbeit wurde ein Optimierungssystem für die Aufgabe des Netzwerkentwurfs entwickelt. Das System ist modular aufgebaut und kann sowohl als ein exakter als auch ein heuristischer Planungsalgorithmus eingesetzt werden. Wir beschreiben zunächst den Grundabgabe des Systems und gehen dann genauer auf die einzelnen Komponenten ein.

Im Bild 4.1 ist das Gesamtsystem hierarchisch dargestellt. Auf der obersten Ebene befinden sich die beiden Hauptalgorithmen für das CNDP (capacitated network design problem): Branch-and-bound und Relax-and-cut. Beide basieren auf der Lagrange-Relaxation als untere Schranke in der Baumsuche. Der Relax-and-cut benutzt zusätzlich zwei Typen von gültigen Ungleichungen: die Überdeckungsungleichungen (*cover inequalities - cover cuts*) und die lokalen Schnitte (*local cuts*). Für die Lösung des Lagrange-Dualen Problems wird ein von uns implementiertes Subgradient-Verfahren eingesetzt. Als Alternative dazu haben wir einen Bundle-method Solver von Antonio Frangioni in das System eingebunden [Frangioni, 1997].

Die Systemkonfiguration mit dem Bundle-method Solver bildet die Standardvariante. Wir haben gültige Ungleichungen für das Netzwerkentwurfproblem untersucht und für den Relax-and-cut-Algorithmus implementiert. Dabei haben wir Algorithmen zum Finden von verletzten Ungleichungen und Methoden zur Verstärkung dieser implementiert (*lifting*). Die von der aktuellen Lösung im Suchbaum verletzten Ungleichungen werden dann mit neuen Lagrange-Multiplikatoren versehen und in der Lagrange-Zielfunktion berücksichtigt. Die Verwaltung der Ungleichungen wird dynamisch vorgenommen, d.h. dass die Ungleichungen in einem Gesamtwohl verwaltet und nur bei Bedarf berücksichtigt werden.

Der Aufbau dieses Kapitels ist wie folgt. Zunächst beschreiben wir die Eingabedaten für das Netzwerkentwurfproblem, den Datengenerator und die Eigenschaften der benutzten Benchmarks (Abschnitt 4.1.1). Danach führen wir eine Untersuchung der Komplexität der vorliegenden Datensätze durch und stellen erste Experimente dazu vor. Im nächsten Unterabschnitt beschreiben wir die Vergleichsmethodik für die entwickelten Algorithmen (Ab-

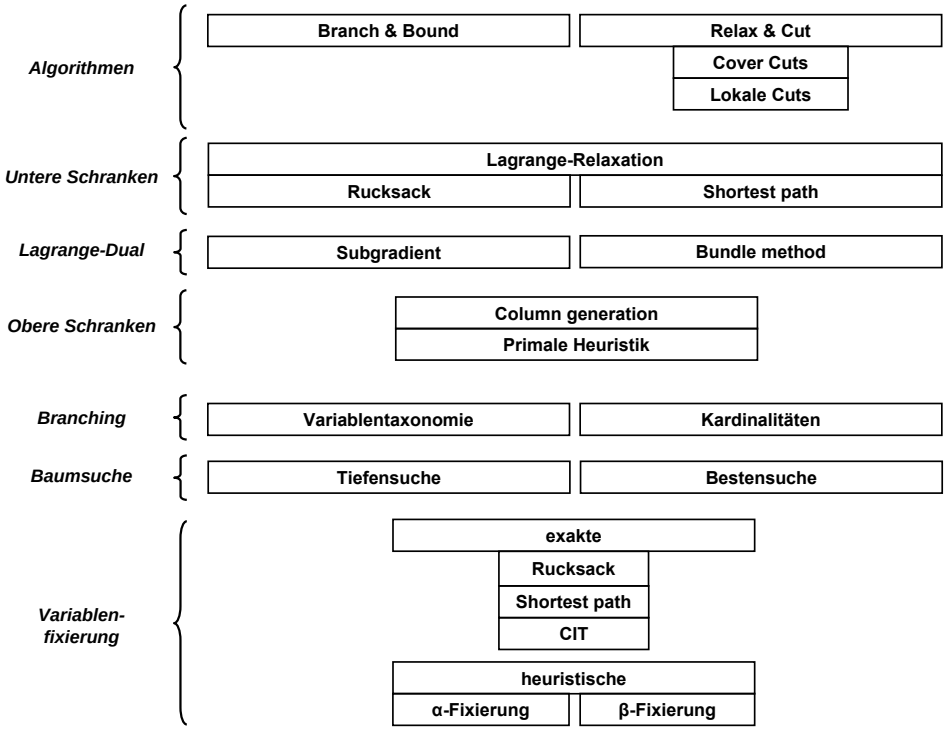


Abbildung 4.1: Überblick über die implementierten Algorithmen

schnitt 4.1.2). Danach kommen wir zu den Ergebnissen, die mit unserem System erreicht wurden. Wir vergleichen im Abschnitt 4.1.3 das System mit anderen Solvern und bewerten die Leistungsfähigkeit einzelner wichtiger Systemkomponenten (Abschnitt 4.1.4). Im letzten Unterabschnitt 4.1.5 fassen wir die Erkenntnisse aus den Experimenten zusammen.

Die meisten Experimente in diesem Kapitel wurden auf einem Pentium 4 Prozessor (3 GHz, 1 GByte Hauptspeicher) durchgeführt.

4.1.1 Benchmark-Daten

Die Bewertung der Leistungsfähigkeit der Algorithmen für Netzwerkentwurf wird anhand von Benchmarkdaten vorgenommen. Diese Benchmarks stellen eine Auswahl von Daten dar, die bereits in mehreren Veröffentlichungen für die Bewertung von anderen Algorithmen benutzt wurde. Dadurch wird eine Vergleichbarkeit der Ergebnisse sichergestellt. Im Folgenden beschreiben wir den Datengenerator und die Eigenschaften der Daten, die beeinflusst werden können. Danach betrachten wir die Größe der Datensätze, die von kleinen und sehr einfach lösbaren Instanzen bis zu relativ großen und nur mit großem Rechenaufwand lösbaren Datensätzen reicht. Eine Betrachtung der Schwierigkeit der Instanzen wird durch die Lösung der LP-Relaxationen mit Hilfe von CPLEX ermöglicht.

Beschreibung des Generators

Ein Generator für das Problem des Netzwerkentwurf wurde von dem kanadischen Wissenschaftler Bernard Gendron entwickelt und in mehreren Veröffentlichungen für Experimente

benutzt [Gendron and Crainic, 1994a], [Gendron and Crainic, 1996]. Es handelt sich dabei um ein Fortran-Programm, das eine Parameterdatei als Eingabe verarbeitet und eine Instanzdatei für Netzwerkentwurf generiert.

In Tabelle 4.1 ist eine mögliche Parameterdatei für den Generator abgebildet. Durch Vorgabe von Netzwerkeingenschaften können die Struktur und die Komplexität der Netzwerke beeinflusst werden.

Parameter	Wert	Bedeutung
nocca		Keine spezifischen Kapazitäten für Commodities
inseed	0	Initialisierung des Zufallszahlgenerators
ctight	3.0	Kapazitätswert (siehe Definition)
fixvar	0.02	Verhältnis der fixen zu variablen Kosten (siehe Definition)
noprll	10	Keine parallelen Kanten
scalex	6	Größe des Netzes in x-Richtung
scaley	2	Größe des Netzes in y-Richtung
commod	50	Anzahl der Commodities
addarc	50	Anzahl der Kanten
minsrc	1	Minimale Anzahl der Quelle-Knoten pro Commodity
maxsrc	1	Maximale Anzahl der Quelle-Knoten pro Commodity
minsnk	1	Minimale Anzahl der Senke-Knoten pro Commodity
maxsnk	1	Maximale Anzahl der Senke-Knoten pro Commodity
minfct	10	Minimale fixe Kosten für eine Kante
maxfct	100	Maximale fixe Kosten für eine Kante
mincst	5	Minimale Kosten für den Transport von Commodities
maxcst	50	Maximale Kosten für den Transport von Commodities
minsup	150	Minimaler Transportbedarf pro Commodity
maxsup	750	Maximaler Transportbedarf pro Commodity
mincap	100	Minimale Kapazität einer Kante
maxcap	500	Maximale Kapazität einer Kante
outfil	A3.dat	Name der Instanzdatei
end		

Tabelle 4.1: Eine Parameterdatei für den Datengenerator für den Netzwerkentwurf

Das Vorgehen des Generators kann durch den Algorithmus 17 beschrieben werden:

Algorithmus 17 Generator für den Netzwerkentwurf

1: Erzeuge ein zufälliges Netz (Knoten und Kanten), indem die vorgegebene Anzahl von Kanten zwischen Zufallsknoten gezogen wird,
2: Wähle zufällig Quellen und Senken für die Commodities,
3: Passe die Kapazitäten der Kanten dem Transportbedarf an (nach Vorgabe des Kapazitätswertes),
4: Passe das Verhältnis der fixen zu variablen Kosten an,
5: Ausgabe der Instanzdatei.

Die beiden wichtigsten Parameter für die Schwierigkeit einer Instanz sind neben der Größe des Netzwerks der Kapazitätswert C und das Verhältnis der fixen zu variablen Kosten F, die folgendermaßen definiert werden (dabei ist $T = \sum_{k \in \mathscr{C}} d^k$ als der Gesamttransportbedarf im Netzwerk definiert):

Definition 4.1 (Capacity ratio)

$$C = \frac{|\mathscr{A}|T}{\sum_{(i,j)\in\mathscr{A}} u_{ij}}$$

Der Kapazitätswert ist das Verhältnis des Transportbedarfs für alle Commodities zu der durchschnittlichen Kapazität einer Kante. Größere Kapazitätswerte definieren demzufolge ein engeres Netzwerk, in dem es schwieriger ist, den Transportbedarf zu befriedigen.

Definition 4.2 (Fixvar ratio)

$$F = \frac{|\mathscr{C}| \sum_{(i,j)\in\mathscr{A}} f_{ij}}{T \sum_{k\in\mathscr{C}} \sum_{(i,j)\in\mathscr{A}} c_{ij}^k}$$

Bei Werten von F nahe 0 sind die fixen Installationskosten für Kanten gering im Vergleich zu den Transportkosten. Bei größeren Werten von F steigt deren Bedeutung.

Wir untersuchen in diesem Kapitel unter anderem auch, wie die Laufzeit und Lösungsqualität von diesen Parametern abhängt.

Größe der Datensätze

Die Größe der Datensätze stellen wir zunächst nach Anzahl der Knoten, Kanten und Commodities gegenüber:

	Canad-R1	Canad-R2	PAD	Canad-C	PAD-S
Knoten	10	20	12-24	20-30	40-60
Kanten	35-83	120-318	50-440	230-700	500-2500
Commodities	10-50	40-200	50-160	40-400	100-500
Anzahl der Instanzen	72	81	41	31	54

Tabelle 4.2: Überblick über die Datensätze. (siehe auch 6.1–6.4 auf den Seiten 158–161)

Die Tabelle 4.2 gibt einen Überblick über die verwendeten Instanzen. Die Instanzen der Klassen **Canad-R1** und **PAD** lassen sich sowohl mit CPLEX als auch mit unseren beiden Solvern (**NDBB** und **NDBC**) exakt lösen. Die Instanzen der Klassen **PAD** und **CANAD-R2** bilden die Grundlage für die Vergleiche unserer exakter Verfahren mit CPLEX. Die Klasse **Canad-C** besteht aus relativ schweren Instanzen, die sich nicht optimal lösen lassen. Diese Klasse dient den Vergleichen der heuristischen Verfahren. Die Klasse **PAD-S** besteht aus den größten Instanzen unserer Benchmarks und zeigt die Grenzen der exakten Verfahren auf. Die heuristischen Verfahren liefern auf diesen Instanzen noch akzeptable Lösungen, deren Qualität sich aber nicht genau abschätzen lässt, weil gute untere Schranken nur sehr schwer berechnet werden können.

Die Abbildung 4.2 soll die Komplexität der Probleminstanzen grafisch veranschaulichen. Sie stellt auf der *x*-Achse die Anzahl der Knoten und auf der *y*-Achse die Anzahl der Kanten der Instanzen dar, wobei die Größe der Kreise der Anzahl der Commodities entspricht. Die zweite Grafik zeigt die Anzahl der Nichtnull-Elemente (*nonzeros*) in der MIP-Formulierung des Netzwerkentwurfproblems, die oft ein Indiz für die Komplexität der LP-Relaxation des Problem ist.

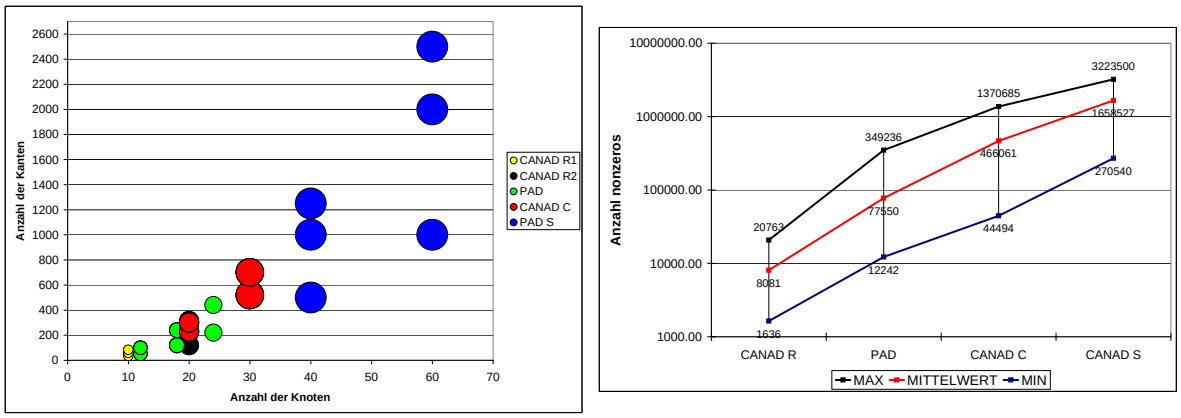


Abbildung 4.2: Größen der Benchmarkdaten

LP- und Lagrange-Relaxationen

In diesem Abschnitt betrachten wir Relaxationen für das Netzwerkentwurf Problem (CNDP). Die LP-Relaxation wird in einem LP-basierten Branch-and-bound Algorithmus einmal in der Wurzel des B&B-Baums gelöst und bestimmt oft ganz wesentlichen die Laufzeit des gesamten B&B-Algorithmus. In einem Lagrange-Relaxation Ansatz wird eine die Lagrange-Relaxation statt der LP-Relaxation gelöst. Es wird eine möglichst gute Annäherung an den Wert der LP-Relaxation gesucht.

Wir vergleichen im Folgenden die Laufzeiten unterschiedlicher Algorithmen, die die LP-Relaxation für Datensätze unterschiedlicher Größe lösen. Die Laufzeiten des Subgradient-Verfahrens und des Bundle-Verfahrens sowie die Qualität der berechneten Lagrange-Schranken werden miteinander verglichen.

	Best upper-lower	CPLEX weak LP	CPLEX strong LP	Lagrangean Subgradient	Lagrangean Bundle
	Ø gap in %	Ø gap in %	Ø gap in %	Ø gap in %	Ø gap in %
CANAD R1	optimal	17.80	2.98	5.30	2.47
CANAD R2	3.38	25.71	3.62	5.14	3.50
PAD	optimal	41.75	0.86	1.01	0.97
CANAD C	1.74	17.29	2.08	1.98	2.10
PAD S	16.17	-	-	17.27	17.22
		CPLEX time	CPLEX time	NDBB time	NDBC time
		Ø in sec	Ø in sec	Ø in sec	Ø in sec
CANAD R1		0.02	0.14	0.14	0.09
CANAD R2		2.80	267.99	1.70	1.01
PAD		0.08	3.97	1.24	0.74
CANAD C		4.88	1985.69	5.53	3.29
PAD S		-	-	36.34	12.06

Tabelle 4.3: Vergleich der unteren Schranken

In der Tabelle 4.3 ist in der oberen Hälfte die durchschnittliche Qualität der unteren Schranken für unterschiedliche Algorithmen angegeben. Die erste Spalte gibt die untersuchten Benchmarks an. In der zweiten Spalte ist der Abstand zwischen der besten bekannten unteren und der besten bekannten oberen Schranke angegeben. Die Klassen **CANAD-R1**

und **PAD** sind komplett optimal gelöst, d.h. dass der Abstand zwischen diesen beiden Schranken geschlossen wurde. Während für die Klassen **CANAD-R2** und **CANAD-C** eine relativ kleine Lücke und somit eine gute Lösungsqualität berechnet werden konnte (3.38% und 1.74%), ist für die Klasse PAD-S die Lücke relativ groß: 16.17%.

In der dritten Spalte ist die Qualität der schwachen LP-Schranke für das CNDP angegeben. Diese ist im Vergleich zu der starken LP-Schranke in der vierten Spalte sehr schlecht. Die schwache LP-Schranke kann in einem B&B-Algorithmus nicht sinnvoll verwendet werden.

In der fünften und sechsten Spalte der Tabelle 4.3 ist die Qualität der Lagrange-Schranken angegeben. Die von der Bundle-Methode gelieferten Schranken übertreffen in der Qualität die Schranken der Subgradient-Suche.

In der unteren Hälfte der Tabelle sind die durchschnittlichen Zeiten für die Berechnung der jeweiligen Schranke angegeben. Wir beobachten dabei, dass die Lagrange-Schranken deutlich schneller als die starke LP-Schranke berechnet werden können. Sie liefern dabei eine gute Abschätzung der LP-Schranke und eignen sich daher hervorragend für den B&B-Algorithmus. Dieser Vergleich liefert eine sehr gute Ausgangsbasis für die Leistungsfähigkeit des auf der Lagrange-Relaxation basierten B&B-Algorithmus. Für die Klasse der größten Instanzen **PAD-S** konnten die LP-Schranken aufgrund der Problemgröße nicht in einer vorgegebenen Zeit von 2 Stunden berechnet werden. Damit konnte der B&C-Algorithmus von CPLEX für diese Instanzen überhaupt keine zulässigen Lösungen liefern.

In der Abbildung 4.3 stellen wir die Konvergenz der Verfahren bei der Berechnung der unteren Schranke in der Wurzel des B&B-Baums dar (zwei Instanzen aus **PAD**: K1 und L1). Deutlich zu sehen ist die schnelle Konvergenz der beiden Lagrange-Schranken im Vergleich zu den LP-Schranken (dualer und primaler Simplex, sowie der Interior-Point Algorithmus CPLEX-baropt).

CPLEX-Experimente

Der auf der LP-Relaxation basierte Branch-and-cut Algorithmus von CPLEX wird in diesem Abschnitt auf der Benchmark **PAD** gestartet. Das Tuning der Parameter soll Aufschluss darüber geben, welche Strategien von CPLEX die besten Ergebnisse liefern.

CPLEX verfügt über eine Vielzahl von Einstellungen, die die Leistungsfähigkeit beeinflussen können. Wir beschränken uns in diesem Abschnitt allerdings auf einige wenige, die die größten Auswirkungen in unseren Experimenten hatten.

Folgende Parameter sind berücksichtigt worden:

- Heuristik für primale Lösungen
- Zusätzliche Ungleichungen (*Cuts*)
- Suchstrategie im B&C-Algorithmus (*Emphasis*)

Die Heuristik für die Suche nach primalen Lösungen kann ein- oder ausgeschaltet werden. Die Häufigkeit, mit der sie gestartet wird, wird von CPLEX automatisch ermittelt. Es gibt mehrere Typen von zusätzlichen Ungleichungen, die die Qualität der unteren Schranken während der B&B-Suche verbessern sollen. Auch hier können sie ein- oder ausgeschaltet werden. Bei der Suchstrategie können mehrere Ziele verfolgt werden. Es existieren zwei extreme Einstellungen, nämlich die vorrangige Suche nach gültigen ganzzahligen Lösungen

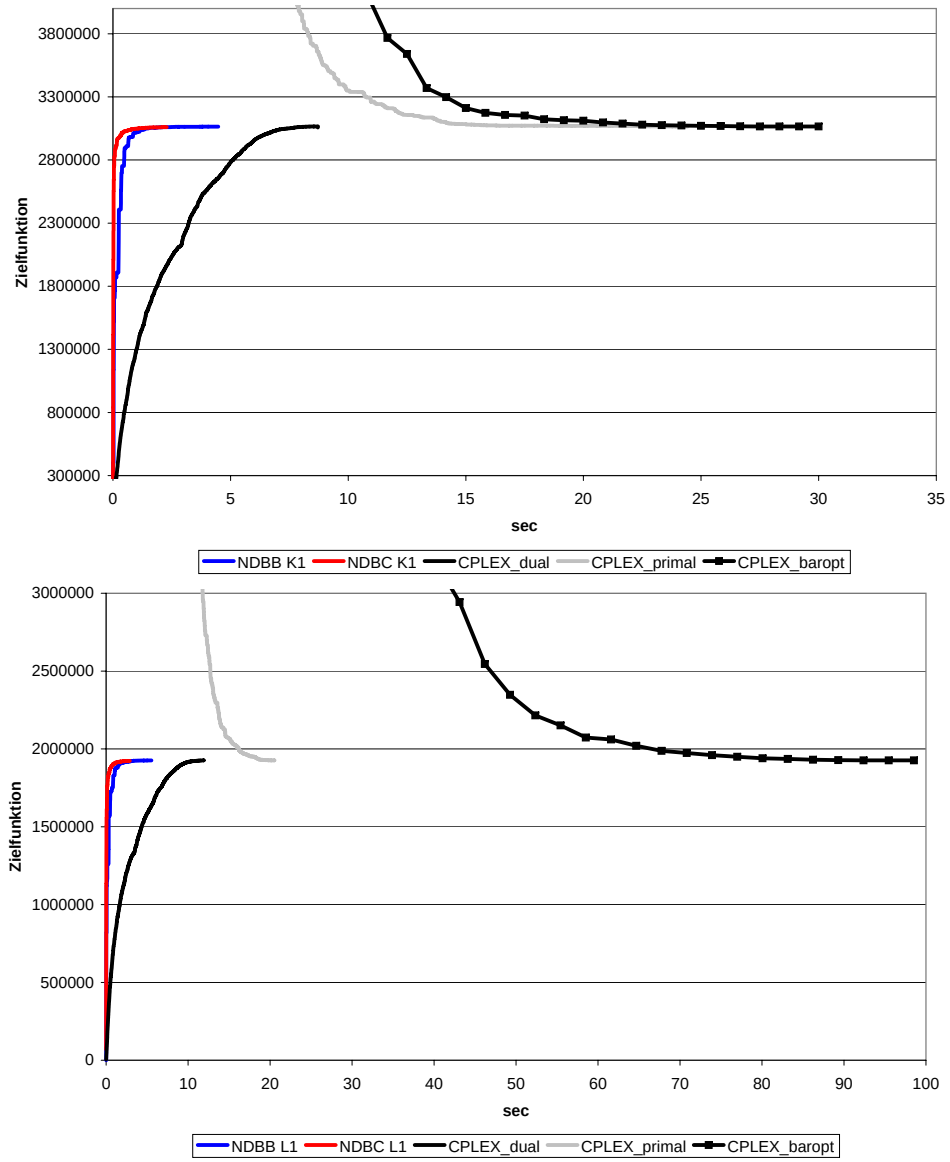


Abbildung 4.3: Konvergenz der Verfahren bei der Berechnung der unteren Schranke

(**EMPINT**) oder der Versuch, möglichst schnell die Optimalität der gefundenen Lösung nachzuweisen (**EMPOPT**). Die Balance zwischen diesen beiden Zielen wird mittels **EMPBAL** erreicht.

Im Vorgriff auf den nächsten Abschnitt nutzen wir in diesem Vergleich bereits das Konzept der Leistungsprofile. In der Tabelle 4.4 sind die durchschnittlichen Leistungswerte der untersuchten Konfigurationen von CPLEX dargestellt. Die Gruppierung erfolgte nach der Suchstrategie. Dadurch konnte die **EMPOPT** als die schlechtere und **EMPINT** als die bessere identifiziert werden. Außerdem zeigen diese Experimente auch, dass das Zuschalten der Heuristik und der zusätzlichen Ungleichungen keine Verbesserung der Laufzeit bewirken. In der Grafik 4.4 ist die Konfiguration **HEU1_CUTS1_EMPBAL** als die Standardeinstellung von CPLEX schwarz markiert.

	HEU 0 CUTS 0	HEU 1 CUTS 0	HEU 0 CUTS 1	HEU 1 CUTS 1
EMP INT	**1.05**	1.33	1.52	2.01
EMP BAL	2.60	2.80	3.86	4.14
EMP OPT	3.67	3.59	5.37	5.45

HEU	Heuristik für zullässige Lösungen	0 = abgeschaltet	1 = automatische Frequenz	
CUTS	Cuts, alle Klassen	0 = abgeschaltet	1 = automatische Frequenz	
EMP	Ziel der MIP Optimierung	INT = integer feasibility	OPT = optimality	BAL = balanciert

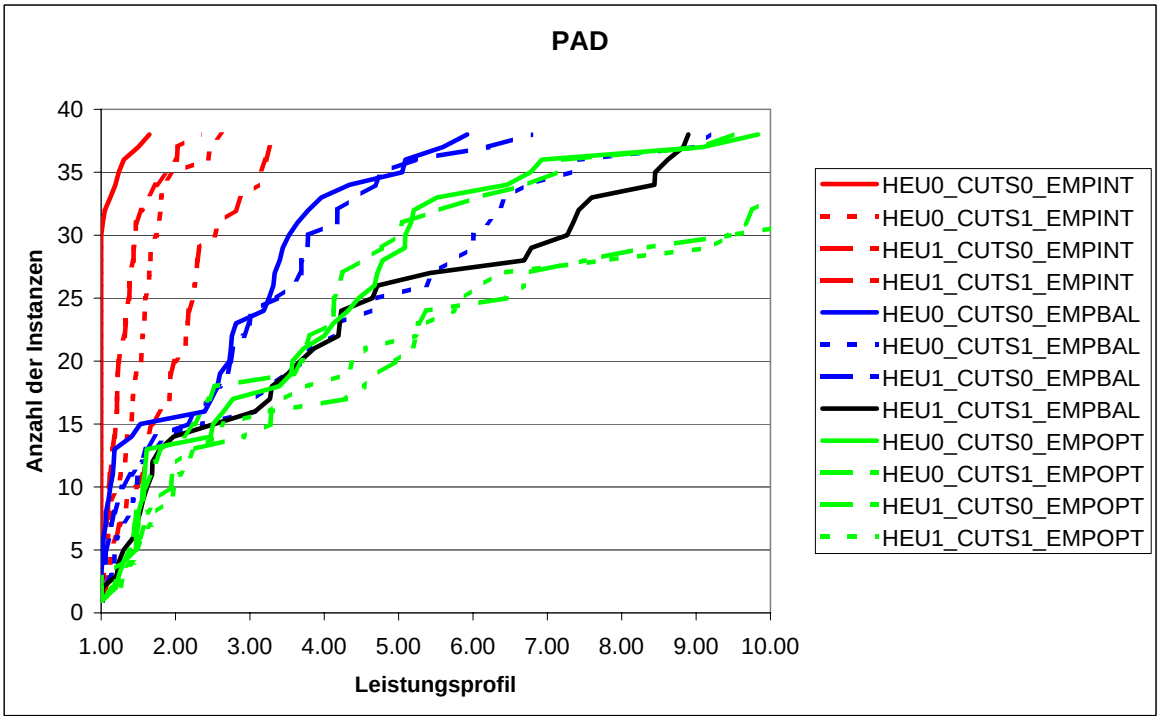


Abbildung 4.4: CPLEX Varianten auf der PAD Benchmark

4.1.2 Methodik der Auswertung

Wenn eine Netzwerkentwurf Probleminstanz von einem Solver gelöst wird, kann eine Vielzahl von Kenngrößen gesammelt und ausgewertet werden. Diese Kenngrößen können in mehreren Hierarchieebenen angeordnet werden. Auf der obersten Ebene interessieren wir uns für die Gesamtperformance eines Solvers:

- erzielter Wert der Zielfunktion,
- benötigte Laufzeit (Prozessorzeit),
- evtl. Speicherbedarf.

Anhand dieser Werte bestimmen wir den besten Solver für eine Klasse von Benchmarkdaten.

Formal erhalten wir für jeden Solver $s_i \in S$ eine Reihe von Tupeln (Zielfunktion, Laufzeit) $(o_{p,s}, t_{p,s})$ für ein $p \in P$ aus der Menge von Datensätzen.

Summe der Laufzeiten Wir betrachten zunächst den Fall wenn das CNDP exakt gelöst wird und die Laufzeiten $t_{p,s}$ verglichen werden sollen. Da bietet sich zunächst die Summe der Laufzeiten als Vergleichskriterium an: $t_s = \sum_p t_{p,s}$. Bei vergleichbaren Laufzeiten eines Solvers auf einer Benchmark ist dieser Wert geeignet, den besten Solver zu identifizieren. Bei stärker schwankenden Laufzeiten wird oft zusätzlich die mittlere Laufzeit und die Standardabweichung hinzugezogen.

Ranking Das Ranking der Verfahren eignet sich gut für einen groben Vergleich der Solver, wenn nur die Reihenfolge der Solver, nicht aber die quantitativen Unterschiede von Interesse sind. Das Ranking eines Solvers auf einer Instanz ist 1 falls er der schnellste auf dieser Probleminstanz war. Der Durchschnitt über die gesamte Benchmark lässt dann den schnellsten Solver ermitteln.

Leistungsprofile Wir definieren an dieser Stelle die Leistungswerte (*performance ratios*) sowie Leistungsprofile (*performance profiles*) (vgl. [Dolan and More, 2002]) und vergleichen diese Kriterien mit dem Durchschnitt bzw. der Summe der Laufzeiten.

Sei S die Menge der zu vergleichenden Verfahren und P die Menge der Testinstanzen und bezeichne

$t_{p,s} =$ die von dem Verfahren s für die Lösung des Problems p aufgewendete Zeit.

Anstelle der Zeit kann auch ein anderes Vergleichskriterium, z.B. die Anzahl der Knoten im B&B-Baum benutzt werden.

Definition 4.3 *Der* **Leistungswert** *(performance ratio) ist definiert als*

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s'} : s' \in S\}}$$

das Verhältnis des gegebenen Solvers s zum besten Solver auf dieser Probleminstanz. Zusätzlich wird ein Parameter $r_M \geq r_{p,s'} \forall p, s'$ definiert für den Fall wenn ein Solver eine Probleminstanz nicht lösen konnte: $r_{p,s} = r_M$.

Definition 4.4 *Das* **Leistungsprofil** *(*performance profile*) des Verfahrens $s \in S$ auf der Testmenge P ist die Funktion $Q_s(r)$, die für alle $r \in \mathbb{R}$ die Anzahl der Probleme in P anzeigt, für deren Lösung das Verfahren s einen Leistungswert von höchstens r aufweist:*

$$Q_s(r) = \left|\left\{p \in P : r_{p,s} \leq r\right\}\right|.$$

Die Funktion Q_s ist die kumulierte Wahrscheinlichkeitsverteilung für die Leistungswerte des Solvers s.

Motivation und Vergleich der Performance-Maße In unserem Szenario werden drei Solver (CPLEX 9.0, NDBB, NDBC) auf der PAD-Benchmark bzgl. ihrer Laufzeiten verglichen. Die kompletten Ergebnisse sind in der Tabelle 6.5 auf der Seite 162 zu finden.

	CPLEX 9.0	NDBB	NDBC
Summe	**3684.40**	**9329.80**	**1941.34**
Summe ohne J4,K1,L1	**1951.23**	**3611.77**	**877.32**
Summe mit Aussreisser bei L1	**9086.80**	**9329.80**	**1941.34**
Faktor in der Laufzeit	**1.90**	**4.81**	**1.00**
Faktor in der Laufzeit ohne J4,K1,L1	**2.22**	**4.12**	**1.00**
Faktor in der Laufzeit mit Aussreisser bei L1	**4.68**	**4.81**	**1.00**
Ranking	**2.17**	**2.59**	**1.24**
Ranking ohne J4,K1,L1	**2.16**	**2.58**	**1.26**
Ranking mit Aussreisser bei L1	**2.20**	**2.56**	**1.24**
Ø performance ratio	**2.72**	**4.85**	**1.83**
Ø performance ratio ohne J4,K1,L1	**2.77**	**4.89**	**1.90**
Ø performance ratio mit Aussreisser bei L1	**2.94**	**4.85**	**1.83**

Tabelle 4.4: Motivation zu Leistungsprofilen

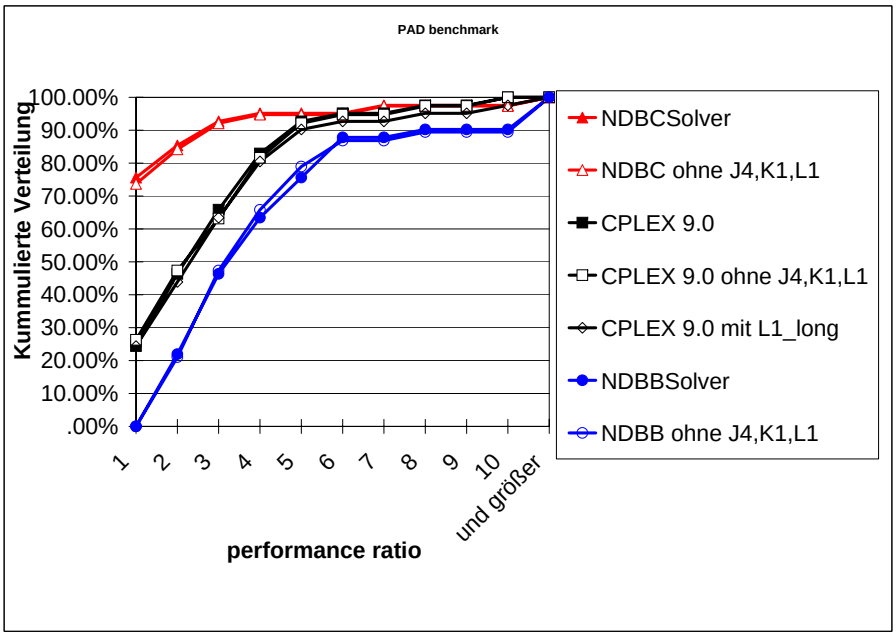


Abbildung 4.5: Diagramm zu performance profiles

In der ersten Zeile der Tabelle 4.4 ist die Summe der Laufzeiten auf der gesamten Benchmark angegeben. CPLEX 9.0 ist dabei um einen Faktor von 1.9 langsamer als NDBC, NDBB hat einen Faktor von 4,81 erzielt.

In der Grafik 4.5 sind die Leistungsprofile der drei Solver dargestellt. Im Bild können viele Informationen abgelesen werden, z.B.:

- NDBC ist auf 75% der Probleminstanzen der schnellste Solver gewesen.

- CPLEX 9.0 war auf den restlichen 25% der schnellste.
- NDBC konnte auf 85% der Probleminstanzen ein Leistungswert von ≤ 2 erreichen, CPLEX 9.0 nur auf 45%.
- Der Leistungswert der Solver auf 90% der Instanzen ist wie folgt: NDBC - 3, CPLEX 9.0 - 5, NDBB - 8.
- Zusätzlich berechnen wir noch die durchschnittlichen Leistungswerte: NDBC - 1.83, CPLEX 9.0 - 2.72, NDBB - 4.85.

Insgesamt stellen wir fest, dass der Solver NDBC der beste in diesem Experiment ist, gefolgt von CPLEX 9.0 und NDBB.

Stabilität der Performance-Maße Ein Performance-Maß, das zum Vergleich von unterschiedlichen Lösungsverfahren benutzt wird, sollte eine verlässliche Aussage aus einem Experiment liefern. Die Entscheidung zugunsten eines Verfahrens sollte sich nicht ändern, wenn sich die Ergebnisse geringfügig ändern. Verändert sich z.B. auf einer Probleminstanz die Laufzeit eines Verfahrens stark, sollte sich das Gesamtbild im Vergleich nicht wesentlich ändern. Eine geringe Schwankung der Laufzeiten auf der ganzen Benchmark sollte ebenfalls keine Änderung des Vergleichs bewirken.

Wir formulieren zwei Aussagen über die Leistungsprofile, die anschließend im experimentellen Vergleich bestätigt werden sollen.

Behauptung 4.1 *Seien zwei Verfahren s und s' auf Instanzen $p \in P \backslash q$ gleich schnell: $t_{p,s} = t_{p,s'}$. Daraus folgt, dass $\forall p \in P \backslash q$ gilt: $r_{p,s} = r_{p,s'}$. Da sich nur der Leistungswert $r_{q,s'}$ ändert, gilt für beliebiges $r \in \mathbb{R}$:*

$$|Q_s(r) - Q'_s(r)| \leq 1/n_p$$

wenn n_p die Anzahl der Probleminstanzen ist. Die Leistungswerte unterscheiden sich also nur gering voneinander.

Weiterhin kann folgende Aussage über die Leistungsprofile getroffen werden:

Behauptung 4.2 *Unterscheiden sich die Leistungswerte von zwei Verfahren bei jeder Probleminstanz nur leicht:*

$$|r_{p,s} - r^1_{p,s'}| \leq \varepsilon, \quad 1 \leq p \leq n_p$$

für ein $\varepsilon > 0$, so ist der Unterschied der Leistungsprofile ebenfalls klein:

$$\int_1^\infty |Q_s(t) - Q'_s(t)| dt \leq \varepsilon$$

Der Beweis dieser Behauptung kann in [Dolan and More, 2002] gefunden werden.

Wir verändern nun die Grunddaten in unserem Experiment auf folgende Weise (siehe Tabelle 4.4 und 6.5 auf Seite 162):

- Es wird eine Teilmenge der Datensätze betrachtet (ohne die Langläufer J4, K1, L1),
- Es wird ein künstlicher Ausreißer bei CPLEX 9.0 produziert (Laufzeit auf L1 um Faktor 10 erhöht).

Die Summe der Laufzeiten reagiert im ersten Fall mit einer relativ kleinen Veränderung mit Faktoren von 2,22 und 4,12. Der Ausreißer bei L1 verändert das Ergebnis des Vergleichs sehr stark: CPLEX 9.0 bekommt eine Gesamtlaufzeit von ca. 9000 sec und einen Faktor von 4,68 statt 1,90 ohne den Ausreißer.

Die durchschnittlichen Leistungswerte werden durch die Veränderungen viel weniger gestört: 2,72 auf 2,77 und dann auf 2,94 bei CPLEX 9.0. Die Veränderung der Leistungsprofile fällt ebenfalls sehr gering aus: Die Profile von CPLEX 9.0 liegen nach wie vor sehr eng beieinander (siehe Grafik 4.5).

Fazit zur Methodik der Auswertung Wir stellen fest: Auch nach einer Störung der Grunddaten ist eine genaue Analyse der Leistungsfähigkeit der Algorithmen möglich. Eine starke Änderung bei wenigen Datensätzen einer Benchmark oder eine geringe Veränderung bei einigen Datensätzen wird von Leistungsprofilen abgefangen und das Gesamtbild wird dadurch nicht zerstört.

Aufgrund dieser Stabilitätseigenschaft der Leistungswerte und Leistungsprofile verwenden wir diese Vergleichskriterien zur Leistungsbewertung von Algorithmen oder Varianten im weiteren Verlauf dieses Ergebnis-Kapitels.

4.1.3 Leistungsfähigkeit der Verfahren

In der Tabelle 6.1 sind die Experimente in diesem Abschnitt im Überblick aufgeführt.

Nr.	Benchmark	Verfahren im Vergleich	Art des Vergleichs
1.1	PAD	NDBB, NDBC, CPLEX 9.0	exakt
1.2	CANAD-R2	NDBC, CPLEX 9.0	exakt
1.3	CANAD-R2	NDBC, CPLEX 9.0, Kanadische Solver	heuristisch
1.4	CANAD-C	NDBC-α, NDBC, CPLEX 9.0, Kanadische Solver	heuristisch
1.5	CANAD-C	NDBB-α, NDBB-β, NDBC-α, NDBC-β	heuristisch
1.6	PAD-S	NDBC-α, NDBC	heuristisch

Tabelle 4.5: Überblick über die Experimente in diesem Abschnitt

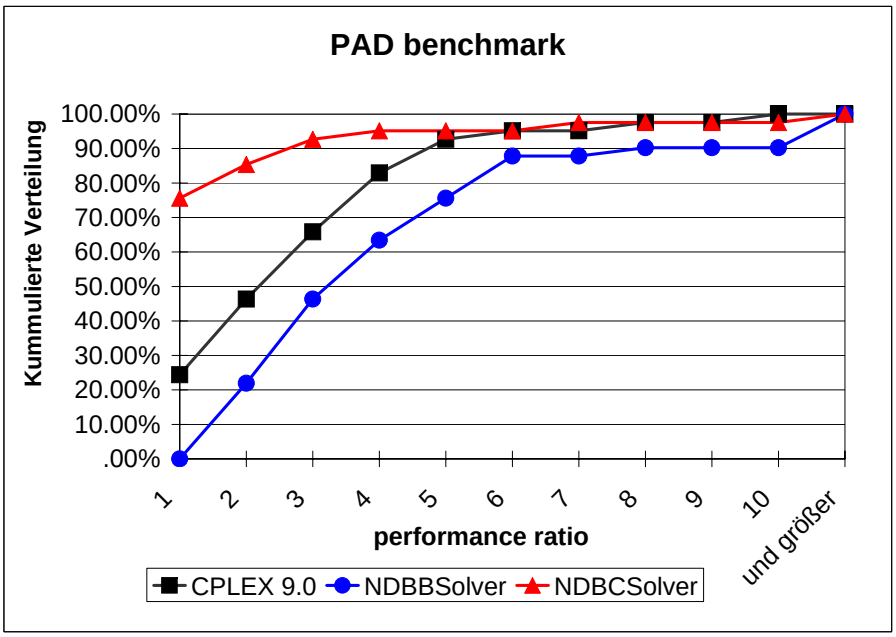
Wir vergleichen die in dieser Arbeit entwickelten Solver NDBB und NDBC mit anderen Solvern für das Problem des Netzwerkentwurfs. Zum einen lösen wir die Instanzen mit Hilfe des Branch-and-cut Algorithmus von ILOG CPLEX 9.0 ([ILOG, 2003]). Weiterhin konnten wir auf Ergebnisse der kanadischen Wissenschaftler zurückgreifen (Bernard Gendron).

In den Arbeiten von Ghamlouche und anderen [Ghamlouche et al., 2003], [Ghamlouche et al., 2004] werden drei heuristische Verfahren für das Capacitated Network Design Problem (CNDP) vorgestellt. Der TABU-PATH Solver aus [Ghamlouche et al., 2003] implementiert einen heuristischen Tabu Search Algorithmus, der auf einer aufwendigen Nachbarschaft für das CNDP

basiert. Eine Erweiterung dieses Ansatzes um Path Relinking ist im Solver PATH-RELINKING [Ghamlouche et al., 2004] implementiert. Eine relativ einfache, kantenbasierte Nachbarschaft wird innerhalb eines Tabu Search Ansatzes (TABU-ARC) benutzt.

In [Crainic et al., 2004] wird der Slope Scaling Ansatz SS/PL/ID für das CNDP vorgestellt. Es wird eine Folge von Multicommodity Flussproblemen gelöst, die gegen die optimale Lösung des CNDP konvergiert.

Experiment 1.1 In diesem Experiment werden die Solver NDBB, NDBC und CPLEX 9.0 auf der Benchmark PAD verglichen.



	CPLEX 9.0	NDBB	NDBC
Summe in sec	**3684.40**	**9329.80**	**1941.34**
Faktor in der Laufzeit	**1.90**	**4.81**	**1.00**
Ø performance ratio	**2.72**	**4.85**	**1.83**

Abbildung 4.6: Experiment 1.1: exaktes Lösen der PAD Benchmark (siehe auch 6.5, Seite 162).

Experimentaufbau. Bei diesem Experiment wird jeder Solver mit einem Zeitlimit von 4 Stunden gestartet. Gemessen wird die Zeit zum Finden der optimalen Lösung und Beweis der Optimalität dieser. Wir berechnen anschließend die Kenngrößen, die in der Grafik und Tabelle 4.6 dargestellt sind.

Ergebnis. Der NDBC Solver löst alle Probleminstanzen in 1941 Sekunden, deutlich schneller als CPLEX (3684 Sek.) und der NDBB Solver (9329 Sek.). Der durchschnittliche Leistungswert liegt beim NDBC Solver bei 1.83, beim CPLEX bei 2.72 und beim NDBB bei 4.85. Auf 75% der Probleminstanzen ist NDBC der schnellste Solver, CPLEX auf 25%. Das Leistungsprofil von NDBC liegt deutlich höher als die der beiden anderen Solver.

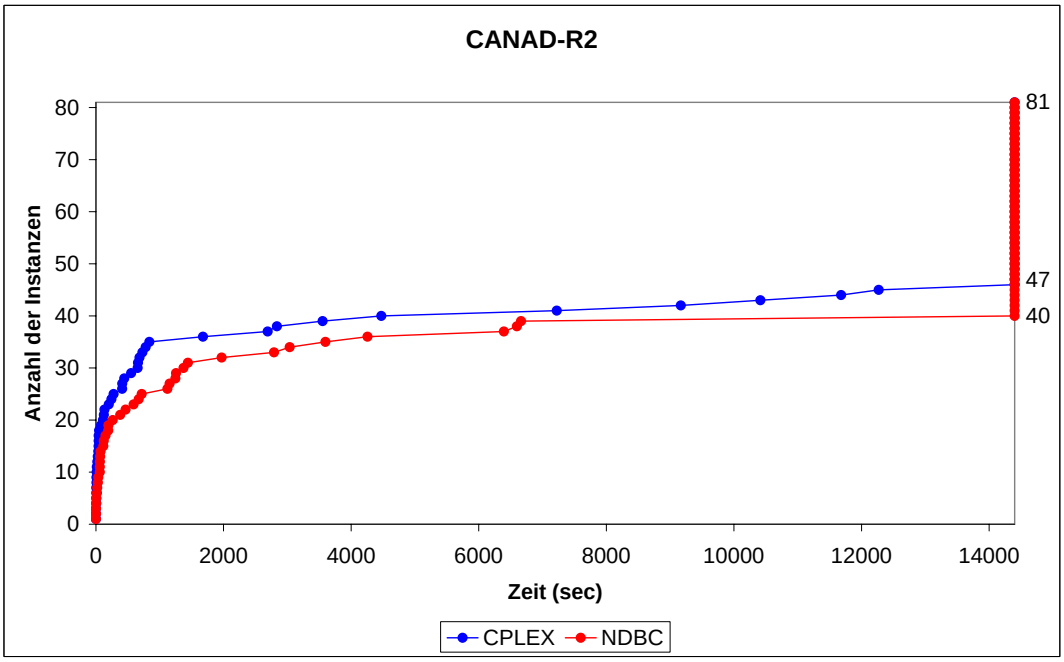
Auswertung. NDBC kann als besserer Solver für das exakte Lösen der Benchmark PAD identifiziert werden. Alle drei Vergleichsgrößen (Summe der Laufzeiten, durchschnittlicher Leistungswert und das Leistungsprofil) sind besser als bei den beiden anderen Solvern NDBB und CPLEX. Damit bewährt sich die Konfiguration unseres Systems, die auf der Bundle-Methode und dem Relax-and-cut-Algorithmus basiert.

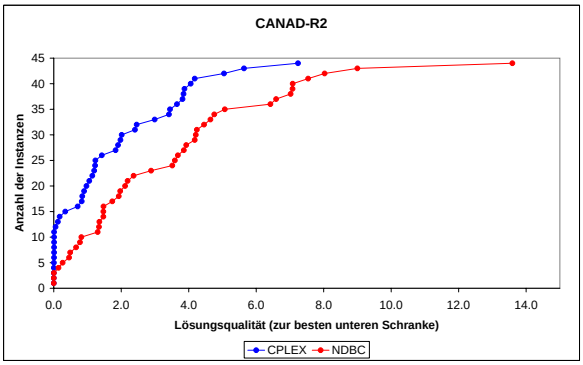
Experiment 1.2 In diesem Experiment werden die Solver NDBC und CPLEX 9.0 auf der Benchmark CANAD-R2 verglichen. Diese Benchmark zeichnet sich dadurch aus, dass nicht jede Probleminstanz in 4 Stunden Zeitlimit optimal gelöst werden kann. Deswegen unterscheiden wir bei der Auswertung nach Instanzen, die optimal gelöst werden konnten und anderen, bei denen wir die Qualität der gefundenen Lösungen vergleichen.

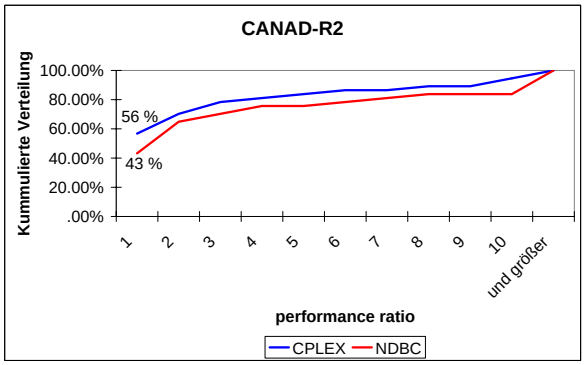
Experimentaufbau. Jeder Solver bekommt für jede der 81 Instanzen 4 Stunden (14400 Sek.) Rechenzeit. Protokolliert werden der Wert der Zielfunktion und die Laufzeit (max. 14400 Sek.). Die Lösungsqualität wird zur besten bekannten unteren Schranke berechnet.

Ergebnis. In den Grafiken 4.7(1)-(3) und Tabelle 4.7 sind folgende Informationen enthalten. Grafik (1): Die kumulierte Verteilung der sortierten Lösungszeit. NDBC konnte für 40 Instanzen die optimale Lösung finden und die Optimalität beweisen. CPLEX fand 47 optimale Lösungen. Grafik (2): Für 44 Instanzen, die von einem der beiden Solvern nicht optimal gelöst wurden, ist die Lösungsqualität im Leistungsprofil angegeben. Grafik (3) ist eine andere Darstellung der Grafik (1): Es wurden die Leistungswerte berechnet und als Leistungsprofil dargestellt. CPLEX zeigt dabei das bessere Verhalten. In der Tabelle zu 4.7 ist die Gesamtzeit zum Lösen der 37 Instanzen, die von beiden Solvern optimal gelöst wurden, angegeben. NDBC benötigte dafür 2% mehr Zeit. Der durchschnittliche Leistungswert spricht an dieser Stelle für NDBC: 2.72 gegen 3.00 bei CPLEX. Die durchschnittliche Lösungsqualität bei nicht optimal gelösten Instanzen ist bei CPLEX deutlich besser: 1.73 gegen 3.40 bei NDBC.

Auswertung. CPLEX zeigt eine bessere Performance beim exakten Lösen der Benchmark CANAD-R2 mit einem Zeitlimit von 4 Stunden. CPLEX konnte mehr Instanzen optimal lösen, war auf diesen Instanzen geringfügig schneller als NDBC und lieferte bei nicht optimal gelösten Instanzen eine bessere Lösungsqualität.







	CPLEX 9.0	NDBC
Optimal gelöste Instanzen		
Summe in sec	37715.19	38571.72
Faktor in der Laufzeit	1.00	1.02
Ø performance ratio	3.00	2.72
Nach 4 Std. nicht gelöste Instanzen		
Ø Lösungsqualität	1.73	3.40

Abbildung 4.7: Experiment 1.2. 14400 sec Zeitlimit (siehe auch 6.6, Seite 163).

Experiment 1.3 In der Grafik 4.8 ist ein Vergleich auf der Benchmark **Canad-R2** dargestellt. Folgende Verfahren aus den Publikationen zum Thema Netzwerkentwurf werden mit unseren heuristischen Verfahren verglichen:

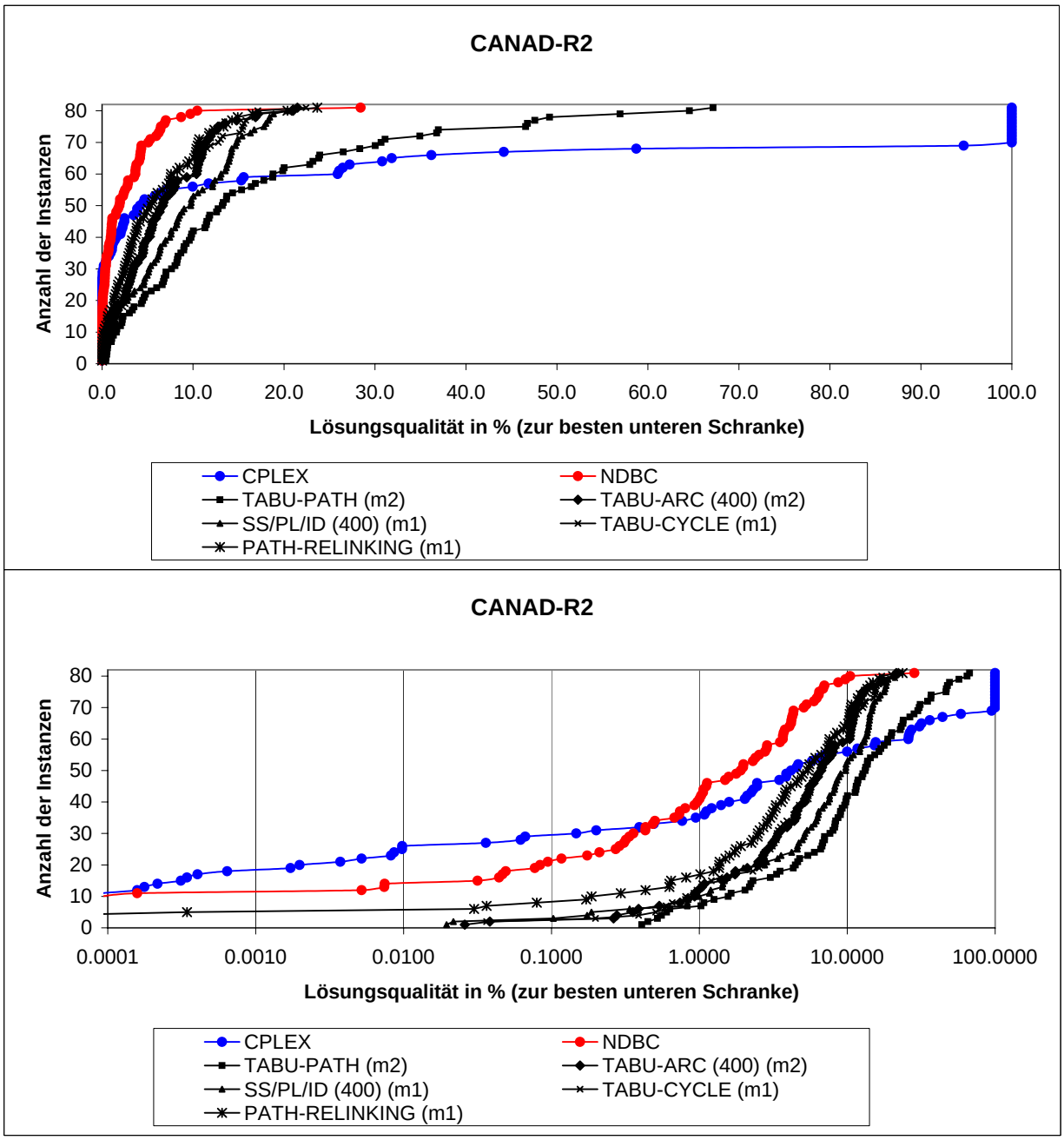
- TABU-CYCLE und TABU-ARC [Ghamlouche et al., 2003],
- TABU-PATH [Crainic et al., 2000b],
- PATH-RELINKING [Ghamlouche et al., 2004],
- SS/PL/ID [Crainic et al., 2004].

Die Ergebnisse dieser Verfahren wurden uns von den Autoren der entsprechenden Publikationen zur Verfügung gestellt. Eine Tabelle mit genauen Daten ist im Anhang (Tabelle 6.7) zu finden. Die Zeiten wurden bei diesem Vergleich nicht berücksichtigt, weil die Verfahren auf unterschiedlichen Rechenplattformen gestartet wurden.

Experimentaufbau. Sämtliche 81 Instanzen der Benchmark CANAD-R2 werden hier untersucht. Der Vergleich basiert auf der erreichten Lösungsqualität bezüglich der besten bekannten unteren Schranke. Die Alpha-Fixierung des NDBC Solvers wird nach 300 Sekunden gestoppt. Der B&C-Algorithmus von CPLEX wird in der heuristischen Variante gestartet und die Lösungsqualität nach 300 Sekunden protokolliert. Die Einstellungen wurden so vorgenommen, dass möglichst schnell eine zulässige Lösung gefunden werden kann, ohne den Optimalitätsbeweis zu führen.

Ergebnis. Das Ergebnis des Vergleichs ist in der Grafik und Tabelle 4.8 dargestellt. Grafik (1) zeigt die kumulierte Verteilung der Lösungsqualität. Dabei liefert NDBC das beste Ergebnis im Vergleich zu allen anderen Verfahren. CPLEX konnte für 11 Instanzen keine zulässige Lösung innerhalb von 300 Sekunden liefern, bei 10 Instanzen lag die Qualität zwischen 25% und 95%. Bei 23 Instanzen liefert CPLEX die beste Qualität, bei 49 gewinnt NDBC, bei 9 Instanzen sind die Ergebnisse gleich. Die kanadischen Solver liefern eine Lösungsqualität von durchschnittlich 5-15% und können für alle Instanzen eine zulässige Lösung finden. In der Grafik (2) sind die Daten in der logarithmischen Darstellung zu sehen: CPLEX kann in 300 Sek. 17 Instanzen optimal lösen und liefert bei diesen Instanzen eine sehr gute untere Schranke.

Auswertung. Die heuristische Konfiguration des NDBC Solvers konnte bessere Ergebnisse auf der Benchmark CANAD-R2 liefern als alle anderen Solver. Obwohl die benötigte Zeit bei diesem Experiment nicht verglichen wurde, sollte man anmerken, dass die heuristischen kanadischen Solver wesentlich längere Laufzeiten aufweisen und trotzdem schlechtere Lösungsqualitäten liefern. Die heuristische Konfiguration eignet sich durch Stabilität und gute Lösungsqualität besonders gut für Probleminstanzen dieser Größe. Ein ähnliches Ergebnis lässt sich bei der größeren Benchmark CANAD-C feststellen (siehe das nächste Experiment 1.4).



	Ø solution quality
TABU-PATH (m2)	15.20
TABU-ARC (400) (m2)	6.25
SS/PL/ID (400) (m1)	8.18
TABU-CYCLE (m1)	6.40
PATH-RELINKING (m1)	5.32
NDBC-Alpha 300 sec	2.38
CPLEX 9.0 300 sec	21.26

Abbildung 4.8: Experiment 1.3. 300 sec Zeitlimit (siehe auch 6.7, Seite 164).

Experiment 1.4 In der Grafik 4.9 ist ein Vergleich auf der Benchmark **Canad-C** dargestellt. Folgende Verfahren aus den Publikationen zum Thema Netzwerkentwurf werden mit unseren heuristischen Verfahren verglichen:

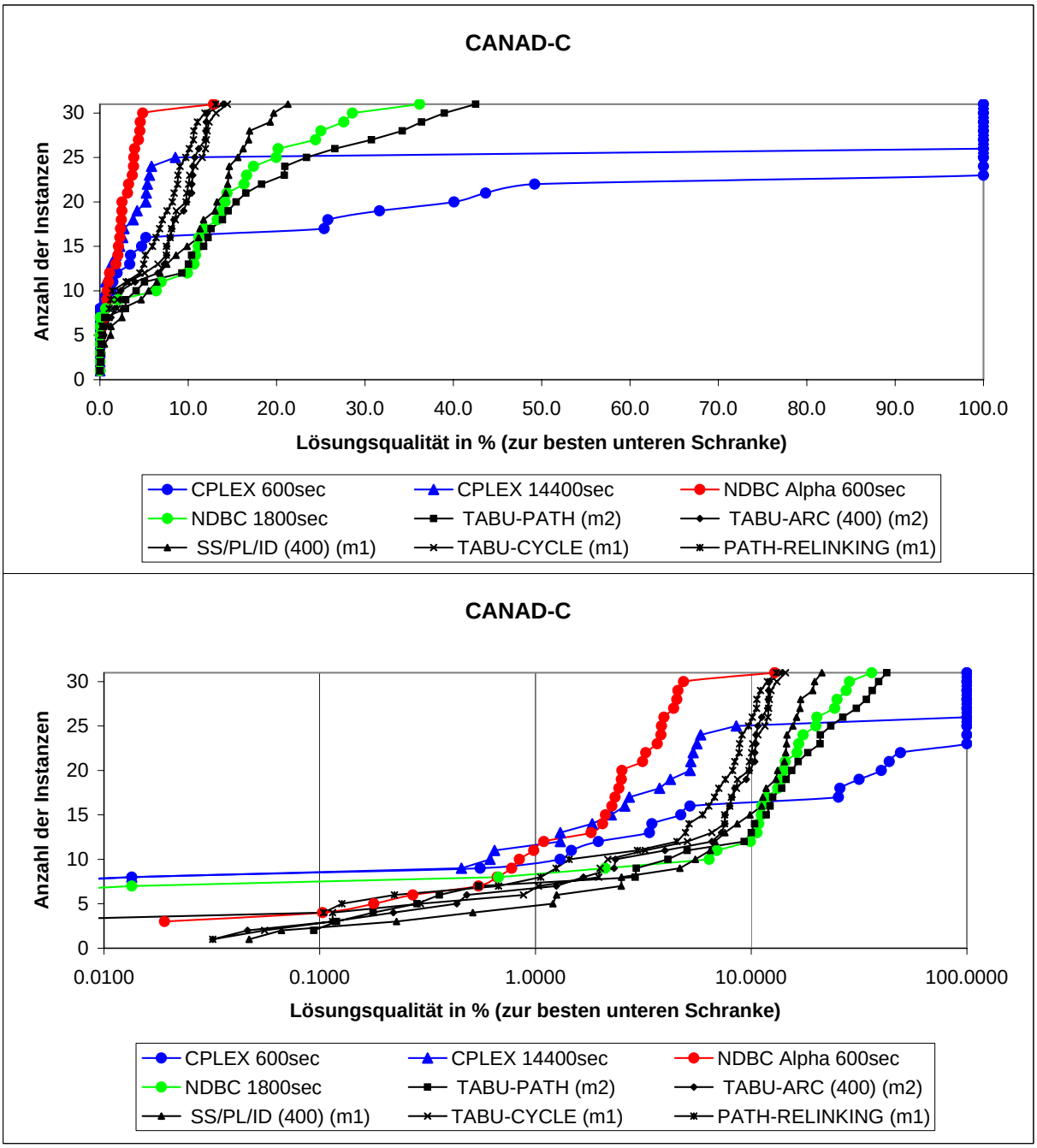
- TABU-CYCLE und TABU-ARC [Ghamlouche et al., 2003],
- TABU-PATH [Crainic et al., 2000b],
- PATH-RELINKING [Ghamlouche et al., 2004],
- SS/PL/ID [Crainic et al., 2004].

Wie auch im Experiment 1.3 können wir auf Ergebnisse der kanadischen Autoren zurückgreifen. Vollständige Daten finden sich im Anhang (Tabelle 6.8).

Experimentaufbau. Die Alpha-Fixierung des NDBC Solvers wird nach 600 Sekunden gestoppt. Die exakte Variante des NDBC Solvers wird für 1800 Sekunden gestartet. Zum Vergleich sind noch zwei Varianten des Branch-and-cut Algorithmus von CPLEX 9.0 angegeben. Die erste, heuristische Variante wurde ebenfalls für 600 Sekunden gestartet. Die Einstellungen wurden so vorgenommen, dass möglichst schnell eine zulässige Lösung gefunden werden kann, ohne den Optimalitätsbeweis zu führen. Die andere Variante sollte innerhalb von 4 Stunden eine möglichst gute Lösung finden.

Ergebnis. In der Grafik (1) vergleichen wir die durchschnittlichen Lösungsqualitäten der Verfahren. Das Leistungsprofil des NDBC Solvers liegt deutlich oberhalb aller anderen Verfahren. CPLEX liefert ebenfalls sehr gute Lösungen, ist aber nicht stabil. Die heuristische Variante konnte für 10 Instanzen keine zulässige Lösung finden, die exakte für 5 Instanzen (entspricht einer Lösungsqualität von 100% zur besten unteren Schranke). Die durchschnittliche Lösungsqualität von CPLEX leider darunter. Die durchschnittliche Lösungsqualität von Alpha-NDBC ist die beste in diesem Vergleich: 2.46%. Die Qualität der exakten Konfiguration von NDBC ist mit 11.92% unterdurchschnittlich. Die kanadischen Solver liefern auch auf dieser Benchmark eine Lösungsqualität von durchschnittlich 5-15% und können für alle Instanzen eine zulässige Lösung finden.

Auswertung. Als Erstes bewerten wir den Unterschied zwischen der exakten und der heuristischen Version des NDBC Solvers. Ohne die heuristische Variablenfixierung (Alpha Strategie) ist die gelieferte Lösungsqualität schlecht im Vergleich zu allen anderen Solvern. Dieses Ergebnis betont die Wichtigkeit und die Leistungsfähigkeit der Alpha-Fixierung. Aus der Grafik wird deutlich, dass die heuristische Konfiguration Alpha-NDBC klar die anderen Systeme in puncto Lösungsqualität dominiert. Sowohl die durchschnittliche Lösungsqualität als auch die kumulierte Verteilungsfunktion sind besser. Somit können auch auf dieser größeren Benchmark in kurzer Zeit sehr gute Lösungen berechnet werden.



	Ø solution quality
TABU-PATH (m2)	14.07
TABU-ARC (400) (m2)	6.82
SS/PL/ID (400) (m1)	9.75
TABU-CYCLE (m1)	6.77
PATH-RELINKING (m1)	5.70
NDBC-Alpha 600 sec	2.46
NDBC 1800 sec	11.92
CPLEX 9.0 600 sec	36.71
CPLEX 9.0 14400 sec	21.21

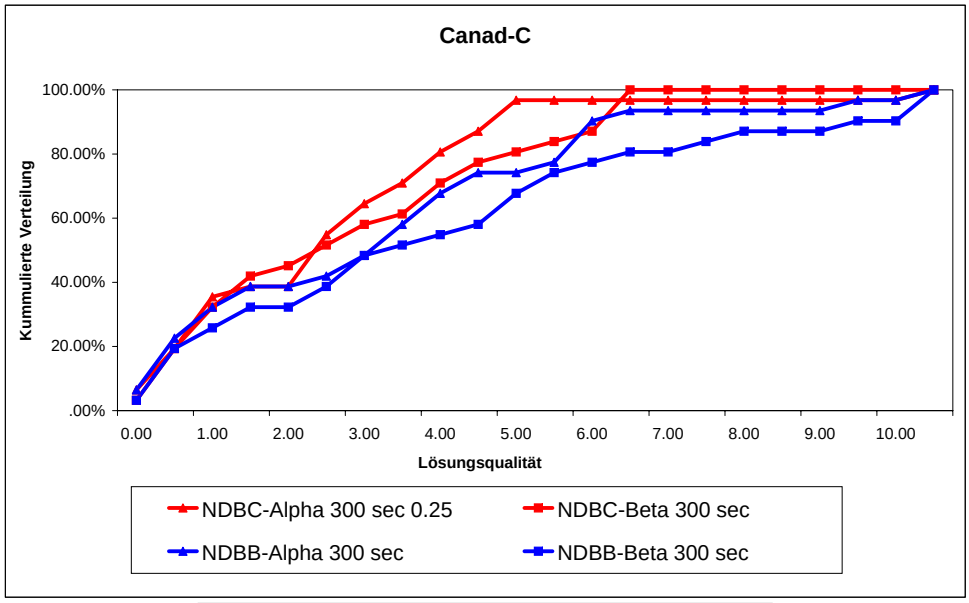
Abbildung 4.9: Experiment 1.4. (siehe auch 6.8, Seite 165).

Experiment 1.5 In diesem Experiment vergleichen wir unterschiedliche Konfigurationen unserer Solver NDBC und NDBB auf der Benchmark **Canad-C**.

Experimentaufbau. Jede Konfiguration wurde für 300 Sekunden gestartet. Wir vergleichen die Lösungsqualität zur besten bekannten unteren Schranke.

Ergebnis. In der Grafik und Tabelle 4.10 zeigt sich, dass der NDBC generell eine bessere Lösungsqualität liefert als der NDBB Solver. Auch lässt sich feststellen, dass die Alpha-Fixierung besser als die Beta-Fixierung ist.

Auswertung. Die heuristische Variante Alpha-NDBC wurde unter anderem als Ergebnis dieses Experiments als die Standard-Konfiguration unseres Systems ausgewählt.



	Ø solution quality
NDBC-Alpha 300 sec	2.51
NDBC-Beta 300 sec	2.70
NDBB-Alpha 300 sec	3.19
NDBB-Beta 300 sec	4.13

Abbildung 4.10: Experiment 1.5 (siehe auch 6.9, Seite 166).

Experiment 1.6 In diesem Experiment vergleichen wir unterschiedliche exakte und heuristische Konfigurationen unserer Solver NDBC und NDBB auf der größten Benchmark **PAD-S**. CPLEX konnte auf dieser Benchmark auch nach einem Zeitlimit von 4 Stunden keine zulässigen Lösungen finden. Der Grund dafür ist die Größe und Komplexität der LP-Relaxation der vorliegenden Probleminstanzen. Kanadische Solver konnten wir nicht auf dieser Benchmark testen.

Experimentaufbau. Folgende Varianten wurden hier verglichen: Alpha-NDBC Solver mit unterschiedlichen Zeitlimits (300, 1800, 7200 Sekunden pro Probleminstanz), sowie zwei exakte NDBC Läufe: mit 24 und 70 Stunden Zeitlimit pro Instanz.

Ergebnis. In der Grafik 4.11(1) sind die Lösungsqualitäten zur besten bekannten zulässigen Lösung dargestellt. Die Grafik (2) zeigt die Ergebnisse zur besten bekannten unteren Schranke. Wir erinnern uns an die durchschnittliche Lücke von 16% zwischen diesen beiden Werten (siehe Tabelle 4.3). In beiden Grafiken lässt sich sehr gut die zeitliche Konvergenz der Verfahren beobachten. Die exakten Verfahren berechneten fast immer die besten bekannten zulässigen Lösungen. Die Alpha-Verfahren konvergieren bereits nach 1800 Sekunden fast gegen diese Lösungen. Nach 300 Sekunden sind die Lösungen noch deutlich schlechter als nach 1800 Sekunden.

Auswertung. Als Ergebnis stellen wir fest, dass für sämtliche Probleminstanzen dieser größten Benchmark zulässige Lösungen berechnet werden konnten. Die Konvergenz des heuristischen Alpha-NDBC Solvers tritt bereits nach 1800 Sekunden ein, eine längere Laufzeit von 7200 Sekunden liefert nur eine geringfügig bessere Lösungsqualität. Negativ zu erwähnen ist die große Lücke zu den unteren Schranken, die durchschnittlich bei 16% liegt. An dieser Stelle kann durch schärfere untere Schranken eine Verbesserung erzielt werden.

Zusammenfassung: Leistungsfähigkeit der Verfahren auf unterschiedlichen Benchmarks

1. Der NDBC Solver ist der beste exakte Solver auf der Benchmark PAD.
2. CPLEX kann die Benchmark CANAD-R2 besser exakt lösen als der NDBC Solver.
3. NDBC ist der beste heuristische Solver auf der Benchmark CANAD-R2.
4. NDBC ist der beste heuristische Solver auf der Benchmark CANAD-C.
5. Die heuristische Konfiguration Alpha-NDBC dominiert andere heuristische Konfigurationen des NDBC Solvers.
6. Die größten Probleminstanzen (PAD-S) konnten von unserem Solver Alpha-NDBC im Gegensatz zu CPLEX gelöst werden.

Als Ergebnis können wir ein sehr leistungsfähiges System für den Netzwerkentwurf vorweisen. Bei der exakten Lösung der Netzwerkentwurfprobleme zeigen unsere Experimente bessere Performance als das auf der Subgradient-Suche basierte System und auch als der Branch-and-Cut Algorithmus von CPLEX 9.0. Das System konnte auch andere heuristische Lösungsverfahren für das Problem des Netzwerkentwurfs in der Qualität der gelieferten Lösungen und der Laufzeit übertreffen.

Nach der Auswertung der Leistungsfähigkeit des Gesamtsystems gehen wir im nächsten Abschnitt auf die Leistungsfähigkeit der einzelnen Komponenten ein.

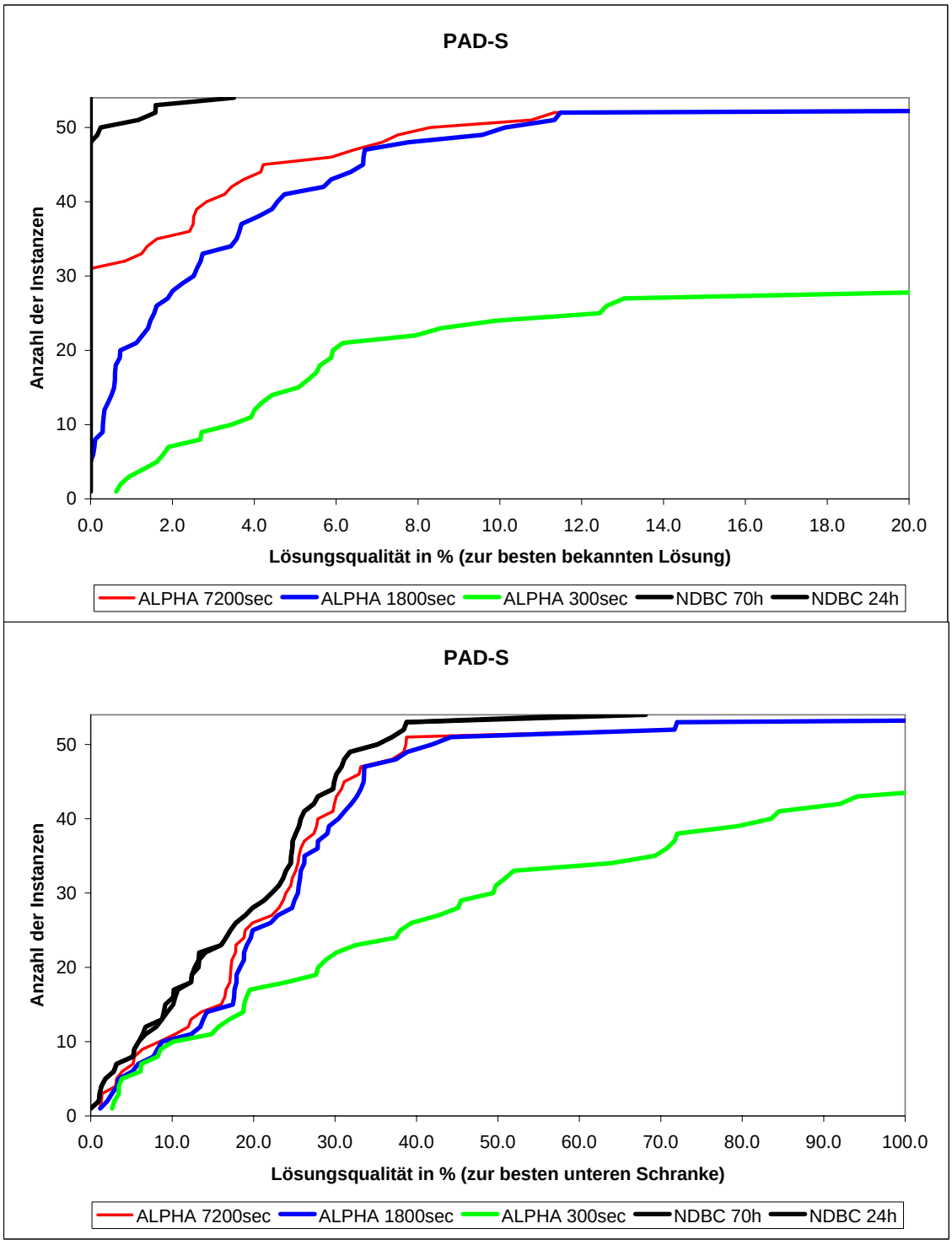


Abbildung 4.11: Experiment 1.6. (siehe auch 6.10, Seite 167).

4.1.4 Wesentliche Systemkomponenten

In Tabelle 6.2 sind die Experimente in diesem Abschnitt im Überblick aufgeführt.

Nr.	Benchmark	Komponente	Art des Experiments
2.1	CANAD-C	Erste Phase	heuristisch
2.2	PAD	Primale Heuristik	exakt
2.3	PAD	Variablenfixierung	exakt
2.4	CANAD-C	Variablenfixierung	heuristisch
2.5	PAD	Zusätzliche Ungleichungen (Cuts)	exakt
2.6	CANAD-R2	Zusätzliche Ungleichungen (Cuts)	exakt
2.7	CANAD-C	Zusätzliche Ungleichungen (Cuts)	heuristisch
2.8	CANAD-C	α und β Werte	heuristisch

Tabelle 4.6: Überblick über die Experimente in diesem Abschnitt

Experiment 2.1: Auswirkungen der ersten Phase

In der ersten Phase des B&B- oder B&C-Algorithmus verwenden wir die heuristische β-Variablenfixierung. Das Ziel dieser kurzen ersten Phase ist das schnelle Finden einer möglichst guten zulässigen Lösung. Der Wert der Zielfunktion dieser Lösung wird in der zweiten Phase helfen, den Suchbaum klein zu halten und die Variablenfixierung möglichst effektiv zu betreiben. Die β-Heuristik eignet sich gut für diesen Zweck, weil sie in jeder Ebene des Suchbaums eine relativ große Zahl der Variablen fixiert und eine kurze Laufzeit garantiert. Die erste Phase wird für einige Knoten durchgeführt, deren Anzahl von der Größe der Probleminstanz abhängt. Für große Probleminstanzen liegt diese Anzahl bei ca. 300 Suchknoten.

Experimentaufbau. Der Vergleich wird auf der Benchmark CANAD-C mit dem heuristischen Alpha-NDBC Solver durchgeführt. Die erste Variante benutzt die 1. Phase, die zweite nicht. Zwei Varianten bekommen jeweils 1800 Sekunden pro Probleminstanz. Die Lösungsqualität zur besten bekannten unteren Schranke ist das Vergleichskriterium.

Ergebnis. Das Leistungsprofil liegt für die Variante 1.+2. deutlich oberhalb der Variante 2. Phase (siehe Grafik 4.12). Die Variante 2. Phase liefert für 20 von 31 Instanzen eine Lösungsqualität von über 10%. Die schlechteste Lösungsqualität der Variante 1.+2. liegt bei 7.54%.

Auswertung. Die 1. Phase verbessert die Qualität der Lösungen ganz entscheidend. Ohne diese heuristische Phase liefert die zweite, exakte Phase keine brauchbaren Resultate. Der B&B-Algorithmus kann ohne gute obere Schranken nicht effektiv arbeiten und die Variablenfixierung ist ebenfalls weniger effizient.

Experiment 2.2: Primale Heuristik

Die primale Heuristik wurde im Abschnitt 2.7 auf Seite 40 beschrieben. Die Heuristik hat eine große Bedeutung für die B&B-Suche, da mit neuen guten Lösungen Suchknoten abgeschnitten werden können und neue Variablen fixiert werden können.

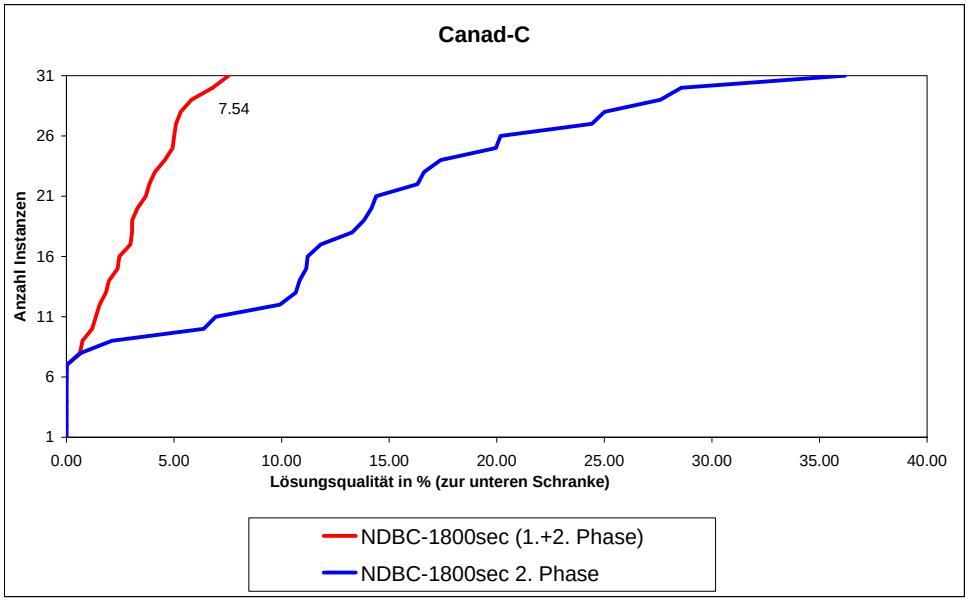


Abbildung 4.12: Experiment 2.1 (siehe auch 6.11, Seite 168).

Experimentaufbau. In diesem Experiment soll untersucht werden, wie stark die Leistungsfähigkeit des Systems von der Häufigkeit abhängt, mit der diese Heuristik aufgerufen wird. Die Häufigkeit wird als die Anzahl der Iterationen der Subgradient-Suche oder des Bundle-Algorithmus definiert, nach der die Heuristik gestartet wird. Wir variieren diesen Wert zwischen 5 und 50 Iterationen, wobei der Standardwert in unserem System bei 20 Iterationen liegt. Wir verwenden für diesen Vergleich die PAD Benchmark.

Ergebnis. In der Grafik 4.13(1) ist auf der x-Achse die Häufigkeit der primalen Heuristik angegeben (5–50). Auf der linken (primären) y-Achse ist der Leistungswert einer Konfiguration angegeben. Auf der rechten (sekundären) y-Achse ist die für die gesamte Benchmark Laufzeit angegeben. In der Grafik sind drei Linien dargestellt: Die Erste (in Rot) ist der durchschnittliche Leistungswert einer Konfiguration, die Zweite (in Blau) der maximale (schlechteste) Leistungswert und die Dritte (in Schwarz), die sich auf die sekundäre y-Achse bezieht und die Laufzeit angibt. In der Grafik 4.13(2) sind die Leistungsprofile der verglichenen Konfigurationen dargestellt. Die ersten drei Linien (in Blau) entsprechen den Werten 5,10,15. Die vierte (rote) Linie entspricht der Standard-Einstellung von 20 Iterationen zwischen zwei aufeinander folgenden Heuristik-Aufrufen. Die anderen Werte liegen sehr eng beieinander und sind alle in Schwarz dargestellt.

Auswertung. Die Grafik 4.13(1) zeigt, dass der Wert 20 einen guten Kompromiss zwischen Leistungswert und Laufzeit darstellt. Diese Einstellung liefert einen guten durchschnittlichen Wert von 1.43 und keine Ausreißer in der Laufzeit (schlechtester Wert: 2.30). Die Laufzeit liegt dabei mit 7220 Sekunden im Mittelfeld. Beim Vergleich der Leistungsprofile in der Grafik 4.13(2) zeigt sich, dass die Werte 5,10 und 15 eine schlechte Performance liefern, die deutlich unter der aller anderen Einstellungen liegt. Das Leistungsprofil der Einstellung 20 liegt deutlich oberhalb dieser Werte und ist vergleichbar mit den anderen sehr guten Einstellungen. Die Wahl des Parameters 20 bei der

Standard-Konfiguration wurde auch aufgrund dieses Experiments vorgenommen.

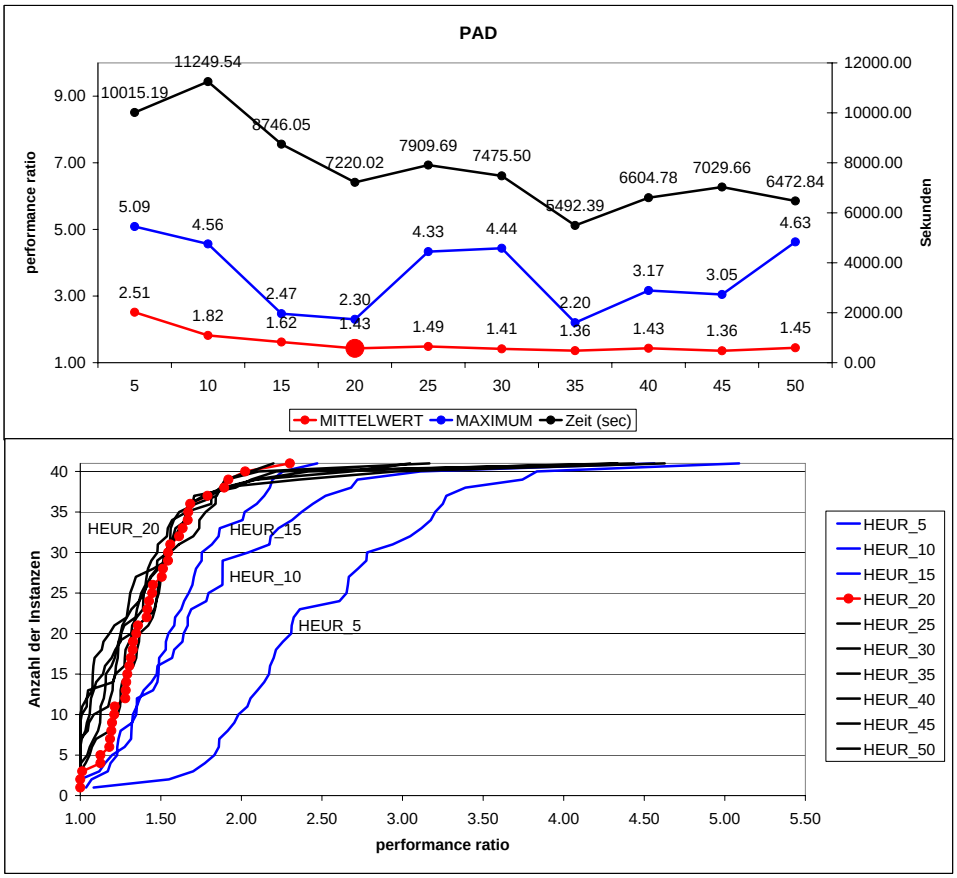


Abbildung 4.13: Experiment 2.2. Primale Heuristik

Auswirkungen der Variablenfixierung

Das Konzept und verschiedene Strategien der Variablenfixierung wurden im Abschnitt 2.9 auf der Seite 50 vorgestellt. In diesem Abschnitt untersuchen wir die Leistungsfähigkeit folgender Strategien der Variablenfixierung:

- Knapsack-Fixierung (siehe Abschnitt 2.9.1, Seite 51)
- Einschränkung der Kardinalitätsintervalle (Cardinality cuts) (siehe Abschnitt 2.9.4, Seite 54)
- Die zusätzlichen Ungleichungen (Cover cuts und local cuts) (siehe Abschnitt 2.11, Seite 58) bleiben dabei immer eingeschaltet. Diese werden in einem der nächsten Abschnitte genauer untersucht.

Experiment 2.3: Variablenfixierung, exakter Vergleich In diesem Experiment wird die Benchmark PAD exakt mit unterschiedlichen Konfigurationen des NDBC Solvers gelöst.

Experimentaufbau. Die untersuchten acht Varianten sind in der Tabelle 4.14 aufgeführt. Das Experiment basiert auf dem Vergleich der Leistungsprofile der Varianten.

Ergebnis. In der Grafik 4.14 sind drei Varianten grafisch dargestellt. Bei der Standard-Variante (**311**) werden die beiden untersuchten Komponenten nacheinander abgeschaltet. Dabei entstehen die Varianten ohne Cardinality cuts (**301**) und ohne Knapsack-Fixierung (**011**). In der Tabelle 4.14 sind durchschnittliche Leistungswerte, Laufzeit und die Anzahl der B&B-Suchknoten dargestellt.

Bei den Leistungsprofilen kann man feststellen, dass die Variante ohne die Cardinality cuts (**301**) zu einem schlechteren Ergebnis führt. Noch schlechter wird das Leistungsprofil bei der Variante ohne Knapsack-Fixierung (**011**). Dies ist gleichzeitig die langsamste Variante von allen - mit 3122 Sekunden. Die Standard-Variante (**311**) ist sowohl die schnellste mit 2148 Sekunden als auch die mit den wenigsten B&B-Suchknoten: 16383. In Prozent ausgedrückt, kostet das Abschalten der Cardinality cuts (**301**) 7% mehr Laufzeit und 8% mehr Knoten. Durch das Abschalten der Knapsack-Fixierung (**011**) vergrößert sich die Laufzeit um 45% und die Anzahl der Knoten um 46%.

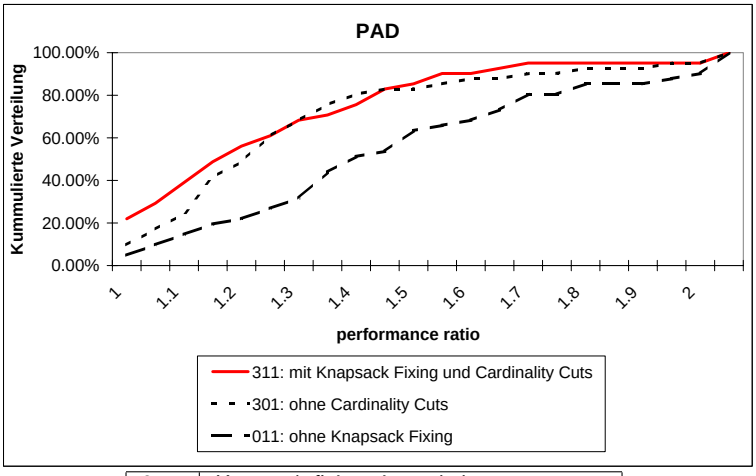
Auswertung. Die Knapsack-Fixierung kann als eine sehr wichtige Systemkomponente identifiziert werden, die zu starken Laufzeitverkürzungen beim exakten Lösen des Netzwerkentwurf Problems führt. Das Konzept der Cardinality cuts leistet ebenfalls einen Beitrag zur Verbesserung der Leistungsfähigkeit des Systems, auch wenn dieser etwas kleiner ausfällt.

Experiment 2.4: Variablenfixierung, heuristischer Vergleich In diesem Experiment wird die Benchmark CANAD-C heuristisch mit unterschiedlichen Konfigurationen des NDBC Solvers gelöst.

Experimentaufbau. Wie auch im exakten Vergleich untersuchen wir acht Varianten, die in der Tabelle 4.15 aufgeführt sind. Der Vergleich basiert auf der Lösungsqualität der Varianten.

Ergebnis. Die Linien der Verfahren liegen in der Grafik 4.15 sehr eng beieinander. Auch die durchschnittliche Lösungsqualität der Varianten **311**, **301** und **011** ist fast identisch. Eine deutliche Verbesserung der Lösungsqualität lässt sich bei der Hinzunahme der zusätzlichen Ungleichungen erzielen: von 1.4% auf 0.7%.

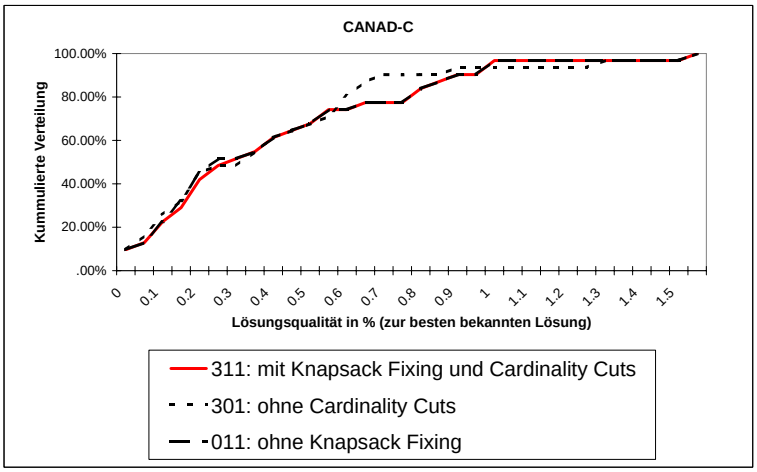
Auswertung. Die Knapsack-Fixierung und die Cardinality cuts zeigen in diesem Szenario praktisch keine Wirkung. Die positive Auswirkung der zusätzlichen Ungleichungen wird im nächsten Abschnitt untersucht.



3 - -	Knapsack fixing eingeschaltet
- 1 -	Cardinality cuts eingeschaltet
- - 1	Cover cuts und local cuts eingeschaltet

	Ø performance ratio	time	nodes
000	1.50	2945.60	26019
001	1.57	2710.18	23597
010	1.52	2506.03	23876
011	**1.52**	**3122.01**	**23911**
300	1.29	2458.66	19744
301	**1.30**	**2300.60**	**17660**
310	1.24	2625.22	18977
311	**1.26**	**2148.30**	**16383**

Abbildung 4.14: Experiment 2.3. Variablenfixierung - Knapsack Fixing (siehe auch 6.12, Seite 169).



3 - -	Knapsack fixing eingeschaltet
- 1 -	Cardinality cuts eingeschaltet
- - 1	Cover cuts und local cuts eingeschaltet

	Ø solution quality
000	1.38
001	0.69
010	1.41
011	**0.71**
300	1.38
301	**0.69**
310	1.41
311	**0.71**

Abbildung 4.15: Experiment 2.4. Variablenfixierung - Knapsack Fixing (siehe auch 6.13, Seite 170).

Auswirkungen der zusätzlichen Ungleichungen

In diesem Abschnitt untersuchen wir die Auswirkungen der zusätzlichen Ungleichungen, die im Kapitel 2.11 auf der Seite 58 präsentiert wurden. Die zwei Typen der Ungleichungen sind die Überdeckungsungleichungen (*cover inequalities*) und die lokalen Schnitte (*local cuts*). Diese zusätzlichen Ungleichungen bilden die Grundlage des Relax&Cut-Algorithmus für das Problem des Netzwerkentwurfs.

Wir präsentieren hier drei Experimente: Zwei in einem exakten Szenario und eines in einem heuristischen. Wir benutzen den NDBC Solver und bezeichnen dabei die unterschiedlichen Varianten wie folgt:

Konfiguration	Generierung von Ungleichungen
NDBC 00	keine zusätzlichen Ungleichungen,
NDBC 01	nur Überdeckungsungleichungen,
NDBC 10	nur lokale Schnitte,
NDBC 11	beide Typen von Ungleichungen.

Experiment 2.5: zusätzliche Ungleichungen, exakter Vergleich auf der PAD Benchmark
In diesem Experiment werden die vier Varianten des NDBC Solvers auf der PAD Benchmark miteinander verglichen.

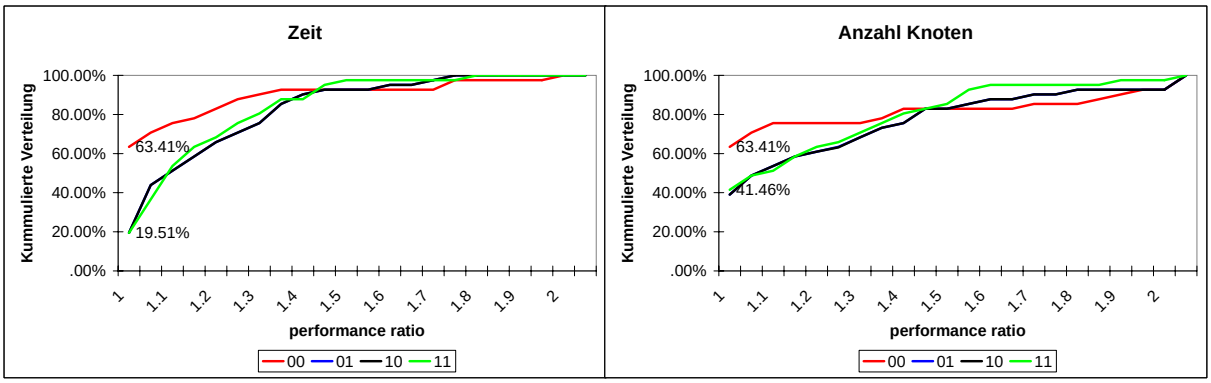
Experimentaufbau. In den Abbildungen 4.16 und 4.17 sind zwei unterschiedliche Szenarien dargestellt. Im ersten Szenario sollte der NDBC Solver sowohl die heuristische erste als auch die exakte zweite Phase des Relax&Cut-Algorithmus ausführen. Im zweiten Szenario wollten wir ohne eine gute Startlösung in die exakte Phase einsteigen und schalten deswegen die erste Phase ab. In diesem Fall vergrößert sich die Laufzeit der Verfahren, wir können aber genauer die Effekte durch die zusätzlichen Ungleichungen beobachten.

Ergebnis. In den Grafiken 4.16 und 4.17 sind die Leistungsprofile der vier Varianten dargestellt. Die Tabellen enthalten die durchschnittlichen Leistungswerte, Gesamtlaufzeit auf der Benchmark PAD und die Anzahl der Suchknoten im Relax&Cut-Algorithmus. Sowohl bei der Laufzeit als auch bei der Anzahl der Knoten dominiert die Variante ohne Ungleichungen (**NDBC 00**) alle andere Varianten. Auch die durchschnittlichen Leistungswerte sind bei **NDBC 00** besser. Alleine die Summe der Laufzeiten und der Gesamtanzahl der Knoten spricht für die Variante **NDBC 11**: 16% Zeitersparnis und 14% weniger Knoten.

Im zweiten Szenario ist die Dominanz der Variante ohne die Ungleichungen nicht so deutlich. Die Leistungsprofile der Laufzeit liegen eng zusammen, bei den Knoten ist die Variante **NDBC 11** deutlich besser. Auch bei den durchschnittlichen Leistungswerten liegt die Variante mit Ungleichungen leicht vorn: 1.13 gegen 1.14. Die Laufzeit konnte um 22% reduziert werden, die Anzahl der Knoten um 20%.

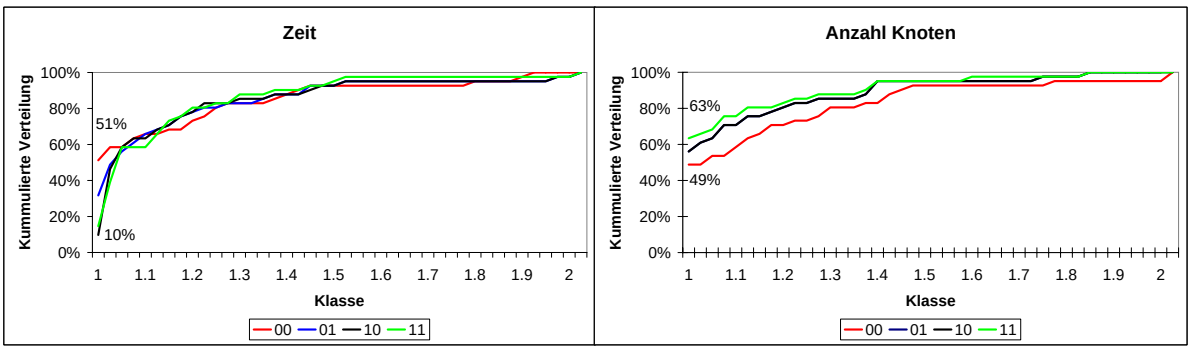
Auswertung. Auf der Benchmark PAD hat sich die Generierung der zusätzlichen Ungleichungen nicht gelohnt. Die Probleminstanzen können alle in einer relativ kurzen Zeit gelöst werden. Der zeitliche Aufwand für die Ungleichungen konnte zwar eine Reduzierung der Suchbaumgröße und der Gesamtlaufzeit mit sich bringen, die Leistungsprofile zeigen

aber, dass Variante ohne sie eindeutig im Vorteil liegt. Bei der größeren Probleminstanzen aus der Benchmark CANAD-R2 konnten wir ein ganz anderes Bild beobachten. Auf diese Ergebnisses gehen wir im nächsten Abschnitt ein.



	time	Ø performance ratio	nodes	Ø performance ratio
NDBC 00	2577.93	1.11	18977	1.22
NDBC 01	2141.81	1.17	16998	1.28
NDBC 10	2731.27	1.15	19444	1.30
NDBC 11	2167.99	1.16	16383	1.24

Abbildung 4.16: Experiment 2.5. Exakt (siehe auch 6.14, Seite 171).



	time	Ø performance ratio	nodes	Ø performance ratio
NDBC 00	9503.10	1.15	43977	1.19
NDBC 01	7368.19	1.14	35564	1.11
NDBC 10	9355.33	1.17	43987	1.19
NDBC 11	7355.54	1.13	35049	1.09

Abbildung 4.17: Experiment 2.5. Exakt (siehe auch 6.15, Seite 172).

Experiment 2.6: zusätzliche Ungleichungen, exakter Vergleich auf der CANAD-R2 Benchmark In diesem Experiment werden die vier Varianten des NDBC Solvers auf ausgewählten Instanzen der CANAD-R2 Benchmark miteinander verglichen.

Experimentaufbau. Untersuchte Probleminstanzen konnten alle in einem Zeitrahmen zwischen 2 Minuten und 4 Stunden gelöst werden. Damit wurde sichergestellt, dass die Suchbäume nicht zu klein werden. Auf diesen schweren Probleminstanzen wurden wie auch im vorigen Experiment die vier Varianten des NDBC-Solvers gestartet. Diese Experimente wurden auf einer anderen Rechnerarchitektur ausgeführt, sodass ein direkter Zeitvergleich zu anderen Experimenten nicht möglich ist.

Ergebnis. Die Abbildung 4.18(1) zeigt die Leistungsprofile der vier Varianten in logarithmischer Darstellung. Zunächst stellen wir fest, dass das Leistungsprofil der Variante **NDBC 11**, die beide Typen der zusätzlichen Ungleichungen generiert alle anderen Varianten deutlich dominiert. An zweiter Stelle liegt die Variante mit Überdeckungsungleichungen (ohne die lokalen Schnitte) **NDBC 01**. An letzter Stelle liegen die Varianten **NDBC 00** und **NDBC 10** mit einigen Ausreißern mit Leistungswerten von über 10. In 13 von 21 Fällen lieferte NDBC 11 das optimale Ergebnis am schnellsten.

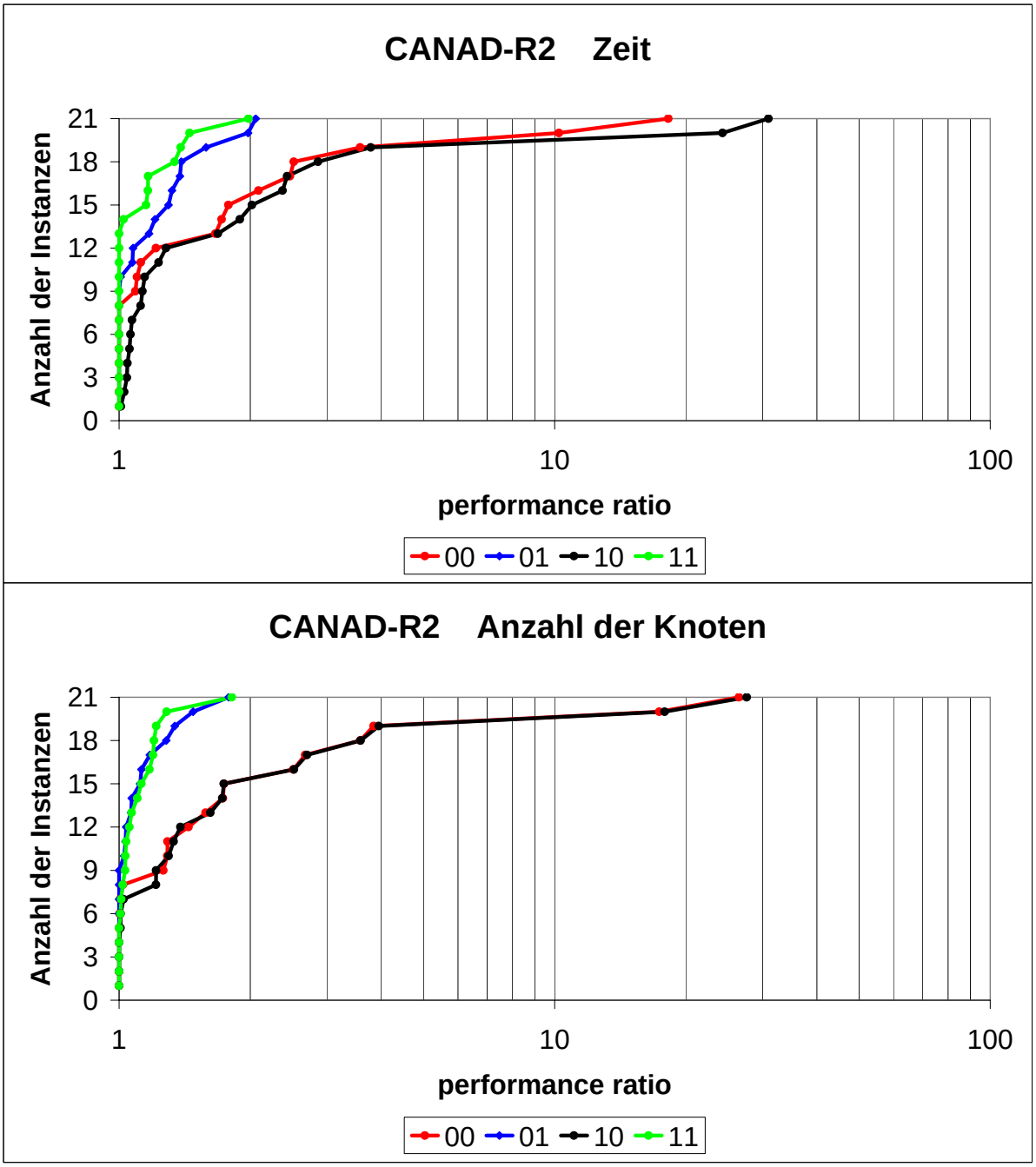
Bei der Anzahl der Knoten zeigt die Grafik 4.18(2) ein ähnliches Bild. Die Varianten **NDBC 11** und **NDBC 01** liegen klar vorne.

Die Gesamtlaufzeiten und Gesamtanzahl der Knoten der Varianten liegen weit auseinander. **NDBC 11** reduziert die Laufzeit von **NDBC 00** um 28%, die Anzahl der Knoten um fast 50%. Die durchschnittlichen Leistungswerte sind ebenfalls deutlich besser: 1.18 gegen 3.15 und 1.11 gegen 3.57.

Die Hinzunahme der lokalen Schnitte verschlechtern in diesem Experiment alle Kriterien (**NDBC 10** sogar schlechter als **NDBC 00**). Eine Beobachtung sollte man aber erwähnen: Die Variante nur mit Überdeckungsungleichungen **NDBC 01** ist schwächer als die Variante mit den zusätzlichen lokalen Schnitten **NDBC 11**. In diesem direkten Vergleich reduzieren die lokalen Schnitte die Gesamtlaufzeit um 6%. Die Leistungsprofile in der Grafik 4.18(1) belegen diese Verbesserung ebenfalls deutlich.

Auswertung. Die Ergebnisse auf den schwierigeren Instanzen der Benchmark CANAD-R2 sind sehr aussagekräftig. Die Generierung der zusätzlichen Überdeckungsungleichungen reduziert die Laufzeiten deutlich und halbiert sogar die Anzahl der Suchknoten im B&B-Baum. Die Leistungsprofile belegen, dass diese Variante stets die schnellere war. Die lokalen Schnitte sind für sich alleine nicht lohnenswert, verbessern aber das Gesamtergebnis noch deutlich.

Im nächsten Abschnitt untersuchen wir die Leistungsfähigkeit der zusätzlichen Ungleichungen in einem heuristischen Vergleich auf der Benchmark CANAD-C.



	time	Ø performance ratio	nodes	Ø performance ratio
NDBC 00	65752.58	3.15	272377	3.57
NDBC 01	50308.22	1.22	137039	1.12
NDBC 10	68700.64	4.07	275880	3.66
NDBC 11	47274.12	1.18	137707	1.11

Abbildung 4.18: Experiment 2.6. Exakt, Zeiten auf dem PSC2-Cluster (Pentium 3, 850 MHz) (siehe auch 6.16, Seite 173).

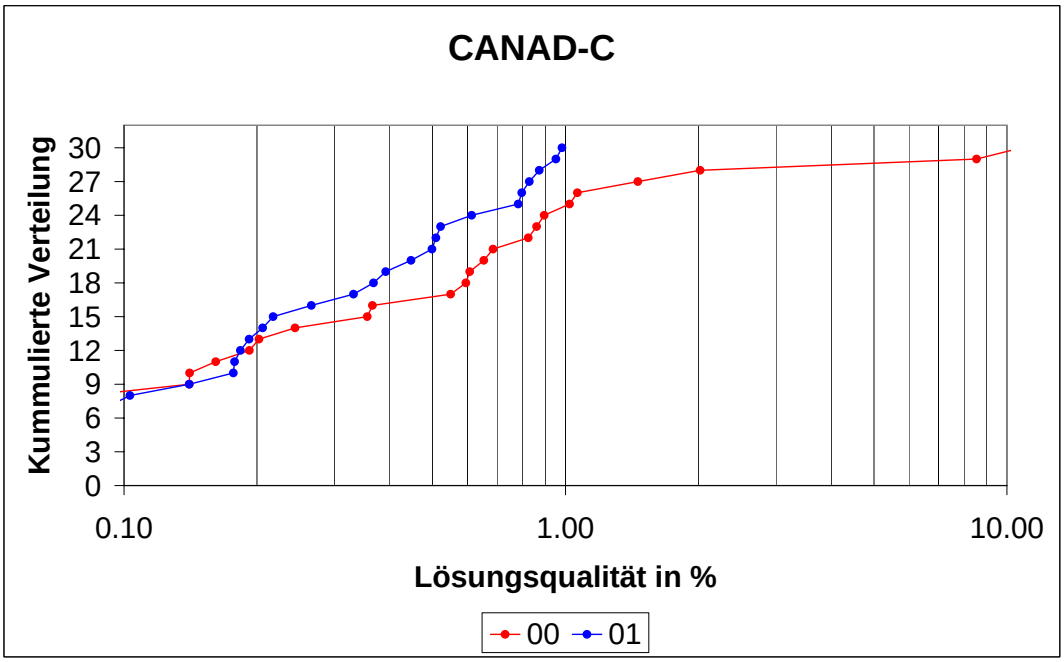
Experiment 2.7: zusätzliche Ungleichungen, heuristischer Vergleich Dieses Experiment soll die Leistungsfähigkeit der zusätzlichen Ungleichungen in einem heuristischen Szenario untersuchen. Wir startet dazu jeweils vier Varianten des Alpha-NDBC und des Beta-NDBC Solvers auf der Benchmark CANAD-C.

Experimentaufbau. Für jede Probleminstanz wurden 300 Sekunden zur Verfügung gestellt. Die Lösungsqualität dient als Grundlage für die kumulierte Verteilungsfunktion.

Ergebnis. In der Grafik 4.19 sind die Linien der zwei Varianten des Alpha-NDBC Solvers zu sehen: **NDBC 00** und **NDBC 01**. Die Variante **NDBC 10** fällt zusammen mit der Linie **NDBC 00** und die Linien von **NDBC 01** und **NDBC 11** verlaufen ebenfalls identisch. Wir sehen eine deutlich bessere Lösungsqualität bei der Variante mit Überdeckungsungleichungen (**NDBC 01**).

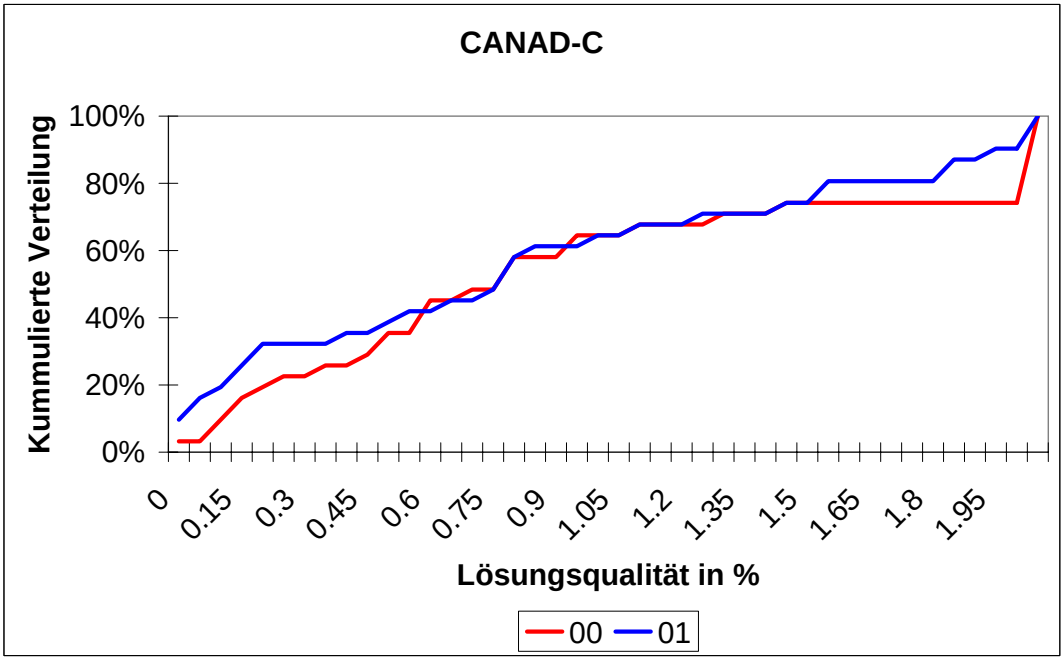
In der Grafik 4.20 sind die Ergebnisse der Beta-Fixierung dargestellt. Die Variante **NDBC 01** hat auch hier einen Vorteil gegenüber der Variante ohne zusätzliche Ungleichungen.

Auswertung. Die Überdeckungsungleichungen konnten in diesem Experiment eine bessere Lösungsqualität liefern. Die lokalen Schnitte haben keinerlei Verbesserung gebracht. Der Grund dafür liegt darin, dass die heuristischen Variablenfixierungen bereits sehr stark eingreifen und sehr viele Variablen fixieren. Die lokalen Schnitte stellen eine Verallgemeinerung der Knapsack-Variablenfixierung dar und konnten keine zusätzlichen Teilbäume in der B&B-Suche abschneiden.



	Ø solution quality
NDBC 00	1.41
NDBC 01	0.71
NDBC 10	1.41
NDBC 11	0.71

Abbildung 4.19: Experiment 2.7. Heuristisch (siehe auch 6.17, Seite 174).



	Ø solution quality
NDBC OO	1.41
NDBC O1	0.88
NDBC 1O	1.41
NDBC 11	0.88

Abbildung 4.20: Experiment 2.7. Heuristisch (siehe auch 6.18, Seite 175).

Experiment 2.8: Variation der α- und β-Werte

In diesem Abschnitt möchten wir die heuristischen Variablenfixierungen genauer untersuchen, die im Abschnitt 2.10 auf der Seite 56 vorgestellt wurden. Wir starten dazu die heuristischen Varianten Alpha-NDBC und Beta-NDBC auf der Benchmark CANAD-C.

Experimentaufbau. Für jede Instanz wird ein Zeitlimit von 300 Sekunden gesetzt. Wir protokollieren für jeden Lauf die Gesamtlaufzeit auf der Benchmark und die durchschnittliche Lösungsqualität. Die Alpha-Heuristik wird für unterschiedliche Werte von α gestartet: Insgesamt 40 Läufe für Werte zwischen 0.01 und 0.4. Genauso wird auch der Parameter β variiert.

Ergebnis. Die Ergebnisse sind entsprechend in der Grafiken 4.21 dargestellt. Auf der x-Achse ist der eingestellte Wert abgetragen. Auf der linken (primären) y-Achse ist die verbrauchte Gesamtzeit für alle Instanzen angezeigt. Auf der rechten (sekundären) y-Achse ist die durchschnittliche Lösungsqualität zu sehen. Die durchgezogene rote Linie entspricht dabei der Laufzeit, die blauen Quadrate der durchschnittlichen Lösungsqualität und blauen Dreiecke der Summe der durchschnittlichen Lösungsqualität und der Standardabweichung (bei 31 Instanzen). In der Grafik 4.21(2) sind die blauen Punkte zur besseren Übersicht mit einer Linie verbunden. Die Standardwerte der Parameter α und β sind in den Grafiken mit einem roten Quadrat und einer roten Verbindung nach ober markiert.

Bei der Alpha-Fixierung beobachten wir eine relativ schlechte Lösungsqualität bei Werten von α zwischen 0.01 und 0.2 sowie einen starken Einbruch der Qualität bei Werten größer von 0.3. Der stabile Bereich liegt zwischen 0.23 und 0.29: es gibt dort fast keine Ausreißer in der Lösungsqualität. Die Laufzeit bei der Standardeinstellung von 0.25 liegt in etwa im Durchschnitt (4500 Sekunden).

Die Beta-Fixierung ist von der Definition her etwas aggressiver als die Alpha-Fixierung. Es werden in jeder Ebene des B&B-Baums eine vorgegebene Anzahl von Variablen fixiert. Dementsprechend instabil sind auch die beobachteten Ergebnisse. Die Laufzeit fällt stetig bei Vergrößerung des β-Wertes. Die Lösungsqualität ist im Intervall zwischen 0.08 und 0.15 sehr gut. Der stabile Bereich ohne Ausreißer befindet sich in etwa zwischen 0.1 und 0.14. Der Standardwert für β in unserem System liegt bei 0.25.

Auswertung. Bei diesem Experiment lassen sich sehr gut die Vorteile der heuristischen Variablenfixierung beobachten. Die Laufzeit wird deutlich reduziert und die Lösungsqualität entscheidend verbessert. Dieses Ergebnis konnten wir bereits in früheren Experimenten feststellen. Weiterhin stellen wir fest, dass die Alpha-Fixierung deutlich stabiler in Bezug auf die Lösungsqualität ist als die Beta-Fixierung. Mit Hilfe dieses Experiments (und noch weiterer) konnten die Standardwerte für Parameter α und β gewählt werden, sodass das System für den Netzwerkentwurf zuverlässig arbeitet und gute Lösungen liefern kann.

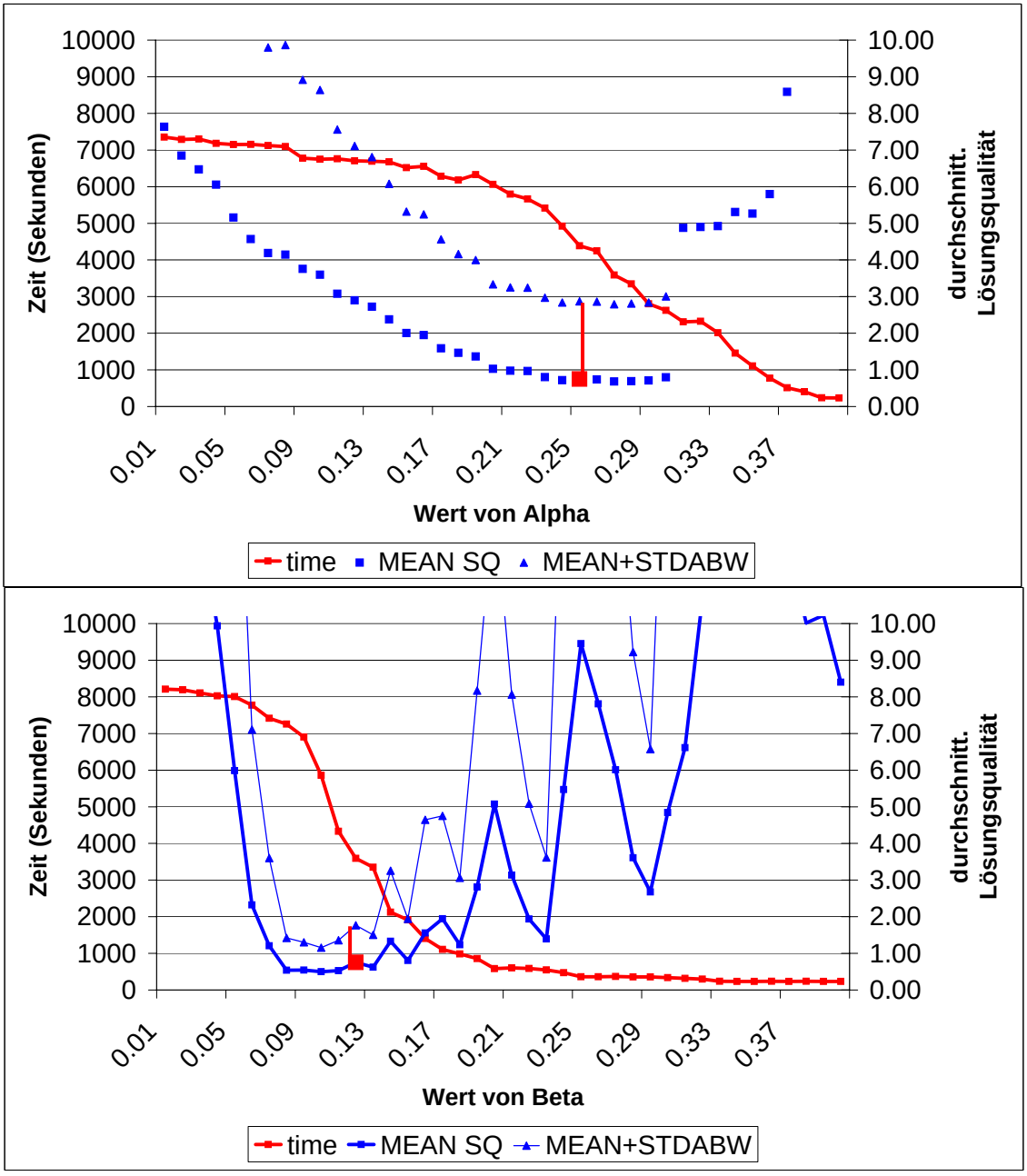


Abbildung 4.21: Experiment 2.8 (siehe auch 6.19, Seite 176, und 6.20, Seite 177).

4.1.5 Zusammenfassung der Ergebnisse

In diesem Abschnitt fassen wir die Erkenntnisse der experimentellen Untersuchungen zusammen. Die Abbildungen 4.7 und 4.8 geben einen Überblick über die ausgeführten Experimente.

Nr.	Benchmark	Verfahren im Vergleich	Art des Vergleichs
1.1	PAD	NDBB, **NDBC**, CPLEX 9.0	exakt
1.2	CANAD-R2	NDBC, **CPLEX 9.0**	exakt
1.3	CANAD-R2	**NDBC**, CPLEX 9.0, Kanadische Solver	heuristisch
1.4	CANAD-C	**NDBC-α**, NDBC, CPLEX 9.0, Kanadische Solver	heuristisch
1.5	CANAD-C	NDBB-α, NDBB-β, **NDBC-α**, NDBC-β	heuristisch
1.6	PAD-S	NDBC-α, NDBC	heuristisch

Tabelle 4.7: Vergleiche mit anderen Verfahren

Im Experiment 1.1 konnte der Solver NDBC als bester exakter Solver identifiziert werden.

Das Experiment 1.2 zeige CPLEX ein besseres Verhalten als der NDBC Solver.

Bei den Experimenten 1.3 und 1.4 lieferte der NDBC (bzw. Alpha-NDBC) Solver bessere Ergebnisse als alle anderen untersuchten Solver.

Im Experiment 1.5 zeigte sich der Alpha-NDBC Solver als der Solver mit der besten Lösungsqualität.

Das Experiment 1.6 zeigte, dass auch auf der größten Benchmark unser System sehr gute Lösungen liefern kann.

Nr.	Benchmark	Komponente	Art des Vergleichs
2.1	CANAD-C	Erste Phase	heuristisch
2.2	PAD	Primale Heuristik	exakt
2.3	PAD	Variablenfixierung	exakt
2.4	CANAD-C	Variablenfixierung	heuristisch
2.5	PAD	Zusätzliche Ungleichungen (Cuts)	exakt
2.6	CANAD-R2	Zusätzliche Ungleichungen (Cuts)	exakt
2.7	CANAD-C	Zusätzliche Ungleichungen (Cuts)	heuristisch
2.8	CANAD-C	α und β Werte	heuristisch

Tabelle 4.8: Untersuchung der Systemkomponenten

Die unterschiedlichen Komponenten des Systems wurden in acht weiteren Experimenten untersucht.

Die erste heuristische Phase im Branch-and-bound Algorithmus ist notwendig zum Erreichen einer guten Performance (Experiment 2.1). Die Häufigkeit der primalen Heuristik wurde im Experiment 2.2 variiert und ein stabiler Standardwert auf dieser Grundlage gewählt. Die Experimente 2.3 und 2.4 untersuchten die Auswirkungen der Variablenfixierung. Im exakten Fall erreicht die Variablenfixierung eine starke Reduktion der Laufzeiten. Im heuristischen Fall zeigt diese Komponente praktisch keine Wirkung. Die Leistungsfähigkeit der zusätzlichen Ungleichungen wurde in exakten Experimenten 2.5 und 2.6 untersucht. Auf der Benchmark PAD

hat sich der zusätzliche Aufwand für die Generierung der Ungleichungen nicht ausgezahlt. Ein deutlich besseres Ergebnis konnte auf der größeren Benchmark CANAD erzielt werden: Sowohl die Laufzeit als auch die Größe des Suchbaums konnten sehr stark verringert werden. Im heuristischen Vergleich 2.7 konnten die Überdeckungsungleichungen als hilfreich identifiziert werden, die lokalen Schnitte zeigten keine Wirkung. Durch die Variation der α- und β-Werte konnten wir im Experiment 2.8 feststellen, dass die α-Variablenfixierung deutlich stabiler ist. Die Leistungsfähigkeit der $\beta-$Heuristik ist entsprechend ihrer sehr aggressiven Art der Variablenfixierung sehr stark vom Wert β abhängig.

Erkenntnisse aus den durchgeführten Experimenten erlauben es, eine Konfiguration des Systems zu definieren, die als Standardkonfiguration verwendet werden kann. Diese Konfiguration, sowohl im exakten als auch im heuristischen Fall, liefert Lösungen mit einer hohen Qualität und verhält sich in unterschiedlichen Szenarien des Netzwerkentwurfs stabil und zuverlässig.

Als Ergebnis können wir ein sehr leistungsfähiges System für den Netzwerkentwurf vorweisen. Bei der exakten Lösung der Netzwerkentwurfprobleme zeigen unsere Experimente eine bessere Performance als das auf der Subgradient-Suche basierte System und auch als der Branch-and-Cut Algorithmus von CPLEX 9.0. Das System konnte auch andere heuristische Lösungsverfahren für das Problem des Netzwerkentwurfs in der Qualität der gelieferten Lösungen und der Laufzeit übertreffen.

4.2 Flottenzuweisung

4.2.1 Beschreibung der Datensätze

Für Experimente in diesem Abschnitt wählten wir zwei Datensätze von Fluggesellschaften, die im Folgenden beschrieben werden. Die Datensätze werden stellvertretend für eine Reihe von Szenarien benutzt, die bei mehreren Kunden von Lufthansa Systems ausgewertet wurden. Eingangs sei angemerkt, dass die Experimente in diesem Kapitel nicht in der Ausführlichkeit durchgeführt werden, wie die Experimente im Kapitel 4.1. Wir beschränken uns auf einige typische Experimente, die die Leistungsfähigkeit der entwickelten Integrationsstrategien untersuchen.

Zunächst beschreiben wir die zugrunde liegenden Netzwerke und die Passagierdaten, die für die Experimente benutzt wurden.

Anzahl	Datensatz A	Datensatz B
Flughäfen	97	160
Flüge	6287	9228
Märkte	20358	6680
Itineraries (Reiseverbindungen)	34963	160635
Direkte itineraries	6740	14240
Single-connections (1 Zwischenstopp)	26573	140160
Double-connections (2 Zwischenstopps)	1650	5440

Der Datensatz A basiert auf einem kleineren Netzwerk als der Datensatz B, es wurden aber mehr Märkte berücksichtigt. Die Anzahl der Reiseverbindungen hängt stark von den Einstellungen des Marktmodells ab.

Als Nächstes geben wir die Eigenschaften der Passagiernachfrage an. Dazu definieren wir einige Parameter, die Aufschluss über die vorliegenden Datensätze liefern.

Eingabedaten und Parameter

$\mathscr{A}$	Menge der Flughäfen
$\mathscr{E}$	Flugkanten des Netzwerks
$OD \subseteq \mathscr{A}^2$	Menge aller Passagier-Märkte
P	Die Menge der möglichen Itineraries
P_c	Die Menge der Itineraries, die über mindestens eine Zwischenstation führen
P^{od}	Mögliche Itineraries für Passagiere aus dem Markt od
d_p	Geschätzte Anzahl der Passagiere auf der Itinerary p
$(\delta_{lp} = 1) \Leftrightarrow$	Itinerary p benutzt die Kante $l \in \mathscr{E}$
c_l	Kapazität der Kante l
x_p^{od}	Anzahl der Passagiere aus dem Markt od auf dem Pfad $p \in P^{od}$

Um die Höhe des Nachfrage zu beschreiben, wird der Nachfragefaktor (*Demand Factor*) folgendermaßen definiert:

Definition 4.5 (Demand Factor) *Der Demand Factor gibt das Verhältnis der angefragten Passagierplätze zu der Kapazität des Netzwerks an:*

$$DF = \frac{\sum_{p \in P} d_p \sum_{l \in \mathscr{E}} \delta_{lp}}{\sum_{l \in \mathscr{E}} c_l}$$

Der geschätzte Bedarf auf einer Itinerary wird mit der Anzahl der Kanten in der Itinerary multipliziert, um die Gesamtanzahl der benötigten Passagierplätze zu ermitteln. Die Gesamtkapazität des Netzwerks berechnet sich als die Summe der Kapazitäten über alle Kanten.

Die Auslastung des Netzwerks (*System Load Factor*) gibt an, wie viele der potentiellen Passagiere einen Platz auf den Flugstrecken bekommen haben. Die Auslastung dient oft als Maß für die Qualität der Kapazitätssteuerung durch das Revenue Management und die Flottenzuweisung. Typische Werte, die in der Praxis gemessen werden, liegen bei ca. 70%.

Definition 4.6 (System Load Factor) *Der System Load Factor ist das Verhältnis der Anzahl der insgesamt verkauften Passagierplätze zu der Gesamtkapazität des Netzwerks:*

$$SLF = \frac{\sum_{p \in P^{od}} x_p^{od} \sum_{l \in \mathscr{L}} \delta_{lp}}{\sum_{l \in \mathscr{L}} c_l}$$

Eine weitere wichtige Kennzahl, die *Passenger Connectivity Ratio*, gibt an, wie groß der Anteil der Passagiere ist, die über mehrere Zwischenstationen zu ihrem Zielflughafen reisen.

Definition 4.7 (Passenger Connectivity Ratio) *Das Passenger Connectivity Ratio ist das Verhältnis der Anzahl der Verbindungs-Passagiere zu der Gesamtanzahl der Passagiere im Netzwerk:*

$$PCR = \frac{\sum_{p \in P_c} d_p}{\sum_{p \in P} d_p}$$

Gerade die Passagiere, die nicht direkt fliegen, sondern umsteigen, verursachen die zuvor beschriebenen Netzwerkeffekte, die es zu berücksichtigen gilt. Der Wert PCR in einem Netzwerk gibt im Wesentlichen an, wie häufig die im Abschnitt 3.5.1 beschriebenen Netzwerkeffekte auftreten können.

Folgende Tabelle enthält Angaben zu diesen drei Kennzahlen, die bei den beiden untersuchten Datensätzen festgestellt wurden.

Kennzahl	Datensatz A	Datensatz B
Demand Factor	3.04	0.56
System Load Factor	79%	63%
Passenger Connectivity Ratio	0.57	0.36

Der Datensatz A besitzt einen deutlich höheren Demand Factor als der Datensatz B. Dies ist auch der wesentliche Grund für die höhere Netzwerkauslastung: 79% im Vergleich zu 63%. Der Anteil der Verbindungs-Passagiere ist ebenfalls deutlich größer: 0.57 gegen 0.36.

Eine weitere Untersuchung gibt eine detailliertere Auskunft über die Wichtigkeit der Netzwerkeffekte bei den vorliegenden Datensätzen. Betrachtet man die Kapazitäten der möglichen Flugzeugtypen auf einer Flugstrecke und die geschätzte Anzahl der Passagiere auf dieser Flugstrecke, so lassen sich vier Klassen von Flugstrecken definieren:

- *uncapacitated*: Die Nachfrage auf der Flugstrecke liegt unterhalb der kleinsten Kapazität der möglichen Flugzeugtypen.
- *capacitated*: Die Nachfrage auf der Flugstrecke liegt oberhalb der kleinsten Kapazität der möglichen Flugzeugtypen.

- *potentially capacitated*: Die Nachfrage auf der Flugstrecke liegt oberhalb der kleinsten Kapazität der möglichen Flugzeugtypen, aber unterhalb der größten möglichen Kapazität auf der Strecke.

- *overcapacitated*: Die Nachfrage auf der Flugstrecke liegt oberhalb der größten Kapazität der möglichen Flugzeugtypen.

Die Menge der *capacitated* Flugstrecken ist die Vereinigung der beiden Mengen *potentially capacitated* und *overcapacitated*.

Flugstrecken	Datensatz A	Datensatz B
uncapacitated	1312 (21%)	2835 (30%)
potentially capacitated	1588 (25%)	3805 (41%)
overcapacitated	3345 (54%)	2591 (29%)

Auch in dieser Tabelle kann beobachtet werden, dass der Datensatz A deutlich mehr stark ausgelastete Flugstrecken besitzt als der Datensatz B.

4.2.2 Ergebnisse der Integrationsstrategien

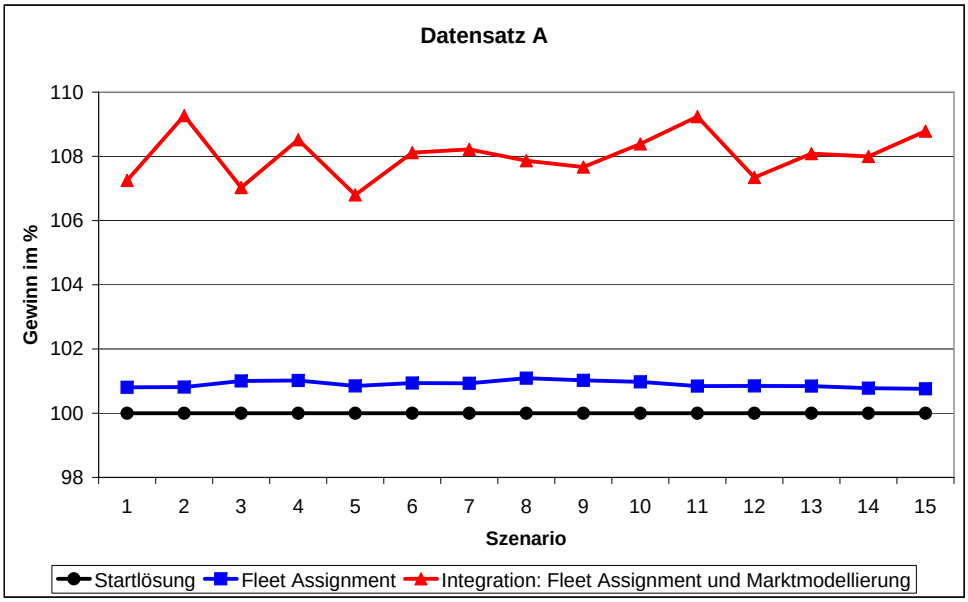
In der Abbildung 4.22 sind 15 Szenarien mit der ersten Integrationsstrategie dargestellt. Der Gewinnwert der Startlösung wird in diesem Experiment mit 100% angegeben. Das Ergebnis der Flottenzuweisung ohne die Interaktion mit dem Marktmodell (*Fleet Assignment*) erzielt im Durchschnitt eine Verbesserung von knapp einem Prozent gegenüber der Startlösung. Die in dieser Arbeit vorgestellte erste Integrationsstrategie erzielt eine durchschnittliche Verbesserung von 8%. Die absoluten Gewinnzahlen können nicht angegeben werden, weil sie eine vertrauliche Information der jeweiligen Fluggesellschaft darstellen.

Die zweite Integrationsstrategie implementiert das Modell des Passagierflusses (*PFM: Passenger Flow Model*) als eine zusätzliche Komponente an der Schnittstelle zwischen der Marktmodellierung und der Flottenzuweisung. Das lineare Programm steuert die Zielfunktion der Flottenzuweisung mittels der dualen Werte der Kapazitätsrestriktionen der Flugstrecken (siehe Abschnitt 3.6).

In der Abbildung 4.23 ist das Konvergenzverhalten der Integration des Modells des Passagierflusses (*PFM*) mit der Flottenzuweisung auf dem Datensatz B dargestellt.

Im oberen Bild der Abbildung 4.23 sind die dualen Werte über die gesamte Laufzeit zu sehen. Bei den ersten Iterationen sind die dualen Werte relativ groß und steuern den Simulated Annealing Algorithmus entsprechend stark an. Zum Ende der Berechnung werden die dualen Werte deutlich kleiner. Die Kapazitätszuweisung im Fleet Assignment wurde an den vorhergesagten Fluss angepasst. Die Erhöhung der Kapazitäten auf den entsprechenden Flugstrecken würde keinen größeren Zugewinn mehr bedeuten.

Im unteren Bild der Abbildung 4.23 wurde die Differenz in der Zielfunktion des Simulated Annealing Algorithmus jeweils vor und nach einer Iteration mit dem *PFM* protokolliert. Auch hier ist die Konvergenz des Verfahrens deutlich zu sehen. Die Bewertung der aktuellen Lösung durch das System der Flottenzuweisung und das Modell des Passagierflusses nähern sich im Verlauf der Optimierung stark an.



			Integration:
Szenario	Startlösung	Fleet Assignment	Fleet Assignment und Marktmodellierung
1	100%	100.8%	107.3%
2	100%	100.8%	109.3%
3	100%	101.0%	107.0%
4	100%	101.0%	108.5%
5	100%	100.9%	106.8%
6	100%	100.9%	108.1%
7	100%	100.9%	108.2%
8	100%	101.1%	107.9%
9	100%	101.0%	107.7%
10	100%	101.0%	108.4%
11	100%	100.8%	109.2%
12	100%	100.8%	107.3%
13	100%	100.8%	108.1%
14	100%	100.8%	108.0%
15	100%	100.8%	108.8%
Mittelwert		100.9%	108.0%

Abbildung 4.22: Ergebnisse der ersten Integrationsstrategie

4.2.3 Zusammenfassung und Ausblick

Anhand der Experimente kann festgestellt werden, dass die Integration der Phasen der Marktmodellierung und der Flottenzuweisung erfolgreich durchgeführt wurde. Die implementierte Strategie wird bei mehreren Fluggesellschaften im Produktionsbetrieb eingesetzt und leistet einen Beitrag zur Steigerung der Profitabilität der Flugpläne.

Die zweite Integrationsstrategie wurde als ein Forschungsprototyp implementiert und mit realen Daten der Fluggesellschaften getestet. Die Übernahme der Erkenntnisse in die Planungssysteme der Fluggesellschaften steht noch bevor.

Die dritte Integrationsstrategie wird in naher Zukunft im Rahmen einer umfangreicheren Studie evaluiert. Die hohen Aufwände für die Anpassung des Revenue Management Systems konnten im Rahmen dieser Arbeit nicht geleistet werden. Eine Auswertung der Ergebnisse dieser Integrationsstrategie steht bevor. Die vorbereitenden Analysen seitens der Lufthansa Systems lassen auf ein hohes Potential der vorgeschlagenen Vorgehensweise schließen.

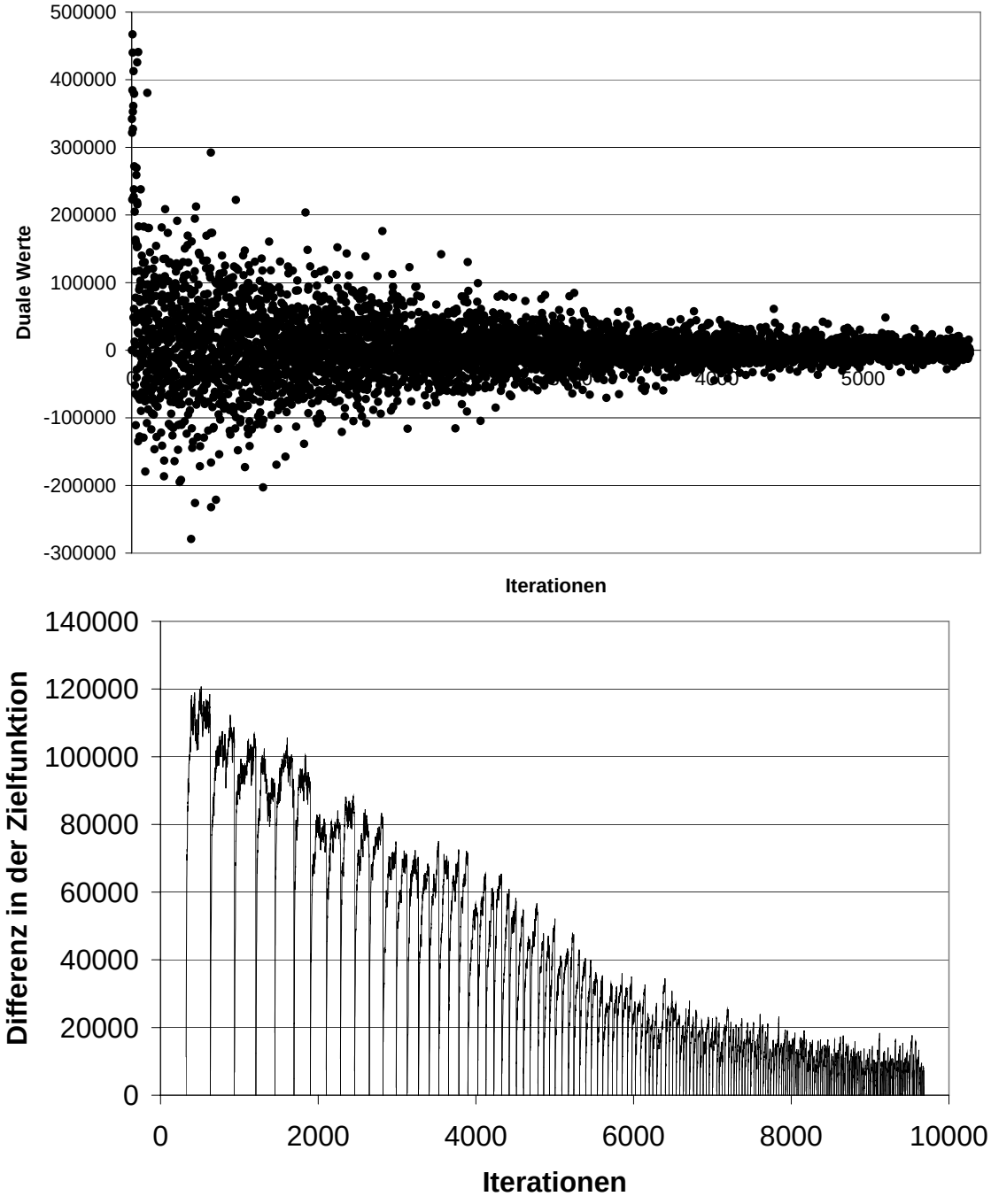


Abbildung 4.23: Konvergenz der dualen Werte und der Differenz in der Zielfunktion

— 5 –

Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde die Optimierung in der Flugplanung untersucht. Wir entwickelten effiziente Algorithmen zur Lösung der zugrunde liegenden Optimierungsprobleme und untersuchten die Leistungsfähigkeit der Verfahren. Die Ergebnisse der Arbeit kommen in Entscheidungsunterstützungssystemen zum Einsatz, verbessern die Qualität der berechneten Lösungen und leisten auf diese Weise einen Beitrag zur Kostensenkung bei Fluggesellschaften.

Wir beschäftigten uns zunächst mit dem Problem des Netzwerkentwurfs. Das zugrunde liegende lineare ganzzahlige Optimierungsproblem zeichnet sich durch seine hohe Komplexität aus. Zur Lösung dieses Problems entwickelten und implementierten wir mehrere Algorithmen. Das entstandene Optimierungssystem beinhaltet sowohl exakte als auch heuristische Lösungsverfahren.

Dafür wurde auf der Basis einer Lagrange-Relaxation ein Branch-and-bound und ein Relax-and-cut-Algorithmus entwickelt. Das Lagrange-Multiplikator-Problem wird mit Hilfe des von uns entwickelten Subgradientenverfahrens oder alternativ durch die von A. Frangioni implementierte Bundle-Methode gelöst.

Wir implementierten mehrere aus der Literatur bekannte Baumsuchverfahren, Branching-Strategien und Algorithmen zur Bestimmung von zulässigen Lösungen innerhalb der Baumsuche.

Weiterhin untersuchten wir Strategien zur Variablenfixierung. Neben klassischen Strategien wurden erweiterte kombinierte Verfahren entwickelt, die zwei Arten von Lagrange-Relaxationen verbinden.

Es wurden heuristische Variablenfixierungen für das Subgradientenverfahren aus der Literatur untersucht. Wir erweiterten diese Strategien so, dass sie auch im Zusammenspiel mit der Bundle-Methode eingesetzt werden können.

Einen wichtigen Beitrag stellt auch die Entwicklung des Relax-and-cut-Algorithmus dar. Wir untersuchten bekannte zulässige Ungleichungen (*cuts*), wie z.B. die Überdeckungsungleichungen,

und implementierten effiziente Algorithmen zum Finden und Verbessern dieser Ungleichungen.

Die neu entwickelten lokalen Schnitte stellen eine Verallgemeinerung der Variablenfixierung dar. Sie erlauben es, zusätzliche gültige Ungleichungen zu finden, die mehrere Variablen beinhalten und auf diese Weise die Baumsuche deutlich beschleunigen.

Ein wesentlicher Beitrag dieser Arbeit ist auch die implementierte dynamische Verwaltung der zusätzlichen Ungleichungen im Relax-and-cut-Algorithmus. Sie würden die Struktur der Unterprobleme zerstören und müssen deswegen mit neuen Lagrange-Multiplikatoren in die Zielfunktion aufgenommen werden.

Die experimentellen Ergebnisse belegen die Leistungsfähigkeit des Systems. Hervorheben möchten wir die Überlegenheit gegenüber anderen Lösungsverfahren, die in der Literatur entwickelt wurden, und auch gegenüber einem Standard-Solver für Optimierungsprobleme CPLEX.

Die einzelnen Komponenten können wie folgt bewertet werden: Der Relax-and-cut-Algorithmus profitiert stark von den zusätzlichen Ungleichungen und übertrifft dadurch den Branch-and-bound Algorithmus. Die Bundle-Methode verhält sich besser als das Subgradientenverfahren.

Die Strategien zur Variablenfixierung liefern eine deutliche Verkürzung der Laufzeiten. Besonders die heuristischen Variablenfixierungen erlauben eine sehr schnelle Lösung auch von größeren Probleminstanzen und liefern Lösungen von hoher Qualität.

Insgesamt wurde im Rahmen dieser Arbeit ein leistungsstarkes Optimierungssystem für den Netzwerkentwurf entwickelt, das in vielen Anwendungsfällen zum Einsatz kommen kann.

Den zweiten Schwerpunkt dieser Arbeit bildet die Planungsaufgabe der Flottenzuweisung. Diesem Planungsschritt kommt eine zentrale Bedeutung innerhalb einer Fluggesellschaft zu, weil hier die wesentlichen Eigenschaften des Flugplans festgelegt werden und dadurch die Betriebskosten definiert werden. Eine verbesserte Flottenzuweisung erhöht deutlich die Profitabilität eines Flugplans.

Die benachbarten Aufgaben der Marktmodellierung und des Revenue Managements sind stark mit der Flottenzuweisung verzahnt. Wir untersuchen die Abhängigkeiten zwischen diesen Planungsschritten und stellen einen großen Verbesserungspotential für eine integrierte Planung fest.

Im Rahmen dieser Arbeit wurden drei Integrationsstrategien entwickelt, die zum Teil bereits im industriellen Einsatz sind.

Die erste Strategie verbindet die Phasen der Marktmodellierung und der Flottenzuweisung. Die Kommunikation erfolgt durch den Austausch der Lösungen und die Anpassung der Zielfunktion in der Flottenzuweisung.

Die zweite Strategie implementiert ein Modell des Passagierflusses und erlaubt dadurch eine schnellere Interaktion mit der Flottenzuweisung. Die dualen Werte der Kapazitätsrestriktionen liefern wichtige Informationen über den Passagierfluss und ermöglichen eine genauere Anpassung der Flottenzuweisung an die Passagiernachfrage.

Die dritte vorgeschlagene Strategie kann in der Kurzfristplanung eingesetzt werden und verbindet das Revenue Management System mit der Flottenzuweisung. Durch die genauen Prognosen der Passagierzahlen im Netzwerk und eine viel detailliertere Schätzung der erzielbaren Erlöse wird die Lösung des Flottenzuweisungsproblems gesteuert. Eine kurzfristige Anpassung der Kapazitäten im Netzwerk an die vorliegende Nachfrage ermöglicht einen kostenminimalen Einsatz der Flugzeuge und einen gesteigerten Gewinn durch zusätzlich transportierte Passagiere.

Ausblick

Die Erkenntnisse aus dieser Arbeit führen uns zu neuen Fragestellungen und neuen Anwendungsmöglichkeiten:

- Erweiterung der Problemdefinition des Netzwerkentwurfs um zusätzliche Restriktionen oder Freiheitsgrade. Die Leistungsfähigkeit der entwickelten Verfahren kann in weiteren komplexeren Szenarien unter Beweis gestellt werden.
- Untersuchung weiterer Klassen von zusätzlichen Ungleichungen, die die Lagrange-Schranken noch weiter verbessern können.
- Ein exaktes Verfahren für die Flottenzuweisung kann durch seine verbesserte Leistungsfähigkeit in unseren Integrationsstrategien an die Stelle des Simulated Annealing Algorithmus treten.
- Definition eines integrierten mathematischen Modells für die Marktmodellierung und die Flottenzuweisung kann angesichts der Fortschritte im Bereich der ganzzahligen Optimierung eine weitere Möglichkeit darstellen.
- Die Integration der Flottenzuweisung mit dem Revenue Management System muss weiter untersucht werden. Eine offene Frage ist z.B., ob ein Konvergenzverhalten bei der Optimierung auch in diesem Fall erreicht werden kann.

6

Anhang: Tabellen zu den experimentellen Ergebnissen

Nr.	Benchmark	Verfahren im Vergleich	Art des Vergleichs
1.1	PAD	NDBB, NDBC, CPLEX 9.0	exakt
1.2	CANAD-R2	NDBC, CPLEX 9.0	exakt
1.3	CANAD-R2	NDBC, CPLEX 9.0, Kanadische Solver	heuristisch
1.4	CANAD-C	NDBC-α, NDBC, CPLEX 9.0, Kanadische Solver	heuristisch
1.5	CANAD-C	NDBB-α, NDBB-β, NDBC-α, NDBC-β	heuristisch
1.6	PAD-S	NDBC-α, NDBC	heuristisch

Tabelle 6.1: Experimente zur Untersuchung der Leistungsfähigkeit der Verfahren

Nr.	Benchmark	Komponente	Art des Vergleichs
2.1	CANAD-C	Erste Phase	heuristisch
2.2	PAD	Primale Heuristik	exakt
2.3	PAD	Variablenfixierung	exakt
2.4	CANAD-C	Variablenfixierung	heuristisch
2.5	PAD	Zusätzliche Ungleichungen (Cuts)	exakt
2.6	CANAD-R2	Zusätzliche Ungleichungen (Cuts)	exakt
2.7	CANAD-C	Zusätzliche Ungleichungen (Cuts)	heuristisch
2.8	CANAD-C	α und β Werte	heuristisch

Tabelle 6.2: Experimente zur Untersuchung der Effizienz der Systemkomponenten

	Knoten	Kanten	Commodities	fixvar	ctight
A1	12	50	50	0.02	1.5
A2	12	50	50	0.04	1.5
A3	12	50	50	0.02	1
A4	12	50	50	0.04	1
B1	12	100	50	0.03	1
B2	12	100	50	0.06	1
B3	12	100	50	0.03	1
B4	12	100	50	0.06	1
C1	12	50	100	0.06	1
C2	12	50	100	0.12	1
C3	12	50	100	0.06	1
C4	12	50	100	0.12	1
D1	12	100	100	0.01	1.5
D2	12	100	100	0.015	1.5
D3	12	100	100	0.01	1
D4	12	100	100	0.015	1
E1	18	120	60	0.024	4.3
E2	18	120	60	0.048	4.3
E3	18	120	60	0.024	8.6
E4	18	120	60	0.048	8.6
F1	18	240	60	0.016	1
F2	18	240	60	0.02	1
F3	18	240	60	0.016	1
F4	18	240	60	0.02	1
G1	18	120	120	0.012	2.15
G2	18	120	120	0.015	2.15
G3	18	120	120	0.012	4.3
G4	18	120	120	0.015	4.3
H1	18	240	120	0.012	1.5
H2	18	240	120	0.015	1.5
H3	18	240	120	0.012	1
I1	24	220	80	0.008	2.65
I2	24	220	80	0.012	2.65
I3	24	220	80	0.008	5.3
I4	24	220	80	0.012	5.3
J1	24	440	80	0.008	5.3
J2	24	440	80	0.012	5.3
J3	24	440	80	0.008	10.6
J4	24	440	80	0.012	10.6
K1	24	220	160	0.004	1
L1	24	440	160	0.002	1

Abbildung 6.1: Datensatz PAD

	Knoten	Kanten	Commocities	fixvar	ctight
c33	20	230	40	0.02	8
c34	20	230	40	0.08	8
c35	20	230	40	0.02	16
c36	20	230	40	0.08	16
c37	20	230	200	0.5	16
c38	20	230	200	1	16
c39	20	230	200	0.5	20
c40	20	230	200	1	22
c41	20	300	40	0.02	8
c42	20	300	40	0.08	10
c43	20	300	40	0.02	16
c44	20	300	40	0.08	16
c45	20	300	200	0.5	25
c46	20	300	200	1	25
c47	20	300	200	0.5	28
c48	20	300	200	1	28
c49	30	520	100	0.1	20
c50	30	520	100	0.5	20
c51	30	520	100	0.1	30
c52	30	520	100	0.5	30
c53	30	520	400	0.2	40
c54	30	520	400	0.4	40
c55	30	520	400	0.2	50
c56	30	520	400	0.4	50
c57	30	700	100	0.1	20
c58	30	700	100	0.2	20
c59	30	700	100	0.1	30
c60	30	700	100	0.2	30
c61	30	700	400	0.2	40
c62	30	700	400	0.4	40
c63	30	700	400	0.2	50
c64	30	700	400	0.4	50

Abbildung 6.2: Datensatz CANAD-C

	Knoten	Kanten	Commodities
r01	10	35	10
r02	10	35	25
r03	10	35	50
r04	10	60	10
r05	10	60	25
r06	10	60	50
r07	10	82	10
r08	10	83	25
r09	10	83	50
r10	20	120	40
r11	20	120	100
r12	20	120	200
r13	20	220	40
r14	20	220	100
r15	20	220	200
r16	20	314	40
r17	20	318	100
r18	20	315	200

	fixvar	ctight
1	0.01	1
2	0.05	1
3	0.1	1
4	0.01	2
5	0.05	2
6	0.1	2
7	0.01	8
8	0.05	8
9	0.1	8

Abbildung 6.3: Datensatz CANAD-R

	Knoten	Kanten	Commodities
s1	40	500	100
s2	40	500	250
s3	40	500	500
s4	40	1000	100
s5	40	1000	250
s6	40	1000	500
s7	40	1250	100
s8	40	1250	250
s9	40	1250	500
s10	60	1000	100
s11	60	1000	250
s12	60	1000	500
s13	60	2000	100
s14	60	2000	250
s15	60	2000	500
s16	60	2500	100
s17	60	2500	250
s18	60	2500	500

	fixvar	ctight
1	0.01	1
2	0.05	2
3	0.1	8

Abbildung 6.4: Datensatz CANAD-S

	CPLEX 9.0	CPLEX 9.0	NDBB	NDBB	NDBC	NDBC
	time (sec)	nodes	time (sec)	nodes	time (sec)	nodes
A1	0.01	1	0.43	1	0.21	1
A2	0.26	2	0.86	5	0.49	5
A3	0.06	1	0.62	3	0.39	3
A4	0.32	2	1.1	15	0.75	7
B1	1.82	3	1.39	11	1.38	25
B2	13.71	16	4.82	75	2.34	41
B3	7.59	11	2.52	39	1.75	21
B4	3.23	1	1.33	15	0.89	9
C1	3.56	2	2.77	19	2.14	17
C2	37.61	93	100.2	1531	13.76	375
C3	5.35	18	7.27	77	2.98	33
C4	41.09	199	81.48	1205	16.73	483
D1	5.74	5	4.98	41	3.04	23
D2	20.87	24	17.1	141	5.91	41
D3	8.65	7	5.32	29	4.56	36
D4	29.14	114	94.97	837	35.36	563
E1	28.83	52	46.57	575	13.06	271
E2	67.31	122	245.48	4102	18.24	480
E3	19.62	62	35.53	547	10.31	275
E4	205.06	309	247.66	3738	43.01	1190
F1	16.76	1	9.25	43	4.64	33
F2	218.64	360	468.22	4116	83.79	1149
F3	77.72	109	68.29	535	28.92	365
F4	41.08	4	6.3	41	4.11	39
G1	28.35	17	26.35	117	8.52	63
G2	74.99	115	179.04	1009	38.4	455
G3	25.22	14	18.52	91	7.36	55
G4	113.09	166	545.89	3685	135.51	1701
H1	3.43	2	8.08	13	8	15
H2	126.24	26	83.36	299	17.68	87
H3	2.91	1	7.51	13	6.95	17
I1	1.01	1	5.09	15	3.67	7
I2	59.93	37	46.96	245	15.35	119
I3	60.71	57	110.02	557	47.57	413
I4	93.98	61	385.19	2071	31.93	343
J1	36.34	2	19.44	57	8.53	54
J2	360.1	70	397.02	1463	80.62	640
J3	110.9	49	324.84	1106	168.47	1609
J4	1018.8	286	2374.6	8979	428.53	3459
K1	116.77	15	72.53	163	42.01	149
L1	597.6	126	3270.9	4575	593.48	1611
Summe	3684.4	2563	9329.8	42199	1941.34	16282

Abbildung 6.5: Experiment 1.1: exaktes Lösen der PAD Benchmark (siehe Seite 123).

			CPLEX 9.0	CPLEX 9.0	CPLEX 9.0	NDBC	NDBC	NDBC
PROB	BEST LOWER	BEST UPPER	solution	sec	nodes	solution	sec	nodes
r10.1	200087.00	200087.00	200087	0.60	71	200087	1.87	108
r10.2	346812.10	346813.50	346813.5	80.18	1831	346813.5	66.21	2769
r10.3	488006.59	488015.00	488015	123.92	1160	488015	30.98	1579
r10.4	229195.91	229196.00	229196	22.63	2063	229196	264	11305
r10.5	411663.66	411664.00	411664	838.20	13345	411664	1255.8	46890
r10.6	609103.86	609104.00	609104	4474.02	29200	609104	2791.6	105453
r10.7	486847.34	486895.00	486895	28.77	1344	487530	6407.53	49
r10.8	950977.81	951056.00	951056	41.75	1267	963728	2.65	67
r10.9	1421607.00	1421746.00	1421746	43.16	2263	1489233	1.39	73
r11.1	714429.45	714431.00	714431	37.50	227	714431	61.02	1073
r11.2	1263712.80	1263713.00	1263713	7223.34	7817	1263713	6393.9	107312
r11.3	1830829.39	1846295.00	1846295	14400.00	8637	1866891	9.5	111
r11.4	870450.88	870451.00	870451	728.33	2937	870451	1441.5	24439
r11.5	1623639.98	1623640.00	1623640	3553.45	3943	1623640	6598.5	103844
r11.6	2413822.22	2414060.00	2414060	9167.19	7932	2433587	2.14	121
r11.7	2294912.00	2294912.00	2294912	6.07	24	2294912	3036.6	45207
r11.8	3506968.95	3507100.00	3507100	13.36	79	3507100	248.14	200
r11.9	4579353.00	4579353.00	4579353	10.42	38	4579353	1245.9	20259
r12.1	1639442.64	1639443.00	1639443	779.32	788	1639443	465.38	3608
r12.2	3368442.73	3403901.00	3403901	14400.00	2754	3426893	22.77	169
r12.3	5181106.99	5279872.00	5279872	14400.00	2155	5363192.222	25.22	121
r12.4	2303556.23	2303557.00	2303557	243.35	183	2303557	4257	34931
r12.5	4669795.17	4669799.00	4669799	411.28	237	4669799	1154.4	9434
r12.6	7100016.36	7100019.00	7100019	413.25	225	7100019	1372.4	11771
r12.7	7635270.00	7635270.00	7635270	7.76	6	7635270	61.53	378
r12.8	10067732.88	10067742.00	10067742	6.03	3	10067742	150.06	1190
r12.9	11967768.00	11967768.00	11967768	5.52	0	11967768	16.31	78
r13.1	142947.00	142947.00	142947	0.59	13	142947	1.01	47
r13.2	263799.18	263800.00	263800	133.31	408	263800	61.33	1736
r13.3	365833.65	365836.00	365836	277.36	432	365836	37.19	1361
r13.4	150976.48	150977.00	150977	7.69	461	150977	115.25	4818
r13.5	282681.84	282682.00	282682	1677.66	7254	282682	381.2	11637
r13.6	406789.92	406790.00	406790	10414.67	16900	406790	1120.2	28930
r13.7	208067.23	208088.00	208088	11680.30	203324	209684	10.23	200
r13.8	436264.36	446764.00	446764	14400.00	53035	458377	4.78	194
r13.9	681259.03	698000.00	698000	14400.00	37738	732623	2.05	141
r14.1	403413.36	403414.00	403414	42.66	197	403414	72.56	1367
r14.2	749430.44	749503.00	749503	12269.46	4541	760447	6.84	200
r14.3	1057434.70	1072504.00	1072504	14400.00	2327	1096342	13.16	200
r14.4	437606.74	437607.00	437607	110.18	450	437607	670.77	8299
r14.5	844870.07	851944.00	853062	14400.00	4367	864870	14.61	200
r14.6	1208160.84	1215904.00	1223074	14400.00	1942	1223899	11.08	200
r14.7	662281.33	669848.54	669848.5424	14400.00	17752	676290.9697	4.02	200
r14.8	1577351.14	1624540.33	1624540.333	14400.00	6686	1643254	17.36	139
r14.9	2572429.42	2623288.00	2623288	14400.00	6950	2737598	15.9	123
r15.1	1000786.84	1000787.00	1000787	654.45	420	1000787	590.08	3770
r15.2	1952335.91	1971622.50	1991646	14400.00	776	1981107	21.17	200
r15.3	2829297.13	2887544.00	2926793	14400.00	225	2940386	38.84	200
r15.4	1147196.34	1148604.00	1148604	14400.00	4240	1150218.5	43.66	200
r15.5	2445116.07	2488910.00	2539170	14400.00	601	2539396	56.15	200
r15.6	3751570.63	3846435.00	3940899	14400.00	375	3909492	40.56	200
r15.7	2293888.23	2297919.00	2297919	14400.00	3823	2309079.125	32.64	167
r15.8	5573316.24	5573412.83	5573412.833	2692.63	1006	5600443	17.55	200
r15.9	8696930.81	8696932.00	8696932	659.08	36	8696932	3596.9	22253
r16.1	136161.00	136161.00	136161	0.45	0	148837	0.19	0
r16.2	239500.00	239500.00	239500	444.96	443	239500	192.58	5425
r16.3	325670.42	325671.00	325671	683.57	561	325671	122.78	3879
r16.4	138531.92	138532.00	138532	3.73	40	138532	5.37	134
r16.5	241796.20	241801.00	241801	201.24	254	241801	21.03	628
r16.6	337761.91	337762.00	337762	2836.31	2903	337762	195.81	5544
r16.7	167773.86	169284.00	169284	14400.00	180141	173796	2.62	200
r16.8	338667.39	350229.00	350229	14400.00	20948	362634	11.58	200
r16.9	510101.49	537419.00	547030	14400.00	13796	579435	8.53	187
r17.1	354119.57	354138.00	354138	49.72	42	354138	4.33	59
r17.2	645488.00	645488.00	647696	14400.00	2318	645488	1970.7	20063
r17.3	896987.36	915823.00	931218	14400.00	1097	936938	19.38	200
r17.4	370589.98	370590.00	370590	552.39	1038	370590	718.18	8000
r17.5	704997.65	706876.00	713637	14400.00	2225	715361	10.05	200
r17.6	1001639.15	1021162.00	1058112	14400.00	1027	1048152	9.46	200
r17.7	496518.01	502469.50	502469.5	14400.00	8485	507385	23.89	200
r17.8	1082067.58	1108263.00	1121588	14400.00	2357	1158692	25.46	200
r17.9	1727562.81	1794487.78	1794487.783	14400.00	1489	1866253	25.26	200
r18.1	827673.34	828117.00	828117	14400.00	2563	831436	49.15	200
r18.2	1533674.72	1533675.00	1544539	14400.00	159	1533675	1345.4	6663
r18.3	2152469.22	2214574	0	14400.00	69	2214574	18.65	200
r18.4	913631.22	921163.00	921216	14400.00	2223	925983	41.28	200
r18.5	1795617.87	1871773	0	14400.00	116	1871773	26.74	200
r18.6	2583876.72	2765264	0	14400.00	59	2765264	37.05	200
r18.7	1460582.61700	1484412.03	1487354.524	14400.00	1073	1488726.178	69.44	200
r18.8	3782429.63400	3931727.80	3935970	14400.00	533	4031701	65.91	191
r18.9	6177754.14100	6435842.00	6435842	14400.00	620	6733707	60.31	177

Abbildung 6.6: Experiment 1.2: CANAD-R2 (siehe Seite 125).

			TABU-PATH		TABU-ARC		SS/PL/ID		TABU-CYCLE		PATH-RELINKING		CPLEX 9.0	CPLEX 9.0	NDBC	NDBC
			(m2)		(400) (m2)		(400) (m1)		(m1)		(m1)				300 sec	300 sec
	best lower bound	best solution	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ
r10.1	200087.00	200087.00	201350	0.63	200613	0.26	201744	0.83	200484	0.20	200147	0.03	200087	0.0000	202094	1.0031
r10.2	346812.10	346813.50	362848	4.62	350573	1.08	368195	6.17	350407	1.04	347459	0.19	346813.5	0.0004	380501	9.7139
r10.3	488006.59	488015.00	568114	16.42	507118	3.92	560787	14.91	502724	3.02	502724	3.02	488015	0.0017	626707	28.4218
r10.4	229195.91	229196.00	231546	1.03	232473	1.43	232482	1.43	231302	0.92	229196	0.00	229196	0.0000	229388	0.0838
r10.5	411663.66	411664.00	439802	6.84	432913	5.16	440435	6.99	430254	4.52	425230	3.30	411664	0.4927	411794	0.0317
r10.6	609103.86	609104.00	659189	8.22	640621	5.17	673263	10.53	646107	6.08	628353	3.16	609104	2.0320	613552	0.7303
r10.7	486847.34	486895.00	489385	0.52	488737	0.39	487695	0.17	490034	0.65	487235	0.08	486895	0.0098	491563.25	0.9687
r10.8	950977.81	951056.00	993363	4.46	980010	3.05	967696	1.76	977346	2.77	963914	1.36	951056	0.0082	957432	0.6787
r10.9	1421607.00	1421746.00	1454329	2.30	1487270	4.62	1442600	1.48	1469020	3.34	1439320	1.25	1421746	0.0098	1426139	0.3188
r11.1	714429.45	714431.00	726155	1.64	725416	1.54	722956	1.19	726585	1.70	720236	0.81	714431	0.0002	714981	0.0772
r11.2	1263712.80	1263713.00	1408514	11.46	1306090	3.35	1298610	2.76	1323620	4.74	1296050	2.56	1263713	1.2123	1269835.5	0.4845
r11.3	1830829.39	1846295.00	2392241	30.66	1914040	4.54	1984400	8.39	1933560	5.61	1940180	5.97	1846295	9.9651	1850215	1.0588
r11.4	870450.88	870451.00	888165	2.04	876894	0.74	879109	0.99	879445	1.03	875908	0.63	870451	0.0619	871275	0.0947
r11.5	1623639.98	1623640.00	1846121	13.70	1694860	4.39	1706510	5.10	1715040	5.63	1674340	3.12	1623640	4.5421	1623640	0.0000
r11.6	2413822.22	2414060.00	2732989	13.22	2607690	8.03	2543000	5.35	2593590	7.45	2546730	5.51	2414060	2.3216	2420779	0.2882
r11.7	2294912.00	2294912.00	2308694	0.60	2295790	0.04	2295360	0.02	2294912	0.00	2294912	0.00	2294912	0.0000	2294912	0.0000
r11.8	3506968.95	3507100.00	3585266	2.23	3568430	1.75	3518830	0.34	3540730	0.96	3529530	0.64	3507100	0.0037	3508521	0.0443
r11.9	4579353.00	4579353.00	4901168	7.03	4621900	0.93	4617850	0.84	4616190	0.80	4592820	0.29	4579353	0.0000	4579353	0.0000
r12.1	1639442.64	1639443.00	1728210	5.41	1713670	4.53	1658640	1.17	1700170	3.70	1668370	1.76	1639443	0.0361	1639565	0.0075
r12.2	3368442.73	3403901.00	4037454	19.86	3746250	11.22	3510110	4.21	3741660	11.08	3717800	10.37	3403901	26.0648	3445991.667	2.3022
r12.3	5181106.99	5279872.00	6415405	23.82	6070200	17.16	5623070	8.53	6055670	16.88	6037960	16.54	5279872	31.8431	5325234	2.7818
r12.4	2303556.23	2303557.00	2339682	1.57	2326230	0.98	2322660	0.83	2331290	1.20	2326930	1.01	2303557	0.0000	2306253	0.1171
r12.5	4669795.17	4669799.00	4950073	6.00	4967940	6.38	4750290	1.72	4942920	5.85	4954250	6.09	4669799	4.1723	4669799	0.0001
r12.6	7100016.36	7100019.00	7781907	9.60	7638050	7.58	7427170	4.61	7562660	6.52	7638050	7.58	7100019	2.4569	7103503	0.0491
r12.7	7635270.00	7635270.00	7666421	0.41	7637250	0.03	7636930	0.02	7638050	0.04	7638050	0.04	7635270	0.0000	7635270	0.0000
r12.8	10067732.88	10067742.00	10175690	1.07	10121700	0.54	10086500	0.19	10106800	0.39	10085000	0.17	10067742	0.0001	10067742	0.0001
r12.9	11967768.00	11967768.00	12958760	8.28	12079300	0.93	11980100	0.10	12064300	0.81	12042800	0.63	11967768	0.0000	11967768	0.0000
r13.1	142947.00	142947.00	143588	0.45	144138	0.83	145007	1.44	143778	0.58	142947	0.00	142947	0.0000	142947	0.0000
r13.2	263799.18	263800.00	284943	8.02	270316	2.47	312434	18.44	272019	3.12	273853	3.81	263800	0.0003	264356	0.2111
r13.3	365833.65	365836.00	439085	20.02	374999	2.51	442663	21.00	375628	2.68	378922	3.58	365836	0.0006	369594	1.0279
r13.4	150976.48	150977.00	152873	1.26	151513	0.36	155283	2.85	151875	0.60	150977	0.00	150977	0.0003	151484	0.3362
r13.5	282681.84	282682.00	308081	8.99	291510	3.12	322230	13.99	290150	2.64	289706	2.48	282682	1.1154	285698	1.0670
r13.6	406789.92	406790.00	462407	13.67	420028	3.25	481111	18.27	430246	5.77	435724	7.11	406790	4.6700	414820	1.9740
r13.7	208067.23	208088.00	214345	3.02	212451	2.11	217623	4.59	213650	2.68	210898	1.36	208088	0.7660	209995	0.9265
r13.8	436264.36	446764.00	491210	12.59	484112	10.97	472086	8.21	485018	11.18	469035	7.51	446764	5.7836	454012	4.0681
r13.9	681259.03	698000.00	772063	13.33	758715	11.37	792368	16.31	787477	15.59	761505	11.78	698000	7.0889	707082	3.7905
r14.1	403413.36	403414.00	420837	4.32	415119	2.90	424280	5.17	416932	3.35	410430	1.74	403414	0.0002	403414	0.0002
r14.2	749430.44	749503.00	961760	28.33	803356	7.20	847412	13.07	775023	3.41	778492	3.88	749503	1.4023	761275	1.5805
r14.3	1057434.70	1072504.00	1386426	31.11	1155840	9.31	1234250	16.72	1133420	7.19	1127980	6.67	1072504	3.8649	1094623	3.5168
r14.4	437606.74	437607.00	452971	3.51	453204	3.56	470819	7.59	448725	2.54	444170	1.50	437607	0.1472	438798	0.2722
r14.5	844870.07	851944.00	1003389	18.76	912456	8.00	898722	6.37	890914	5.45	883241	4.54	853062	36.1959	869130	2.8714
r14.6	1208160.84	1215904.00	1630263	34.94	1333440	10.37	1317020	9.01	1309970	8.43	1335170	10.51	1223074	44.1559	1231155	1.9032
r14.7	662281.33	669848.54	728766	10.04	702226	6.03	685912	3.57	698381	5.45	689762	4.15	669848.5424	2.2445	672173.5	1.4937
r14.8	1577351.14	1624540.33	1873897	18.80	1748930	10.88	1694680	7.44	1782720	13.02	1740600	10.35	1624540.333	26.4699	1645693	4.3327
r14.9	2572429.42	2623288.00	3006180	16.86	2882710	12.06	2824250	9.79	2901140	12.78	2939990	14.29	2623288	27.2122	2668006.667	3.7154
r15.1	1000786.84	1000787.00	1087543	8.67	1049360	4.85	1050460	4.96	1044180	4.34	1027930	2.71	1000787	0.0666	1000861	0.0074
r15.2	1952335.91	1971622.50	2912747	49.19	2158720	10.57	2194510	12.40	2164870	10.89	2135640	9.39	1991646	100.0000	2051772	5.0932
r15.3	2829297.13	2887544.00	4729421	67.16	3135760	10.83	3196590	12.98	3156520	11.57	3131640	10.69	2926793	100.0000	2951269	4.3110
r15.4	1147196.34	1148604.00	1253329	9.25	1215130	5.92	1201200	4.71	1211560	5.61	1184780	3.28	1148604	0.2008	1151305	0.3581
r15.5	2445116.07	2488910.00	3348931	36.96	2756680	12.74	2690650	10.04	2736570	11.92	2732310	11.75	2539170	100.0000	2548908	4.2449
r15.6	3751570.63	3846435.00	5505598	46.75	4384640	16.87	4080930	8.78	4393650	17.11	4264090	13.66	3940899	100.0000	3908093.333	4.1722
r15.7	2293888.23	2297919.00	2454883	7.02	2355730	2.70	2329480	1.55	2355980	2.71	2350730	2.48	2297919	1.0822	2305383.667	0.5011
r15.8	5573316.24	5573412.83	5839188	4.77	5926330	6.33	5761700	3.38	5941080	6.60	5830430	4.61	5573412.833	6.4216	5597469	0.4334
r15.9	8696930.81	8696932.00	9547358	9.78	9180920	5.57	8869500	1.98	9140060	5.10	9123020	4.90	8696932	100.0000	8700994	0.0467
r16.1	136161.00	136161.00	139134	2.18	136538	0.28	143938	5.71	136851	0.51	136161	0.00	136161	0.0000	136161	0.0000
r16.2	239500.00	239500.00	261086	9.01	247682	3.42	265913	11.03	245003	2.30	244070	1.91	239500	0.3933	240242	0.3098
r16.3	325670.42	325671.00	372377	14.34	338807	4.03	372088	14.25	346724	6.46	330138	1.37	325671	0.0002	329341	1.1271
r16.4	138531.92	138532.00	139301	0.56	139973	1.04	147230	6.28	140433	1.37	139131	0.43	138532	0.0001	138532	0.0001
r16.5	241796.20	241801.00	257779	6.61	246014	1.74	275935	14.12	250332	3.53	248717	2.86	241801	0.0020	243740	0.8039
r16.6	337761.91	337762.00	371043	9.85	355610	5.28	384183	13.74	352450	4.35	343165	1.60	337762	3.4797	341524	1.1138
r16.7	167773.86	169284.00	173354	3.33	172268	2.68	180511	7.59	173164	3.21	171651	2.31	169284	2.1089	170755	1.7769
r16.8	338667.39	350229.00	381866	12.76	365214	7.84	384932	13.66	374372	10.54	371129	9.59	350229	11.7276	352877	4.1957
r16.9	510101.49	537419.00	570012	11.74	569874	11.72	585508	14.78	587242	15.12	564151	10.60	547030	15.5696	543137	6.4763
r17.1	354119.57	354138.00	383719	8.36	370090	4.51	388664	9.76	367923	3.90	364325	2.88	354138	0.0052	354138	0.0052
r17.2	645488.00	645488.00	816660	26.52	688554	6.67	776748	20.34	693482	7.44	669935	3.79	647696	3.8733	669207	3.6746
r17.3	896987.36	915823.00	1323708	47.57	971151	8.27	1056580	17.79	990967	10.48	973486	8.53	931218	94.7099	974998	8.6970
r17.4	370589.98	370590.00	395645	6.76	380850	2.77	401140	8.24	382868	3.31	378905	2.24	370590	0.0086	371237	0.1746
r17.5	704997.65	706876.00	865932	22.83	753188	6.84	831923	18.00	761476	8.01	743173	5.41	713637	1.5910	710212	0.7396
r17.6	1001639.15	1021162.00	1370038	36.78	1108180	10.64	1141400	13.95	1104380	10.26	1084280	8.25	1058112	58.7273	1054871	5.3145
r17.7	496518.01	502469.50	534490	7.65	524038	5.54	528399	6.42	530313	6.81	521468	5.02	502469.5	2.4595	508408	2.3947
r17.8	1082067.58	1108263.00	1332873	23.18	1195140	10.45	1213600	12.16	1249440	15.47	1191580	10.12	1121588	25.9075	1112935.167	2.8526
r17.9	1727562.81	1794487.78	2246433	30.03	1945080	12.59	1980780	14.66	2017720	16.80	1985220	14.91	1794487.783	30.7980	1835051	6.2220
r18.1	827673.34	828117.00	923006	11.52	872888	5.46	908872	9.81	866838	4.73	858043	3.67	828117	0.9472	844129	1.9882
r18.2	1533674.72	1533675.00	2247779	46.56	1716680	11.93	1769740	15.39	1671810	9.01	1649730	7.57	1544539	100.0000	1540297	0.4318
r18.3	2152469.22	2214574	3542160	64.56	2377560	10.46	2458240	14.21	2383760	10.75	2418190	12.34	0	100.0000	2280780	5.9611
r18.4	913631.22	921163.00	1017002	11.31	975396	6.76	968334	5.99	980992	7.37	964367	5.55	921216	15.3125	947171	3.6710
r18.5	1795617.87	1871773	2225384	23.93	2037950	13.50	2014130	12.17	2033500	13.25	1980170	10.28	0	100.0000	1921457.5	7.0082
r18.6	2583876.72	2765264	4054910	56.93	2966370	14.80	3070830	18.85	2981980	15.41	2928930	13.35	0	100.0000	2854452.5	10.4717
r18.7	1460582.61700	1484412.03	1633508	11.84	1622520	11.09	1559600	6.78	1611800	10.35	1568050	7.36	1487354.524	100.0000	1497683.25	2.5401
r18.8	3782429.63400	3931727.80	4454958	17.78	4576750	21.00	4085010	8.00	4379870	15.80	4553560	20.39	3935970	100.0000	4024891.8	6.4102
r18.9	6177754.14100	6435842.00	7125415	15.34	7504310	21.47	7014430	13.54	7562480	22.41	7639160	23.66	6435842	100.0000	6602176.333	6.8702

Abbildung 6.7: Experiment 1.3: CANAD-R2 (siehe Seite 127).

		TABU-PATH		TABU-ARC		SS/PL/ID		TABU-CYCLE		PATH-RELINKING		CPLEX 9.0		CPLEX 9.0		NDBC		NDBC	
		(m2)		(400) (m2)		(400) (m1)		(m1)		(m1)		600 sec		14400 sec		600 sec		1800 sec	
	best lower bound	best solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ
c33	423848	423848	0.28	425046	0.22	426020	0.51	425091	0.29	424385	0.13	423848	0.00	423848	0.00	423848	0.00	423848	0.00
c35	371467.0847	371475	0.09	371816	0.09	371893	0.11	371642	0.05	371893	0.11	371475	0.00	371475	0.00	372470	0.27	371475	0.00
c36	643036	643036	0.18	644172	0.18	645812	0.43	651100	1.25	643774	0.11	644464	0.22	643036	0.00	649352	0.98	647370	0.67
c37	91333.66015	94218	34.22	122592	34.22	98995	8.39	106709	16.83	100620	10.17	100205	9.71	96085	5.20	95191	4.22	94300	3.25
c38	132315.7426	137854	42.53	188590	42.53	146535	10.75	152957	15.60	149715	13.15	148044	11.89	190091	43.66	139732	5.60	138107	4.38
c39	95675.94029	97914	23.39	118057	23.39	104752	9.49	109622	14.58	103945	8.64	104128	8.83	98991	3.46	98159	2.60	97914	2.34
c40	131545.752	136067	38.99	182829	38.99	147385	12.04	152880	16.22	147385	12.04	144857	10.12	196276	49.21	138451	5.25	136567	3.82
c41	429398	429398	0.12	429912	0.12	429535	0.03	430373	0.23	429535	0.03	429398	0.00	429398	0.00	429398	0.00	429398	0.00
c42	585998.1248	586077	0.54	589190	0.54	593322	1.25	600648	2.50	592043	1.03	593304	1.25	586077	0.01	586077	0.01	590930	0.84
c43	464509	464509	0.00	464509	0.00	464724	0.05	464818	0.07	464767	0.06	464509	0.00	464509	0.00	464509	0.00	465336	0.18
c44	604198	604198	0.00	606364	0.36	607100	0.48	611481	1.21	609511	0.88	604828	0.10	604198	0.00	604198	0.00	604821	0.10
c45	73130.90357	74902	20.88	88398	20.88	80819	10.51	82666	13.04	80266	9.76	79128.5	8.20	92005	25.81	75122	2.72	74970.66667	2.52
c46	110929.4482	115754	36.41	151317	36.41	123347	11.19	134532	21.28	122072	10.04	123179	11.04	-	100.00	120395.25	8.53	116006	4.58
c47	74025.16598	74991	11.75	82724	11.75	79619	7.56	82301	11.18	79600	7.53	77839.5	5.15	75112	1.47	74991	1.30	75588	2.11
c48	103714.0588	107102	30.74	135593	30.74	114484	10.38	118712	14.46	116106	11.95	111590	7.59	130037	25.38	109767	5.84	107521.5	3.67
c49	53714.15788	53964	5.05	56426	5.05	54958	2.32	59015	9.87	54877	2.16	54282	1.06	54012	0.55	53958	0.45	54007	0.55
c50	90207.66494	94119	15.42	104117	15.42	99586	10.40	107956	19.67	100646	11.57	102016	13.09	126366	40.08	95061	5.38	94588	4.86
c51	51768.33327	52070	2.94	53288	2.94	52985	2.35	53065	2.50	52786	1.97	52510	1.43	52442	1.30	52103	0.65	52170	0.78
c52	94216.68396	97881	14.52	107894	14.52	105523	12.00	107638	14.25	104346	10.75	104188	10.58	124039	31.65	99126	5.21	98481	4.53
c53	111763.0914	112830	12.59	125831	12.59	120652	7.95	118976	6.45	120652	7.95	118873	6.36	-	100.00	113219.6548	1.30	112986.7308	1.09
c54	146693.242	149798	20.94	177409	20.94	161098	9.82	163897	11.73	161098	9.82	159007	8.39	-	100.00	-	100.00	149696.3793	2.05
c55	114083.2571	114680	10.02	125518	10.02	121588	6.58	119414	4.67	121588	6.58	119744	4.96	-	100.00	114786.2788	0.62	114841.1538	0.66
c56	149772.3093	152690	16.53	174526	16.53	167939	12.13	166745	11.33	167939	12.13	163352	9.07	-	100.00	-	100.00	153422.2222	2.44
c57	47602.9153	47603	2.90	48984	2.90	48398	1.67	50227	5.51	48553	2.00	47924	0.67	47603	0.00	47603	0.00	47612	0.02
c58	58229.46348	60195	12.24	65356	12.24	62471	7.28	68097	16.95	62611	7.52	62349	7.07	60978	4.72	60419	3.76	60464	3.84
c59	45222.45428	45880	4.11	47083	3.99	47025	3.99	48300	6.81	46685	3.23	46565	2.97	46108	1.96	46051	1.83	51043	12.87
c60	53809.67879	54974	9.28	58804	9.28	57886	7.58	61689	14.64	56534	5.06	56246	4.53	55627	3.38	55018	2.25	55156	2.50
c61	96605.02529	98030.1	13.87	110000	13.87	106777	10.53	104929	8.62	104860	8.55	103138	6.76	-	100.00	-	100.00	98355	1.81
c62	130723.5808	135403	26.59	165484	26.59	148950	13.94	155936	19.29	149583	14.43	144615	10.63	-	100.00	-	100.00	135877.3226	3.94
c63	94012.32584	95536.9	10.38	103768	10.38	101672	8.15	101116	7.56	101623	8.10	99606	5.95	-	100.00	-	100.00	96139.44444	2.26
c64	127572.1129	130689.8462	18.30	150919	18.30	142778	11.92	144487	13.26	143324	12.35	138777	8.78	-	100.00	-	100.00	131574.5	3.14
			Ø SQ		14.07		Ø SQ		9.75		Ø SQ		6.77		Ø SQ		5.70		Ø SQ

Abbildung 6.8: Experiment 1.4: CANAD-C (siehe Seite 129).

			NDBC-Alpha		NDBC-Beta		NDBB-Alpha		NDBB-Beta	
	best lower bound	best solution	solution	SQ	solution	SQ	solution	SQ	solution	SQ
c33	423848	423848	423848	0.00	424241	0.09	425200	0.32	424890	0.25
c35	371467.0847	371475	372470	0.27	372197	0.20	371961	0.13	371995	0.14
c36	643036	643036	649352	0.98	647829	0.75	645259	0.35	647419	0.68
c37	91333.66015	94218	94300	3.25	94828.33333	3.83	94530	3.50	96031	5.14
c38	132315.7426	137854	138107	4.38	140352	6.07	139723.5	5.60	141909.5	7.25
c39	95675.94029	97914	97914	2.34	97914	2.34	98370	2.82	99751	4.26
c40	131545.752	136067	136567	3.82	137145	4.26	138505.5	5.29	138419	5.22
c41	429398	429398	429398	0.00	429398	0.00	429398	0.00	429398	0.00
c42	585998.1248	586077	590930	0.84	591056	0.86	589508	0.60	593742	1.32
c43	464509	464509	465336	0.18	465008.5	0.11	464509	0.00	466105.5	0.34
c44	604198	604198	604821	0.10	604208	0.00	604208	0.00	606657	0.41
c45	73130.90357	74902	74970.66667	2.52	75812.5	3.67	75263	2.92	76589	4.73
c46	110929.4482	115754	116006	4.58	117511	5.93	117845.6	6.23	119464.5	7.69
c47	74025.16598	74991	75588	2.11	74995	1.31	75102.66667	1.46	75616	2.15
c48	103714.0588	107102	107521.5	3.67	109081	5.17	107745	3.89	110046	6.11
c49	53714.15788	53964	54007	0.55	54246	0.99	54117	0.75	55100	2.58
c50	90207.66494	94119	94588	4.86	95926	6.34	95194	5.53	101232	12.22
c51	51768.33327	52070	52170	0.78	52462	1.34	52354.5	1.13	53004	2.39
c52	94216.68396	97881	98481	4.53	100068	6.21	99443	5.55	103035	9.36
c53	111763.0914	112830	113062.5	1.16	113263.859	1.34	114011.2857	2.01	113264.4838	1.34
c54	146693.242	149798	150063	2.30	150973.5	2.92	151595.3333	3.34	153703.2545	4.78
c55	114083.2571	114680	114841.1538	0.66	114875.05	0.69	114864.5758	0.68	114886.7379	0.70
c56	149772.3093	152690	153466.8511	2.47	153426	2.44	156249.125	4.32	153821.2222	2.70
c57	47602.9153	47603	47612	0.02	47671	0.14	47652	0.10	47671	0.14
c58	58229.46348	60195	60464	3.84	61893	6.29	60785	4.39	65886	13.15
c59	45222.45428	45880	51043	12.87	46587	3.02	51043	12.87	47312	4.62
c60	53809.67879	54974	55156	2.50	56194	4.43	55426	3.00	56909	5.76
c61	96605.02529	98030.1	98796.4	2.27	98122	1.57	100276.5	3.80	99145.5	2.63
c62	130723.5808	135403	136108.7938	4.12	136738.25	4.60	142553.5714	9.05	147490	12.83
c63	94012.32584	95536.9	96370.46154	2.51	96587.07143	2.74	97615	3.83	96977.67442	3.15
c64	127572.1129	130689.8462	131933.1667	3.42	132575	3.92	134610	5.52	132490.0175	3.85
			Ø SQ	2.51	Ø SQ	2.70	Ø SQ	3.19	Ø SQ	4.13

Abbildung 6.9: Experiment 1.5: CANAD-C (siehe Seite 130).

			NDBC 70h		NDBC 24h		NDBC 7200 sec		NDBC 1800 sec		NDBC 300 sec	
	best lower bound	best solution	solution	SQ	solution	SQ	solution	SQ	solution	SQ	solution	SQ
s01.1	515581.94	515582.00	515582	0.00	515582	0.00	522700	1.38	528970	2.60	533332	3.44
s01.2	917096.41	975501.00	975501	6.37	990957	8.05	1011982	10.35	1040569	13.46	1052798	14.80
s01.3	1279836.60	1394648.00	1394648	8.97	1416852	10.71	1453459	13.57	1456472	13.80	1480561	15.68
s02.1	1420812.74	1434958.00	1434958	1.00	1434958	1.00	1475633	3.86	1493623	5.12	1507758	6.12
s02.2	2615306.69	2844352.00	2844352	8.76	2844352	8.76	3080439	17.79	3116558	19.17	3126235	19.54
s02.3	3666073.77	4133726.00	4133726	12.76	4181785	14.07	4269129	16.45	4322416	17.90	4352958	18.74
s03.1	3036404.42	3122679.00	3122679	2.84	3122679	2.84	3230086	6.38	3270529	7.71	3297768	8.61
s03.2	5330266.37	6036406.00	6036406	13.25	6036406	13.25	6686365	25.44	6729390	26.25	9564349	79.43
s03.3	7411882.32	8989067.00	8989067	21.28	8989067	21.28	9629184	29.92	9899454	33.56	13604349	83.55
s04.1	383177.32	388186.00	388186	1.31	388186	1.31	388186	1.31	391004	2.04	393186	2.61
s04.2	720998.90	840705.00	840705	16.60	840705	16.60	840705	16.60	850102	17.91	856709	18.82
s04.3	1023021.85	1308408.00	1308408	27.90	1308408	27.90	1308408	27.90	1308408	27.90	1318040	28.84
s05.1	1076860.32	1140134.00	1140134	5.88	1140134	5.88	1167704	8.44	1171330	8.77	1185830	10.12
s05.2	1977564.49	2483342.00	2483342	25.58	2483342	25.58	2483342	25.58	2486019	25.71	2620220	32.50
s05.3	2769734.86	3433615.00	3433615	23.97	3433615	23.97	3433615	23.97	3454679	24.73	3860701	39.39
s06.1	2345824.39	2503826.00	2503826	6.74	2503826	6.74	2787552	18.83	2787552	18.83	3404093	45.11
s06.2	4172669.63	5248396.00	5248396	25.78	5248396	25.78	5248396	25.78	5248396	25.78	9330719	123.62
s06.3	5817707.80	7669945.00	7669945	31.84	7669945	31.84	7733974	32.94	7766806	33.50	17236472	196.28
s07.1	364960.98	383975.00	383975	5.21	383975	5.21	383975	5.21	386260	5.84	387588	6.20
s07.2	694920.66	806134.00	806134	16.00	806134	16.00	806134	16.00	822288	18.33	827751	19.11
s07.3	986634.44	1214597.00	1214597	23.11	1214597	23.11	1214597	23.11	1233450	25.02	1262219	27.93
s08.1	1015003.36	1139985.00	1139985	12.31	1139985	12.31	1139985	12.31	1139985	12.31	1187755	17.02
s08.2	1869051.47	2359041.00	2359041	26.22	2359041	26.22	2359041	26.22	2359041	26.22	2666709	42.68
s08.3	2607713.17	3383331.00	3383331	29.74	3383331	29.74	3383331	29.74	3400667	30.41	3583665	37.43
s09.1	2151850.60	2418872.00	2418872	12.41	2418872	12.41	2519507	17.09	2556609	18.81	3222099	49.74
s09.2	3821142.68	4782815.00	4782815	25.17	4782815	25.17	4782815	25.17	4797239	25.54	10708110	180.23
s09.3	5309683.93	6564164.00	6564164	23.63	6564164	23.63	6564164	23.63	6789202	27.86	15621222	194.20
s10.1	527003.42	536431.00	536431	1.79	536431	1.79	543118	3.06	545082	3.43	545138	3.44
s10.2	1009636.33	1143975.00	1143975	13.31	1143975	13.31	1733344	71.68	1733344	71.68	1733344	71.68
s10.3	1436495.85	1792603.00	1792603	24.79	1792603	24.79	4351725	202.94	4351725	202.94	4351725	202.94
s11.1	1485993.28	1636475.00	1636475	10.13	1636475	10.13	1741951	17.22	1745435	17.46	1842814	24.01
s11.2	2802131.87	3674166.00	3674166	31.12	3674166	31.12	3674166	31.12	3743322	33.59	4742866	69.26
s11.3	3978919.22	5178881.00	5178881	30.16	5178881	30.16	5178881	30.16	5251655	31.99	6515765	63.76
s12.1	3022774.72	3299068.00	3299068	9.14	3307161	9.41	3546826	17.34	3554882	17.60	4397424	45.48
s12.2	5633799.36	7020229.00	7020229	24.61	7020229	24.61	7196256	27.73	7270757	29.06	10935427	94.10
s12.3	8026995.20	10426534.00	10426534	29.89	10426534	29.89	10689099	33.16	10689099	33.16	21294873	165.29
s13.1	417013.46	430187.00	430187	3.16	430187	3.16	430187	3.16	430588	3.26	432886	3.81
s13.2	807030.00	967448.00	967448	19.88	967448	19.88	967448	19.88	967448	19.88	1050417	30.16
s13.3	1143843.17	1858254.00	1858254	62.46	1923337	68.15	1967471	72.01	1967471	72.01	1967471	72.01
s14.1	1189728.11	1415378.00	1415378	18.97	1415378	18.97	1415378	18.97	1423513	19.65	1777615	49.41
s14.2	2246472.17	2861844.00	2861844	27.39	2861844	27.39	2861844	27.39	2903666	29.25	4144993	84.51
s14.3	3197920.96	4381562.00	4381562	37.01	4381562	37.01	4381562	37.01	4396445	37.48	6589810	106.07
s15.1	1867246.76	2057513.00	2057513	10.19	2060893	10.37	2090923	11.98	2133342	14.25	2577232	38.02
s15.2	2995702.86	4049877.00	4049877	35.19	4049877	35.19	4154875	38.69	4321261	44.25	7477998	149.62
s15.3	3938047.69	5465210.00	5465210	38.78	5465210	38.78	5465210	38.78	5587692	41.89	10963547	178.40
s16.1	381339.18	401783.00	401783	5.36	401783	5.36	401783	5.36	412588	8.19	412683	8.22
s16.2	739755.17	904185.00	904185	22.23	904185	22.23	904185	22.23	909578	22.96	944337	27.66
s16.3	1059838.81	1320124.00	1320124	24.56	1320124	24.56	1320124	24.56	1329522	25.45	1610407	51.95
s17.1	1083953.20	1269909.00	1269909	17.16	1269909	17.16	1269909	17.16	1275353	17.66	1635627	50.89
s17.2	2083872.03	2725406.00	2725406	30.79	2725406	30.79	2725406	30.79	2733432	31.17	4001454	92.02
s17.3	2980622.02	4126209.00	4126209	38.43	4126209	38.43	4126209	38.43	4139265	38.87	6398735	114.68
s18.1	1305864.97	1320195.00	1320195	1.10	1320195	1.10	1320195	1.10	1321022	1.16	1343682	2.90
s18.2	1861959.52	2193615.00	2193615	17.81	2193615	17.81	2193615	17.81	2273356	22.09	3178667	70.72
s18.3	2267621.54	2828850.00	2828850	24.75	2828850	24.75	2828850	24.75	3008523	32.67	5042919	122.39

Abbildung 6.10: Experiment 1.6: PAD-S (siehe Seite 132).

			NDBC-1800sec		NDBC-1800sec	
	best lower bound	best solution	nur 2. Phase	SQ	1.+2. Phase	SQ
c33	423848	423848	423848	0.00	423848	0.00
c35	371467.0847	371475	371475	0.00	371475	0.00
c36	643036	643036	647370	0.67	647829	0.75
c37	91333.66015	94218	101066	10.66	94853.5	3.85
c38	132315.7426	137854	154299	16.61	139053	5.09
c39	95675.94029	97914	106343	11.15	98833	3.30
c40	131545.752	136067	164443	25.01	138133	5.01
c41	429398	429398	429398	0.00	429398	0.00
c42	585998.1248	586077	586077	0.01	586077	0.01
c43	464509	464509	464509	0.00	464509	0.00
c44	604198	604198	604198	0.00	604198	0.00
c45	73130.90357	74902	83659	14.40	75362	3.05
c46	110929.4482	115754	138016	24.42	116819.6667	5.31
c47	74025.16598	74991	86903	17.40	75382.66667	1.83
c48	103714.0588	107102	141246.5	36.19	108835	4.94
c49	53714.15788	53964	54845	2.11	54355	1.19
c50	90207.66494	94119	108213	19.96	96339	6.80
c51	51768.33327	52070	55360	6.94	52565	1.54
c52	94216.68396	97881	106730	13.28	101317	7.54
c53	111763.0914	112830	142616	27.61	113294	1.37
c54	146693.242	149798	167489	14.18	151057.2	2.97
c55	114083.2571	114680	127571	11.82	114803	0.63
c56	149772.3093	152690	192577.5	28.58	153336	2.38
c57	47602.9153	47603	47603	0.00	47603	0.00
c58	58229.46348	60195	64538	10.83	61612	5.81
c59	45222.45428	45880	50293	11.21	46604	3.05
c60	53809.67879	54974	57245	6.38	56020	4.11
c61	96605.02529	98030.1	106186.5	9.92	98517.75	1.98
c62	130723.5808	135403	157096	20.17	136704.3784	4.58
c63	94012.32584	95536.9	107015	13.83	96325.27273	2.46
c64	127572.1129	130689.8462	148393.5	16.32	132272.5714	3.68

Abbildung 6.11: Experiment 2.1: CANAD-C (siehe Seite 134).

	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC
	000	001	010	011	300	301	310	311
	sec	sec	sec	sec	sec	sec	sec	sec
A1	0.42	0.44	0.42	0.42	0.17	0.16	0.21	0.21
A2	0.78	0.68	0.49	0.6	0.61	0.52	0.45	0.43
A3	0.39	0.52	0.32	0.53	0.32	0.34	0.34	0.36
A4	0.62	0.88	0.74	0.61	0.46	0.63	0.69	0.66
B1	1.26	1.29	1.09	1.18	1.21	1.13	0.75	1.31
B2	2.09	2.27	1.95	1.99	2.09	2.3	1.86	2.17
B3	1.69	1.34	1.51	1.5	1.73	1.35	1.55	1.32
B4	0.86	1.27	0.91	1.29	0.77	0.99	0.9	0.89
C1	2.16	2.14	2.17	2.24	1.76	2.04	1.79	2
C2	12.96	12.4	14.5	12.61	14.34	13.55	14.92	13.8
C3	2.81	2.97	3	2.96	2.69	2.96	2.85	2.91
C4	12.37	14.14	12.84	15.44	12.73	15.6	11.76	15
D1	4.26	3.78	3.83	3.84	3.45	3.33	3.24	2.91
D2	6.22	8.12	6.02	6.69	6.09	8.04	5.73	5.95
D3	3.38	3.87	5.08	4.98	2.82	3.56	4.31	4.27
D4	36.64	39.21	20.46	24.47	37.29	37.31	19.2	27.04
E1	15.72	16.97	16.89	12.55	15.36	19.56	16.8	15.93
E2	35.12	20.67	29.18	25.94	34.84	23.09	26.66	26.8
E3	15.39	16.7	19.85	12.31	14.43	15.11	14.98	12.22
E4	67.17	41.44	66.6	40.93	67.24	43.98	69.76	33.78
F1	3.55	4.57	4.52	7.45	3.65	4.27	3.64	5.33
F2	125.01	85.39	67.11	94.06	117.63	75.32	68.88	82.73
F3	28.99	21.53	29.76	26.79	31.97	24.16	30.55	23.44
F4	5.07	4.2	3.16	4.19	6.5	3.5	3.22	3.77
G1	13.9	12.21	7.79	10.03	12.75	10.88	6.11	7.55
G2	36.97	34.14	46.4	46.04	39.34	38.19	47.08	41.31
G3	10.32	9.76	7.88	8.87	8.75	8.86	6.62	6.91
G4	180.98	154.41	173.69	108.89	185.03	113.68	155.96	110.98
H1	10.14	10.17	11.01	10.96	8.1	8.18	7.83	7.51
H2	20.18	21.33	24.3	21.99	21.17	18.36	23.1	19.86
H3	7.12	7.51	9.52	9.52	3.95	4.9	6.48	6.9
I1	4.77	5.15	4.87	5.48	2.96	3.45	3.29	3.55
I2	23.44	12.68	18.14	21.44	13.39	10.99	11.67	16.83
I3	32.98	24.72	48.67	36.1	33.07	21.85	37.57	46.19
I4	29.14	67.76	65	37.28	26.2	61.31	52.62	28.99
J1	12.91	16.06	14.79	12.96	9.49	10.91	9.53	8
J2	68.02	77.48	78.66	75.34	59.49	98.19	58.73	64.58
J3	239.32	475.4	362.53	302.23	72.05	191.05	117.01	166.49
J4	433.02	408.09	393.76	529.82	347.32	437.26	339.41	490.11
K1	27.66	33.92	34.72	34.19	44.35	28.29	29.07	34.08
L1	1409.8	1032.6	891.9	1545.3	1191.1	931.45	1408.1	803.23
Summe	2945.6	2710.18	2506.03	3122.01	2458.66	2300.6	2625.22	2148.3

Abbildung 6.12: Experiment 2.3: PAD (siehe Seite 137).

			NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC
			000	001	010	011	300	301	310	311
	best lower bound	best solution	SQ	SQ	SQ	SQ	SQ	SQ	SQ	SQ
c33	423848	423848	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
c35	371467.0847	371475	0.06	0.34	0.14	0.14	0.06	0.34	0.14	0.27
c36	643036	643036	0.42	1.27	0.82	0.98	0.42	1.27	0.82	0.98
c37	91333.66015	94218	0.08	0.08	0.08	0.09	0.08	0.08	0.08	0.09
c38	132315.7426	137854	1.46	0.18	1.46	0.18	1.46	0.18	1.46	0.18
c39	95675.94029	97914	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
c40	131545.752	136067	0.36	0.37	0.36	0.37	0.36	0.37	0.36	0.37
c41	429398	429398	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
c42	585998.1248	586077	0.59	0.59	0.59	0.83	0.59	0.59	0.59	0.83
c43	464509	464509	0.20	0.18	0.20	0.18	0.20	0.18	0.20	0.18
c44	604198	604198	0.00	0.00	0.00	0.10	0.00	0.00	0.00	0.10
c45	73130.90357	74902	0.04	0.09	0.04	0.09	0.04	0.09	0.04	0.09
c46	110929.4482	115754	0.24	0.22	0.24	0.22	0.24	0.22	0.24	0.22
c47	74025.16598	74991	0.16	0.67	0.16	0.80	0.16	0.67	0.16	0.80
c48	103714.0588	107102	0.69	0.39	0.69	0.39	0.69	0.39	0.69	0.39
c49	53714.15788	53964	10.80	0.08	10.80	0.08	10.80	0.08	10.80	0.08
c50	90207.66494	94119	1.02	0.50	1.02	0.50	1.02	0.50	1.02	0.50
c51	51768.33327	52070	8.53	0.19	8.53	0.19	8.53	0.19	8.53	0.19
c52	94216.68396	97881	2.02	0.61	2.02	0.61	2.02	0.61	2.02	0.61
c53	111763.0914	112830	0.11	0.11	0.19	0.21	0.11	0.11	0.19	0.21
c54	146693.242	149798	0.37	0.19	0.37	0.18	0.37	0.19	0.37	0.18
c55	114083.2571	114680	0.10	0.10	0.14	0.14	0.10	0.10	0.14	0.14
c56	149772.3093	152690	0.55	0.51	0.55	0.51	0.55	0.51	0.55	0.51
c57	47602.9153	47603	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02
c58	58229.46348	60195	0.02	0.45	0.02	0.45	0.02	0.45	0.02	0.45
c59	45222.45428	45880	11.25	11.25	11.25	11.25	11.25	11.25	11.25	11.25
c60	53809.67879	54974	1.06	0.33	1.06	0.33	1.06	0.33	1.06	0.33
c61	96605.02529	98030.1	0.74	0.62	0.65	0.78	0.84	0.62	0.65	0.78
c62	130723.5808	135403	0.70	0.57	0.86	0.52	0.70	0.57	0.86	0.52
c63	94012.32584	95536.9	0.60	0.58	0.90	0.87	0.60	0.58	0.90	0.87
c64	127572.1129	130689.8462	0.45	0.88	0.61	0.95	0.45	0.88	0.61	0.95
		Ø SQ	**1.38**	**0.69**	**1.41**	**0.71**	**1.38**	**0.69**	**1.41**	**0.71**

Abbildung 6.13: Experiment 2.4: CANAD-C (siehe Seite 137).

	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC
	00	00	01	01	10	10	11	11
	sec	nodes	sec	nodes	sec	nodes	sec	nodes
A1	0.2	1	0.21	1	0.2	1	0.2	1
A2	0.45	5	0.49	5	0.45	3	0.48	5
A3	0.28	3	0.38	3	0.33	3	0.37	3
A4	0.68	15	0.68	7	0.67	15	0.67	7
B1	0.75	9	1.29	27	0.75	9	1.33	27
B2	1.81	41	2.16	43	1.82	41	2.21	41
B3	1.5	25	1.23	19	1.57	29	1.35	21
B4	0.81	9	0.85	9	0.84	9	0.91	9
C1	1.76	13	1.99	15	1.76	13	2.04	15
C2	14.69	459	13.62	335	14.38	449	14.38	331
C3	2.81	29	2.88	41	2.81	31	2.94	41
C4	11.76	281	15.34	443	11.6	275	15.68	435
D1	3.2	32	2.97	17	3.18	32	3.06	17
D2	5.64	87	5.96	39	5.71	89	5.94	39
D3	4.22	31	4.33	29	4.16	31	4.5	29
D4	19.14	341	26.56	451	18.1	329	27.27	459
E1	16.35	335	16.87	378	16.82	349	15.94	341
E2	26.23	805	26.78	747	26.79	829	26.41	761
E3	14.7	413	13.68	459	18.96	553	12.41	443
E4	68.82	1831	45.87	1319	68.54	1803	35.04	940
F1	3.6	21	5.95	42	4.26	39	4.75	20
F2	66.81	1005	84.81	1231	70.59	1026	82.87	1165
F3	29.69	541	23.64	351	31.92	551	23.81	326
F4	3.11	29	3.8	37	3.17	29	3.85	37
G1	5.96	33	7.77	51	5.88	33	7.72	51
G2	44.26	441	36.36	425	44.46	445	41.34	511
G3	6.57	35	6.55	45	7.06	45	6.88	51
G4	151.57	2001	114.77	1491	149.17	1983	113.76	1446
H1	7.48	11	7.74	11	7.29	11	7.49	11
H2	22.7	85	19.9	111	22.64	85	20.1	113
H3	6.4	9	6.44	9	6.8	17	6.98	17
I1	3.09	5	3.52	5	3.08	5	3.55	7
I2	11.65	87	18.57	147	12.58	101	16.84	137
I3	36.41	293	45	409	36.76	297	45.73	405
I4	50.57	733	28.95	331	51.18	729	29.42	333
J1	9.41	31	10.15	63	11.17	47	7.99	35
J2	58.09	313	68.24	553	65.4	371	64.55	473
J3	115.2	1184	166.31	1682	105.29	986	166.94	1538
J4	335.75	2885	451.04	3110	409.44	3104	492.9	3267
K1	28.91	131	34.2	125	29.39	139	34.37	125
L1	1384.9	4339	813.96	2382	1454.3	4508	813.02	2350
Summe	2577.93	18977	2141.81	16998	2731.27	19444	2167.99	16383

Abbildung 6.14: Experiment 2.5a: PAD (siehe Seite 139).

	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC
	00	00	01	01	10	10	11	11
	sec	nodes	sec	nodes	sec	nodes	sec	nodes
A1	0.17	1	0.17	1	0.17	1	0.16	1
A2	0.24	5	0.32	7	0.23	5	0.31	7
A3	0.22	3	0.24	3	0.21	3	0.25	3
A4	0.38	13	0.31	9	0.38	13	0.32	9
B1	0.8	19	0.67	21	0.83	19	0.69	21
B2	2.25	75	1.83	59	2.25	75	1.82	59
B3	1.11	31	1.34	37	1.14	31	1.36	37
B4	0.61	15	0.61	15	0.61	15	0.64	15
C1	0.99	15	1.06	15	0.96	15	1.09	15
C2	19.25	553	16.23	407	19.41	549	16.5	407
C3	1.75	35	2.08	44	1.8	35	2.08	44
C4	30.1	746	16.02	426	29.92	744	16.25	429
D1	2.01	33	1.99	27	2.1	33	2.03	27
D2	4.4	71	4.45	73	4.5	71	4.31	73
D3	2.33	43	2.73	39	2.35	43	2.67	39
D4	37.37	882	43.7	775	38.26	882	44.41	775
E1	23.57	539	19.48	459	23.87	539	19.85	459
E2	26.51	763	25.37	695	26.8	763	25.5	687
E3	12.22	502	6.58	246	12.35	434	6.36	210
E4	38.14	1211	37.5	947	38.5	1205	37.36	947
F1	3.65	61	2.05	23	3.83	63	2.11	23
F2	144.23	2456	102.48	1728	148.54	2368	104.68	1728
F3	33.53	613	42.63	657	34.96	613	43	651
F4	2.66	33	3.62	45	2.75	33	3.64	45
G1	7.9	85	11.4	119	8.01	85	11.37	119
G2	54.84	633	50.49	585	55	633	51.32	585
G3	5.52	55	5.99	59	5.64	55	6.15	59
G4	211.95	2457	154.44	1921	219.49	2457	156.43	1921
H1	5.85	37	6.06	37	5.92	37	6.05	37
H2	40.42	293	28.18	253	40.39	293	28.08	253
H3	3.35	23	3.4	23	3.42	23	3.48	23
I1	2.07	19	1.64	13	2.14	20	1.71	13
I2	10.43	125	20.39	229	10.73	125	20.9	229
I3	29.8	331	33.05	319	30.67	333	33.06	319
I4	69.13	791	61.12	845	71.23	791	61.81	845
J1	5.1	31	7.37	43	5.39	31	7.56	49
J2	66.36	573	67.32	533	76.33	603	68.29	533
J3	55.68	683	111.52	1125	60.18	682	63.52	652
J4	596.63	4511	682.58	5033	711.99	4663	694.86	5055
K1	50.78	287	76.28	347	52.88	287	76.96	347
L1	7898.8	24325	5713.5	17322	7599.2	24317	5726.6	17299
Summe	9503.10	43977	7368.19	35564	9355.33	43987	7355.54	35049

Abbildung 6.15: Experiment 2.5b: PAD (siehe Seite 139).

		NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC	NDBC
		00	00	01	01	10	10	11	11
	optimal	sec	nodes	sec	nodes	sec	nodes	sec	nodes
r10.2	346813.5	3660.9	45713	322.6	2629	7834.8	47034	357.73	3145
r10.3	488015	4278.6	45207	169.08	1706	5234.8	47102	234.59	2076
r11.4	870451	11653	34367	13649	40208	11768	34173	13588	40102
r11.9	4579353	4245.6	13959	8396.5	24924	4422.6	14289	8410.7	25323
r12.1	1639443	10375	16509	5971.4	4446	10977	16967	2901.2	4297
r12.5	4669799	5512.6	9623	6662.4	10292	5663.1	9702	6418.5	10281
r12.7	7635270	296.24	392	411.61	503	309.56	392	410.3	504
r12.8	10067742	796.99	1279	803.29	1279	841.99	1288	816.04	1288
r13.2	263800	313	2098	242.08	1782	332.43	2096	175.58	1205
r13.3	365836	176.09	1079	232.82	1211	188.49	1076	236.07	1210
r13.4	150977	547.54	5220	428.37	4044	527.88	4918	502.5	4269
r13.5	282682	5247.7	44514	1982.1	12435	5673.7	44584	2126.7	12572
r14.1	403414	603.7	2288	534.84	1618	636.31	2346	496.78	1448
r14.4	437607	4388.7	9648	2425	5573	4086.9	9618	2552.5	5753
r15.1	1000787	3019.8	4483	2644.9	3549	2993.2	4734	2692.7	3617
r16.2	239500	486.43	3260	770.1	3315	516.83	3194	706.26	3317
r16.3	325671	491.18	3657	422.02	2601	483.24	3503	446.87	2533
r16.5	241801	302.39	1621	158.64	645	320.11	1626	181.74	710
r16.6	337762	2665.9	18876	1130.8	7055	2683.6	19061	1276.3	7290
r17.1	354138	115.02	281	149.37	377	279.7	365	132.66	338
r17.4	370590	6576.2	8303	2801.3	6847	2926.4	7812	2610.4	6429
Summe		65752.58	272377	50308.22	137039	68700.64	275880	47274.12	137707

Abbildung 6.16: Experiment 2.6: CANAD-R2 (siehe Seite 141).

			NDBC	NDBC	NDBC	NDBC
			00	01	10	11
	best lower bound	best solution	SQ	SQ	SQ	SQ
c33	423848	423848	0.00	0.00	0.00	0.00
c35	371467.0847	371475	0.14	0.27	0.14	0.27
c36	643036	643036	0.82	0.98	0.82	0.98
c37	91333.66015	94218	0.08	0.09	0.08	0.09
c38	132315.7426	137854	1.46	0.18	1.46	0.18
c39	95675.94029	97914	0.00	0.00	0.00	0.00
c40	131545.752	136067	0.36	0.37	0.36	0.37
c41	429398	429398	0.00	0.00	0.00	0.00
c42	585998.1248	586077	0.59	0.83	0.59	0.83
c43	464509	464509	0.20	0.18	0.20	0.18
c44	604198	604198	0.00	0.10	0.00	0.10
c45	73130.90357	74902	0.04	0.09	0.04	0.09
c46	110929.4482	115754	0.24	0.22	0.24	0.22
c47	74025.16598	74991	0.16	0.80	0.16	0.80
c48	103714.0588	107102	0.69	0.39	0.69	0.39
c49	53714.15788	53964	10.80	0.08	10.80	0.08
c50	90207.66494	94119	1.02	0.50	1.02	0.50
c51	51768.33327	52070	8.53	0.19	8.53	0.19
c52	94216.68396	97881	2.02	0.61	2.02	0.61
c53	111763.0914	112830	0.19	0.21	0.19	0.21
c54	146693.242	149798	0.37	0.18	0.37	0.18
c55	114083.2571	114680	0.14	0.14	0.14	0.14
c56	149772.3093	152690	0.55	0.51	0.55	0.51
c57	47602.9153	47603	0.02	0.02	0.02	0.02
c58	58229.46348	60195	0.02	0.45	0.02	0.45
c59	45222.45428	45880	11.25	11.25	11.25	11.25
c60	53809.67879	54974	1.06	0.33	1.06	0.33
c61	96605.02529	98030.1	0.65	0.78	0.65	0.78
c62	130723.5808	135403	0.86	0.52	0.86	0.52
c63	94012.32584	95536.9	0.90	0.87	0.90	0.87
c64	127572.1129	130689.8462	0.61	0.95	0.61	0.95
		Ø SQ	**1.41**	**0.71**	**1.41**	**0.71**

Abbildung 6.17: Experiment 2.7: CANAD-C (siehe Seite 143).

			NDBC	NDBC	NDBC	NDBC
			00	01	10	11
	best lower bound	best solution	SQ	SQ	SQ	SQ
c33	423848	423848	0.05	0.00	0.00	0.09
c35	371467.0847	371475	0.17	0.19	0.17	0.19
c36	643036	643036	0.59	0.75	0.59	0.75
c37	91333.66015	94218	0.07	0.65	0.07	0.65
c38	132315.7426	137854	3.37	1.81	3.37	1.81
c39	95675.94029	97914	0.79	0.00	0.79	0.00
c40	131545.752	136067	0.92	0.79	0.92	0.79
c41	429398	429398	0.00	0.00	0.00	0.00
c42	585998.1248	586077	1.27	0.85	1.27	0.85
c43	464509	464509	0.23	0.11	0.23	0.11
c44	604198	604198	0.11	0.00	0.11	0.00
c45	73130.90357	74902	0.79	1.22	0.79	1.22
c46	110929.4482	115754	0.43	1.52	0.43	1.52
c47	74025.16598	74991	1.44	0.01	1.44	0.01
c48	103714.0588	107102	2.95	1.85	2.95	1.85
c49	53714.15788	53964	1.05	0.52	1.05	0.52
c50	90207.66494	94119	2.17	1.92	2.17	1.92
c51	51768.33327	52070	2.20	0.75	2.20	0.75
c52	94216.68396	97881	4.53	2.23	4.53	2.23
c53	111763.0914	112830	0.55	0.38	0.55	0.38
c54	146693.242	149798	0.47	0.78	0.47	0.78
c55	114083.2571	114680	0.14	0.17	0.14	0.17
c56	149772.3093	152690	0.48	0.48	0.48	0.48
c57	47602.9153	47603	0.34	0.14	0.34	0.14
c58	58229.46348	60195	6.16	2.82	6.16	2.82
c59	45222.45428	45880	3.75	1.54	3.75	1.54
c60	53809.67879	54974	5.83	2.22	5.83	2.22
c61	96605.02529	98030.1	0.55	0.09	0.55	0.09
c62	130723.5808	135403	0.80	0.99	0.80	0.99
c63	94012.32584	95536.9	0.65	1.10	0.65	1.10
c64	127572.1129	130689.8462	0.92	1.44	0.92	1.44
		Ø SQ	**1.41**	**0.88**	**1.41**	**0.88**

Abbildung 6.18: Experiment 2.7: CANAD-C (siehe Seite 143).

Alpha	MEAN SQ	MAX SQ	STDABW	MEAN+STDABW	time
0.01	7.63	21.26	10.26	17.89	7350.92
0.02	6.85	19.37	9.32	16.17	7286.82
0.03	6.47	19.96	8.72	15.19	7300.12
0.04	6.05	17.06	8.04	14.09	7183.05
0.05	5.16	15.52	6.96	12.11	7148.99
0.06	4.57	15.25	6.07	10.64	7153.84
0.07	4.19	13.45	5.61	9.80	7123.45
0.08	4.14	13.64	5.72	9.86	7093.01
0.09	3.76	13.31	5.16	8.91	6776.64
0.1	3.60	12.27	5.04	8.64	6746.77
0.11	3.07	12.41	4.48	7.56	6760.84
0.12	2.90	11.25	4.21	7.11	6703.10
0.13	2.72	11.25	4.08	6.81	6693.29
0.14	2.38	11.25	3.70	6.08	6675.27
0.15	2.01	11.25	3.31	5.32	6517.24
0.16	1.95	11.25	3.30	5.24	6553.85
0.17	1.59	11.25	2.98	4.56	6285.66
0.18	1.46	11.25	2.70	4.16	6180.09
0.19	1.37	11.25	2.63	4.00	6328.95
0.2	1.03	11.25	2.31	3.33	6060.05
0.21	0.98	11.25	2.27	3.25	5795.89
0.22	0.97	11.25	2.27	3.24	5665.86
0.23	0.80	11.25	2.17	2.97	5414.40
0.24	0.72	11.25	2.12	2.84	4921.79
0.25	0.75	11.25	2.12	2.87	4387.86
0.26	0.74	11.25	2.13	2.86	4248.53
0.27	0.68	11.25	2.11	2.79	3587.12
0.28	0.69	11.25	2.12	2.81	3346.05
0.29	0.71	11.25	2.13	2.84	2801.11
0.3	0.80	11.25	2.21	3.00	2624.27
0.31	4.87	90.89	17.74	22.61	2309.51
0.32	4.90	90.89	17.74	22.64	2327.56
0.33	4.92	90.89	17.74	22.67	2014.55
0.34	5.31	90.89	17.81	23.11	1457.06
0.35	5.26	90.89	17.80	23.06	1106.39
0.36	5.80	90.89	17.97	23.76	774.70
0.37	8.58	90.89	23.29	31.87	512.19
0.38	17.66	133.49	39.90	57.57	406.58
0.39	20.12	133.49	41.42	61.54	239.04
0.4	20.22	133.49	41.43	61.65	235.67

Abbildung 6.19: Experiment 2.8 (siehe Seite 145).

Beta	MEAN SQ	MAX SQ	STDABW	MEAN+STDABW	time
0.01	12.83	53.95	17.93	30.76	8210.39
0.02	13.50	52.14	19.42	32.92	8193.95
0.03	12.62	47.55	17.32	29.94	8106.44
0.04	9.94	47.03	15.89	25.83	8026.20
0.05	5.99	47.03	11.89	17.87	8008.26
0.06	2.32	20.10	4.78	7.11	7774.75
0.07	1.21	10.63	2.39	3.60	7416.42
0.08	0.54	3.08	0.88	1.42	7255.29
0.09	0.54	1.98	0.76	1.30	6904.80
0.1	0.50	1.44	0.65	1.15	5859.09
0.11	0.53	2.66	0.83	1.36	4332.30
0.12	0.75	2.82	1.01	1.77	3593.99
0.13	0.63	3.01	0.88	1.50	3350.90
0.14	1.33	6.28	1.93	3.25	2125.70
0.15	0.81	3.12	1.13	1.94	1914.58
0.16	1.56	14.88	3.09	4.65	1405.05
0.17	1.95	7.51	2.81	4.75	1105.70
0.18	1.24	4.65	1.82	3.06	985.28
0.19	2.81	16.05	5.36	8.17	859.05
0.2	5.08	22.90	8.31	13.38	584.15
0.21	3.14	15.34	4.92	8.06	601.77
0.22	1.94	9.00	3.15	5.08	589.34
0.23	1.40	7.37	2.22	3.62	549.41
0.24	5.47	34.20	10.44	15.91	471.52
0.25	9.45	32.44	14.01	23.46	361.12
0.26	7.81	26.71	11.57	19.38	359.62
0.27	6.01	21.55	9.00	15.01	367.59
0.28	3.61	17.67	5.61	9.22	358.70
0.29	2.68	9.92	3.90	6.57	358.04
0.3	4.84	56.95	12.86	17.70	338.36
0.31	6.61	56.73	16.74	23.36	319.44
0.32	10.66	50.46	19.46	30.13	297.07
0.33	16.59	47.03	22.97	39.56	237.32
0.34	15.68	45.34	21.63	37.31	236.19
0.35	14.30	41.94	20.02	34.32	235.06
0.36	13.10	39.36	18.60	31.70	238.02
0.37	12.02	35.05	16.71	28.73	235.31
0.38	10.02	30.23	13.78	23.79	238.60
0.39	10.23	28.70	13.68	23.91	233.51
0.4	8.40	24.08	11.19	19.59	234.54

Abbildung 6.20: Experiment 2.8 (siehe Seite 145).

Literaturverzeichnis

[Abara, 1989] Abara, J. (1989). Applying integer linear programming to the fleet assignment problem. *Interfaces*, 19(4):20–28. 95, 96

[Ahuja et al., 1993] Ahuja, R. K., Magnanti, T. L., and Orlin, J. B. (1993). *Network flows : theory, algorithms, and applications*. Prentice Hall, Englewood Cliffs, New Jersey. 16, 24, 42

[Allen et al., 1987] Allen, E., Helgason, R., Kennington, J., and Shetty, B. (1987). A generalization of Polyak's convergence result for subgradient optimization. *Mathematical Programming*, 37:309–317. 31

[Armacost et al., 2002] Armacost, A., Barnhart, C., and Ware, K. (2002). Composite variable formulations for express shipment service network design. *Transportation Science*, 36(1):1–20. 12

[Atamtürk, 1999] Atamtürk, A. (1999). On capacitated network design cut–set polyhedra. *Math. Program.*, Ser. B 92:425—-437. 18

[Aykin, 1994] Aykin, T. (1994). Lagrangian-relaxation based approaches to capacitated hub-and-spoke network design problem. *European Journal of Operational Research*, 79(3):501–523. 12

[Balakrishnan et al., 1997] Balakrishnan, A., Magnanti, T., and Mirchandani, P. (1997). Network design. *Annotated Bibliographies in Combinatorial Optimization*, pages 311–334. 13

[Balas and Christofides, 1981] Balas, E. and Christofides, N. (1981). A restricted Lagrangian approach to the traveling salesman problem. *Mathematical Programming*, 21:19–46. 69

[Barnhart et al., 2003] Barnhart, C., Belobaba, P., and Odoni, A. R. (2003). Applications of operations research in the air transport industry. *Transportation Science*, 37(4):368–391. 10

[Barnhart et al., 1998] Barnhart, C., Boland, N., Clarke, L., Johnson, E., Nemhauser, G., and Shenoi, R. (1998). Flight string models for aircraft fleeting and routing. *Transportation Science*, 32(3):208–220. 83

[Barnhart and Kim, 1997] Barnhart, C. and Kim, D. (1997). Algorithms. *Computer-Aided Transit Scheduling*, pages 259–283. 18

[Belobaba, 1989] Belobaba, P. (1989). Application of a probabilistic decision model to airline seat inventory control. *Operations Research*, 37(2):183–197. 92

[Ben-Akiva and Bierlaire, 1985] Ben-Akiva, M. and Bierlaire, M. (1985). *Discrete choice analysis*. The MIT Press Series in Transportation Studies. 85

[Ben-Akiva and Bierlaire, 1999] Ben-Akiva, M. and Bierlaire, M. (1999). *Discrete Choice Methods and their Applications to Short Term Travel Decisions*. Handbook of Transportation Sciences. 85

[Berdy, 2002] Berdy, P. (2002). *Developing effective route networks*. McGraw-Hill. 12

[Berge, 1994] Berge, M. (1994). Timetable optimization. In *AGIFORS Symposium*. Boeing. 12

[Berge and Hopperstad, 1993] Berge, M. and Hopperstad, C. (1993). Demand driven dispatch: A method for dynamic aircraft capacity assignment, models and algorithms. *Operations Research*, 41(1):153–168. 96

[Bienstock, 1999] Bienstock, D. (1999). Experiments with a network design algorithm using epsilon-approximate linear programs. Technical report, CORC Report 1999-4. 18

[Bienstock et al., 1998] Bienstock, D., Chopra, S., Günlük, O., and Tsai, C.-Y. (1998). Minimum cost capacity installation for multicommodity network flows. *Math. Program.*, 81:177–19. 18

[Boer et al., 2002] Boer, S., Freling, R., and Piersma, N. (2002). Mathematical programming for network revenue management revisited. *European Journal of Operational Research*, 137:72–92. 106

[Boyd, 2002] Boyd, A. (2002). Revenue management and dynamic pricing. In *IMA Tutorial: Supply Chain and Logistics Optimization*. 106

[Camerini et al., 1975] Camerini, P., Fratta, L., and Maffioli, F. (1975). On improving relaxation methods by modified gradient techniques. *Mathematical Programming Study*, 3(26–34). 31

[Carl and Gesing, 2000] Carl, G. and Gesing, T. (2000). *Informationsmanagement im Verkehr*, chapter Flugplanung als Instrument des Informationsmanagements zur Ressourcenplanung und -steuerung einer Linienfluggesellschaft, pages 167–198. Physica-Verlag Heidelberg. 10, 12

[Carraresi et al., 1995] Carraresi, P., Frangioni, A., and Nonato, M. (1995). Applying bundle methods to optimization of polyhedral functions: An applications-oriented development. *Ricerca Operativa*, XXV(74):5–49. 32

[Carrier, 2003] Carrier, E. (2003). *Modeling Airline Passenger Choice: Passenger Preference for Schedule in the Passenger Origin-Destination Simulator (PODS)*. Master thesis, MIT. 88

[Chouman et al., 2003] Chouman, M., Crainic, T. G., and Gendron, B. (2003). A cutting-plane algorithm based on cutset inequalities for multicommodity capacitated fixed charge network design. unpublished. 61, 64

[Clarke and Gong, 1998] Clarke, L. and Gong, P. (1995 (revised 1998)). Capacitated network design with column generation. Technical report, Industrial and Systems Engeneering, Georgia Institute of Technology, Atlanta, Georgia. 18

[Clarke et al., 1996] Clarke, L., Hane, C., Johnson, E., and Nemhauser, G. (1996). Maintenance and crew considerations in fleet assignment. *Transportation Science*, 30:249–260. 15

[Cohn and Barnhart, 2003] Cohn, A. M. and Barnhart, C. (2003). Improving crew scheduling by incorporating key maintenance routing decisions. *Operations Research*, 51(3):387 –396. 84

[Crainic, 2000] Crainic, T. (2000). Service network design in freight transportation. *European Journal of Operational Research*, 122:272–288. 13

[Crainic et al., 2001a] Crainic, T., Frangioni, A., and Gendron, B. (2001a). Bundle-based relaxation methods for multicommodity capacitated fixed charge network design problems. *Discrete Applied Mathematics*. 17, 20, 31

[Crainic et al., 2000a] Crainic, T., Gendreau, M., and Farvolden, J. (2000a). A simplex-based tabu search method for capacitated network design. *INFORMS Journal on Computing*, 12(3):223–236. 17

[Crainic et al., 2000b] Crainic, T., Gendreau, M., and Farvolden, J. (2000b). A simplex-based tabu search method for capacitated network design. *INFORMS Journal on Computing*, 12(3):223–236. 126, 128

[Crainic et al., 2004] Crainic, T., Gendron, B., and Hernu, G. (2004). A slope scaling/Lagrangean perturbation heuristic with long-term memory for multicommodity fixed-charge network design. *Journal of Heuristics*, 10(5):525–545. 18, 123, 126, 128

[Crainic et al., 2001b] Crainic, T. G., Frangioni, A., and Gendron, B. (2001b). Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics*, 112:73–99. 32, 39, 40

[Crowder, 1976] Crowder, H. (1976). Computational improvements for subgradient optimization. *Symposia Mathematica*, XIX:357–372. Academic Press, London. 31

[Daskin and Panayotopoulos, 1989] Daskin, M. and Panayotopoulos, N. (1989). A Lagrangian relaxation approach to assigning aircraft to routes in hub and spoke networks. *Transportation Science*, 23(2):91–99. 95

[Derigs et al., 2001] Derigs, U., Eßer, M., and Völkner, P. (2001). A meta-heuristic based decision support system for hub-optimization. Technical report, Universität zu Köln. 12

[Desaulniers and Desrosiers, 1997] Desaulniers, G. and Desrosiers, J. (1997). Daily aircraft routing and scheduling. *Management Science*, 43(6):841–855. 96

[Dobson and Lederer, 1993] Dobson, G. and Lederer, P. (1993). Airline scheduling and routing in a hub-and-spoke system. *Transportation Science*, 27(3):281–297. 12

[Dolan and More, 2002] Dolan, E. D. and More, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91(2):201 – 213. 119, 121

[Dumas and Soumis, 2004] Dumas, J. and Soumis, F. (2004). Passenger flow models for airline networks. *unpublished*. 89

[Emden-Weinert, 1998] Emden-Weinert, T. (1998). *Kombinatorische Optimierungsverfahren für die Flugdienstplanung*. PhD thesis, Berlin. 10

[Erdmann, 1999] Erdmann, A. (1999). *Combinatorial Optimization Problems arising in Airline Industry*. PhD thesis, Universität zu Köln. 12

[Etschmaier and Mathaisel, 1984] Etschmaier, M. and Mathaisel, D. (1984). Aircraft scheduling: The state of the art. In *AGIFORS Symposium*. 12, 88, 95

[Fahle, 2002] Fahle, T. (2002). *Integrating concepts from constraint programming and operations research algorithms*. Dissertation, Universität Paderborn. 10

[Farkas, 1996] Farkas, A. (1996). *The influence of network effects and yield management on airline fleet assignment decisions*. PhD thesis, MIT. Supervised by P. Belobaba. 88, 106

[Feo and J.F., 1989] Feo, T. and J.F., B. (1989). Flight scheduling and maintenance base planning. *Management Science*, 35(12):1415–1432. 12

[Ferguson and Dantzig, 1956] Ferguson, A. and Dantzig, G. (1956). The allocation of aircraft to routes - an example of linear programming under uncertain demand. *Management Science*, 3. 95

[Frangioni, 1997] Frangioni, A. (1997). *Dual-Ascent Methods and Multicommodity Flow Problems*. Ph.d. thesis, Dipartimento di informatica, Università di Pisa. 29, 32, 37, 39, 40, 111

[Gendron and Crainic, 1994a] Gendron, B. and Crainic, T. (1994a). Relaxations for multicommodity capacitated network design problems. Technical report, Center for Research on Transportation, Montreal, Canada. 14, 17, 20, 113

[Gendron et al., 1998] Gendron, B., Crainic, T., and Frangioni, A. (1998). Multicommodity capacitated network design. *Telecommunications Network Planning*, pages 1–19. 17

[Gendron and Crainic, 1994b] Gendron, B. and Crainic, T. G. (1994b). Relaxations for multicommodity capacited network design problems. Publication crt-965, Centre de recherche sur les transports, Université de Montréal. 61

[Gendron and Crainic, 1996] Gendron, B. and Crainic, T. G. (1996). Bounding procedures for multicommodity capacitated fixed charge network design problems. Publication crt-96-06, Centre de recherche sur les transports, Université de Montréal. 113

[Ghamlouche et al., 2003] Ghamlouche, I., Crainic, T., and Gendreau, M. (2003). Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. *Operations research*, 51(4):655–667. 17, 122, 126, 128

[Ghamlouche et al., 2004] Ghamlouche, I., Crainic, T., and Gendreau, M. (2004). Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design. *Annals of Operations research*, 131(1–4):109–133. 17, 122, 123, 126, 128

[Glover et al., 1982] Glover, F., Glover, R., Lorenzo, J., and McMillan, C. (1982). The passenger-mix problem in the scheduled airlines. *Interfaces*, 12(3):73–80. 106

[Günlük, 1999] Günlük, O. (1999). A branch-and-cut algorithm for capacitated network design problems. *Math. Program.*, Ser. A 86:17–39. 18

[Götz et al., 1999] Götz, S., Grothklags, S., Kliewer, G., and Tschöke, S. (1999). Solving the weekly fleet assignment problem for large airlines. In *Proceedings of the Third Metaheuristics International Conference (MIC 1999)*, pages 241–246, Angra dos Reis, Brasil. 96

[Götz et al., 2004] Götz, S., Grothklags, S., Kliewer, G., and Tschöke, S. (2004). A local search approach for the weekly fleet assignment problem. submitted to the Journal of Transportation Science. 96

[Grünert et al., 2000] Grünert, T., Büdenbender, K., and Sebastian, H.-J. (2000). A hybrid tabu search/branch and bound algorithm for the direct flight network design problem. *Transportation Science*, 34(4):364–380. 12

[Grothklags, 2000] Grothklags (2000). Simulated Annealing für das Fleet Assignment Problem. Diplomarbeit, Universität Paderborn. 96, 98

[Grothklags, 2003] Grothklags, S. (2003). Fleet assignment with connection dependent ground times. In Battista, G. D. and Zwick, U., editors, *11th Annual European Symposium on Algorithms (ESA2003)*, volume LNCS 2832, pages 667–678. 83, 96, 97

[Gu et al., 1994] Gu, Z., Johnson, E., Nemhauser, G., and Wang, Y. (1994). Some properties of the fleet assignment problem. *Operations Research Letters*, 15:59–71. 96

[Guignard, 1998] Guignard, M. (1998). Efficient cuts in Lagrangean 'relax-and-cut' schemes. *European Journal of Operational Research*, 105:216–223. 74

[Hane et al., 1995] Hane, C., Barnhart, C., Johnson, E., Marsten, R., Nemhauser, G., and Sigismondi, G. (1995). The fleet assignment problem: solving a large-scale integer program. *Mathematical Programming*, 70:211–232. 95, 96

[Hiriart-Urruty and Lemaréchal, 1993] Hiriart-Urruty, J. and Lemaréchal, C. (1993). *Convex Analysis and Minimization Algorithms II*. Number 306 in A Series of Comprehensive Studies in Mathematics. Springer-Verlag. 31, 39

[Holmberg and Yuan, 2000] Holmberg, K. and Yuan, D. (2000). A Lagrangean heuristic based branch-and-bound approach for the capacitated network design problem. *Operations Research*, 48:461–481. 17, 31, 42, 46, 56, 77

[Hunting et al., 2001] Hunting, M., Faigle, U., and Kern, W. (2001). A Lagrangian relaxation approach to the edge-weighted clique problem. *European Journal of Operational Research*, 131(1):119–131. 74

[ILOG, 2003] ILOG, S. (2003). *ILOG CPLEX 9.0 User's Manual*. 45, 122

[Jacobs and Günther, 2000] Jacobs, T. L. and Günther, D. (2000). Benefits and barriers associated with the integration of airline pricing, yield management and scheduling decisions. In *Proceedings of the 40th Annual Symposium of the Airline Group of the International Federation of Operational Research Societies (AGIFORS)*. 97

[Jacobs et al., 2000a] Jacobs, T. L., Ratliff, R., and Smith, B. C. (2000a). Soaring with synchronized systems. *OR/MS Today*, pages 36–44. 84, 97

[Jacobs et al., 2000b] Jacobs, T. L., Smith, B. C., and Johnson, E. (2000b). System and method for incorporating origination and destination effects into a vehicle assignment process. Sabre Inc. United States Patent No.: 6,076,067. 89, 97

[Jaillet et al., 1996] Jaillet, P., Song, G., and Yu, G. (1996). Airline network design and hub location problems. *Location Science*, 4(3):195–212. 12, 19

[Jarrah et al., 2000] Jarrah, A., Goodstein, J., and Narasimhan, R. (2000). An efficient airline re-fleeting model for the incremental modification of planned fleet assignments. *Transportation Science*, 34(4):349–363. 96

[Kellerer et al., 2004] Kellerer, H., Pferschy, U., and Pisinger, D. (2004). *Knapsack Problems*. Springer. 66

[Kelley, 1960] Kelley, J. (1960). The cutting plane method for solving convex programs. *Journal of the SIAM*, 8:703–712. 33

[Kiwiel, 1985] Kiwiel, K. C. (1985). *Methods of descent for nondifferentiable optimization*. Springer-Verlag, Berlin, Heidelberg. 29, 33, 39

[Klabjan, 2005] Klabjan, D. (2005). Large-scale models in the airline industry. In Desaulniers, G., Desrosiers, J., and Solomon, M., editors, *Column Generation*. Kluwer Academic Publishers. to appear. 10

[Klabjan et al., 2002] Klabjan, D., Johnson, E., Nemhauser, G., Gelman, E., and Ramaswamy, S. (2002). Airline crew scheduling with time windows and plane count constraints. *Transportation Science*, 36:337–348. 84

[Kliewer and Unruh, 1998] Kliewer, G. and Unruh, P. (1998). Parallele Simulated Annealing Bibliothek: Entwicklung und Einsatz in der Flugplanoptimierung. Diplomarbeit, Universität Paderborn. 98

[Klincewicz and Rosenwein, 1995] Klincewicz, J. and Rosenwein, M. (1995). The airline exception scheduling problem. *Transportaiton Science*, 29(1):4–16. 96

[Klose, 2002] Klose, A. (2002). Ganzzahlige Optimierung. Vorlesungsunterlagen, Universität Zürich. 61, 71

[Kniker, 1998] Kniker, T. (1998). *Itinerary-based airline fleet assignment*. PhD thesis, MIT. Supervised by C. Barnhart. 88

[Koberstein, 2002] Koberstein, A. (2002). Heuristische und exakte Verfahren für das Netzwerkentwurfproblem mit Kantenkapazitäten. Diplomarbeit, Universität Paderborn. 80

[Koppelman and Sethi, 2000] Koppelman, F. and Sethi, V. (2000). *Closed Form Discrete Choice Models*. Handbook of Transport Modeling. Pergamon Press. 85

[Lederer and Nambimadom, 1998] Lederer, P. and Nambimadom, R. (1998). Airline network design. *Operations Research*, 46(6):785–804. 12

[Lefeld and Pölt, 1995] Lefeld, M. and Pölt, S. (1995). Standardwochen Marktmodell für OPNET. Analyse des SPEED-Marktmodells. Technical report, Lufthansa Systems. 87

[Leibold, 2001] Leibold, K. (2001). *Optimierung von Flugplänen. Anwendung quantitativer Methoden im Luftverkehr*. PhD thesis, Frankfurt. 10

[Lemaréchal, 1978] Lemaréchal, C. (1978). Bundle methods in nonsmooth optimization. In Claude Lemaréchal, R. M., editor, *Nonsmooth Optimization, Proceedings of a IIASA Workshop, March 29 - April 8, 1977*, pages 79–102. Pergamon Press. 29

[Lemaréchal, 1989] Lemaréchal, C. (1989). *Optimization*, volume 1 of *Handbook in Operations Research and Management Science*, chapter Nondifferentiable Optimization, pages 529–572. 31

[Lemaréchal et al., 1995] Lemaréchal, C., Nemirovskii, A., and Nesterov, Y. (1995). New variants of bundle methods. *Mathematical Programming*, 69:111–147. 29

[Lettovsky et al., 1999] Lettovsky, L., Johnson, E., and Smith, B. (1999). Schedule generation model. In *AGIFORS Symposium*. 12

[Littlewood, 1972] Littlewood, K. (1972). Forecasting ans control of passenger bookings. In *AGIFORS Symposium*. 91

[Lohatepanont, 2002] Lohatepanont, M. (2002). *Airline fleet assignment and schedule design: integrated models and algorithms*. PhD thesis, MIT. Supervised by C. Barnhart. 12, 83, 97

[Lohatepanont and Barnhart, 2004] Lohatepanont, M. and Barnhart, C. (2004). Airline schedule planning: Integrated models and algorithms for schedule design and fleet assignment. *Transportation Science*, 38(1):19–32. 12

[Lucena, 1992] Lucena, A. (1992). Steiner problem in graphs: Lagrangian relaxation and cutting planes. In *COAL Bulletin*, volume 21, pages 2–8. Mathematical Programming Society. 74

[LufthansaSystems, 1997] LufthansaSystems (1997). SPEED-V: Technische Dokumentation. Technical report, Lufthansa Systems. 88

[LufthansaSystems, 2005] LufthansaSystems (2005). NetLine/Plan: The network planning solution. Technical report, Lufthansa Systems. 87

[Magnanti and Wong, 1984] Magnanti, T. and Wong, R. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55. 13

[Marchand et al., 2002] Marchand, H., Martin, A., Weismantel, R., and Wolsey, L. (2002). Cutting planes in integer and mixed integer programming. *Discrete Applied Mathematics*, 123:397–446. 71

[Marsten et al., 1996] Marsten, R., Subramanian, R., and Gibbons, L. (1996). Junior analyst extraordinaire (JANE): Route development at Delta Airlines. In *AGIFORS Symposium*. 12

[Martinhon et al., 2000] Martinhon, C., Lucena, A., and Maculan, N. (2000). A relax and cut algorithm for the vehicle routing problem. 74

[Maurer, 2003] Maurer, P. (2003). *Luftverkehrsmanagement*. Oldenbourg Verlag. 10

[M.Held et al., 1974] M.Held, P.Wolfe, and H.Crowder (1974). Validation of subgradient optimization. *Math. Program.*, 6:62–88. 30

[Müller-Bungart, 2003] Müller-Bungart, M. (2003). *Prognose der Passagiernachfrage im Linienluftverkehr*. Diplomarbeit, Universität zu Köln. 86, 88

[Nemhauser and Wolsey, 1988] Nemhauser, G. L. and Wolsey, L. A. (1988). *Integer and combinatorial optimization*. John Wiley & Sons, New York. 24, 26, 61, 65, 66

[Neumann and Morlock, 1993] Neumann, K. and Morlock, M. (1993). *Operations Research*. Carl Hanser Verlag, München, Wien. 35, 36

[Nitschke, 1997] Nitschke, T. (1997). *Dynamic Fleet Assignment*. Diplomarbeit, Martin-Luther-University Halle-Wittenberg, Lufthansa Systems. 96

[Papadimitriou and Steiglitz, 1982] Papadimitriou, C. and Steiglitz, K. (1982). *Combinatorial Optimization*. Prentice Hall, Inc. 42, 52

[PARALOR, 1997] PARALOR (1997). PARALOR: Gemeinsamer Abschlußbericht. Technical report. PARALOR: Parallel Algorithms for Large Scale Operations Research Problems (BMBF Projekt). 88

[Parker, 2003] Parker, R. (2003). The evolution of market allocation modeling at Boeing. In *AGIFORS Schedule and Strategic Planning Study Group Meeting*. 88

[Phillips et al., 1991] Phillips, R., Boyd, D., and Jr., T. (1991). An algorithm for calculating consistent itinerary flows. *Transportation Science*, 25(3):225–239. 106

[Pölt, 2001] Pölt, S. (2001). Revenue management. In *AGIFORS Reservations and Yield Management Study Group Meeting*. 91

[Polyak, 1969] Polyak, B. (1969). Minimization of nonsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 9(14–29). 30

[Radicke, 1994] Radicke, U.-D. (1994). *Algorithmen für das Fleet Assignment von Flugplänen. Optimierung auf Marktmodellbasis*. Verlag Shaker. 87, 96

[Ralphs and Galati, 2005] Ralphs, T. and Galati, M. (2005). Decomposition and dynamic cut generation in integer programming. Technical report, Lehigh University, PA. To appear in Mathematical Programming. 74

[Ramming, 2002] Ramming, M. (2002). *Network Knowledge and Route Choice*. PhD thesis, MIT. Supervised by Moshe Emanuel Ben-Akiva. 88

[Rexing, 1997] Rexing, B. (1997). *Airline Fleet Assignment with time windows*. Master, MIT. Supervised by Barnhart. 97

[Rushmeier and Kontogiorgis, 1997] Rushmeier, R. and Kontogiorgis, S. (1997). Advances in the optimization of airline fleet assignment. *Transportaiton Science*, 31(2):159–169. 96

[Sabre, 2004] Sabre (2004). Sabre AirFlite Profit Manager. Technical report, Sabre. 89

[Scheidler, 2003] Scheidler, M. (2003). *Discrete Choice Models for Airline Network Management*. PhD thesis, Universität Frankfurt. 88

[Schramm and Zowe, 1992] Schramm, H. and Zowe, J. (1992). A version of the bundle idea for minimizing a nonsmooth function: conceptual idea, convergence analysis, numerical results. *SIAM J. Optim.*, 2(1):121–152. 40

[Sellmann, 2002] Sellmann, M. (2002). *Reduction techniques in constraint programming and combinatorial optimization*. Dissertation, Universität Paderborn. 10, 80

[Sharma et al., 2000] Sharma, D., Ahuja, R. K., and Orlin, J. B. (2000). Neighborhood search algorithms for the combined through-fleet assignment model. Talk at 17th International Symposium on Mathematical Programming (ISMP'00). 96

[Shenoi, 1996] Shenoi, R. (1996). *Integrated Airline Schedule Optimization: Models and solution methods*. PhD thesis, MIT Cambridge. Supervised by Barnhart. 83

[Sieber, 1995] Sieber, G. (1995). Spill and recapture user guide. Technical report, Lufthansa Systems. 88

[Sivakumar, 2003] Sivakumar, R. (2003). Codeshare optimizer maximizing codeshare revenues. In *AGIFORS Schedule and Strategic Planning Study Group Meeting*. 89

[Sosnowska, 1999] Sosnowska, D. (1999). Optimization of a simplified fleet assignment problem with metaheuristics: Simulated annealing and GRASP. In *Approximation and complexity in numerical optimization: continuous and discrete problems*. Kluwer. 97

[Sosnowska and Rolim, 2000] Sosnowska, D. and Rolim, J. (2000). Fleet scheduling optimization: A simulated annealing approach. In Burke, E. and Erben, W., editors, *Practice and Theory of Automated Timetabling III (PATAT 2000)*. Springer-Verlag Heidelberg. 97

[Soumis et al., 1980] Soumis, F., Ferland, J.-A., and Rousseau, J. (1980). A model for large-scale aircraft routing and scheduling problems. *Transportation Research B*, 14B(1/2):191–201. 12, 95

[Soumis and Nagurney, 1993] Soumis, F. and Nagurney, A. (1993). A stochastic, multiclass airline network equilibrium model. *Operations Research*, 41(4):721–730. 89

[Sridhar and Park, 2000] Sridhar, V. and Park, J. S. (2000). Benders-and-cut algorithm for fixed-charge capacitated network design problem. *European Journal of Operational Research*, 125(3):622–632. 18

[Sterzenbach and Conrady, 2003] Sterzenbach, R. and Conrady, R. (2003). *Luftverkehr*. Oldenbourg Verlag. 10, 91

[Strauß, 2001] Strauß, C. (2001). *Quantitative Personaleinsatzplanung im Airline Business*. Peter Lang Verlag, Frankfurt. 10

[Subramanian et al., 1994] Subramanian, R., Sheff, R., Quillinan, J., Wiper, D., and Marsten, R. (1994). Coldstart: Fleet assignment at Delta Air Lines. *Interfaces*, 24(1):104–120. 96

[Suhl, 1995] Suhl, L. (1995). *Computer-aided scheduling: an airline perspective*. Gabler Edition Wissenschaft. Berlin, Tech. Univ., Habil.-Schr.,1993. 10

[Talluri, 1996] Talluri, K. (1996). Swapping applications in a daily airline fleet assignment. *Transportation Science*, 30(3):237–248. 96

[Talluri and van Ryzin, 1998] Talluri, K. T. and van Ryzin, G. J. (1998). An analysis of bid-price controls for network revenue management. *Manage. Sci.*, 44(11):1577–1593. 93

[Talluri and van Ryzin, 2005] Talluri, K. T. and van Ryzin, G. J. (2005). *The Theory and Practice of Revenue Management*, volume 68 of *International Series in Operations Research and Management Science*. Springer. 89, 90

[Teodorovic, 1988] Teodorovic, D. (1988). *Airline Operations Research*. Gordon and Breach Science Publishers. 12

[Teodorovic and Krcmar-Nozic, 1989] Teodorovic, D. and Krcmar-Nozic, E. (1989). Multicriteria model to determine flight frequencies on an airline network under competitive conditions. *Transportation Science*, 23:14–25. 12

[Timajev, 2004] Timajev, L. (2004). Ein Branch-and-cut Ansatz für das Netzwerkentwurfproblem. Diplomarbeit, Universität Paderborn. 82

[Usman, 2002] Usman, K. (2002). Demand forecasting: Using advanced econometric modeling. In *AGIFORS Schedule and Strategic Planning Study Group Meeting*. 89

[Viswanathan, 1999] Viswanathan, V. (1999). Demand forecasting. In *AGIFORS Reservations and Yield Management Study Group Meeting*. 88

[Williamson, 1988] Williamson, E. (1988). *Comparison of Optimization Techniques for Origin-Destination Seat Inventory Control*. Master thesis, MIT. Supervised by Simpson. 106

[Williamson, 1992] Williamson, E. (1992). *Airline Network Seat Inventory Control: Methodologies and Revenue Impacts*. PhD thesis, MIT. Supervised by P. Belobaba. 106

[Yan and Tseng, 2002] Yan, S. and Tseng, C. (2002). A passenger demand model for airline flight scheduling and fleet routing. *Computers and Operations Research*, 29:1559–1581. 12

[Yan and Wang, 2001] Yan, S. and Wang, C. (2001). The planning of aircraft routes and flight frequencies in an airline network operations. *Journal of Advanced Transportation*, 35:33–46. 12

[Yan and Young, 1996] Yan, S. and Young, H.-F. (1996). A decision support framework for multi-fleet routing and multi-stop flight scheduling. *Transportation research A*, 30(5):379–398. 96

[Yu and Thengvall, 2002] Yu, G. and Thengvall, B. G. (2002). *Handbook of applied optimization*, chapter Airline optimization, pages 689–703. Oxford University Press. 10

[Yu and Yang, 1998] Yu, G. and Yang, J. (1998). *Handbook of combinatorial optimization*, chapter Optimization applications in the airline industry, pages 635–726. Kluwer Academic Publishers. 10

Ausgewählte Publikationen

Internationale Zeitschriften und begutachtete Konferenzen:

- Georg Kliewer and Larissa Timajev. Relax-and-cut for capacitated network design. In Proceedings of the 13th Annual European Symposium on Algorithms (ESA-2005), Springer, LNCS 3669, pages 47-58, Palma der Mallorca, Spain, 2005.
- Weber K.; Sun J.; Sun Z.; Kliewer G.; Grothklags S.; Jung N. Systems integration for revenue-creating control processes. Journal of Revenue and Pricing Management, July 2003, vol. 2, no. 2, pp. 120-137(18) Henry Stewart Publications.
- Georg Kliewer, Sven Grothklags, and Klaus Weber. Improving revenue by system integration and co-operative optimization. In Proceedings of the 43rd Annual Symposium of the Airline Group of the International Federation of Operational Research Societies (AGIFORS-2002), Honolulu, Hawaii, USA, 2002.
- Meinolf Sellmann, Georg Kliewer, and Achim Koberstein. Lagrangian cardinality cuts and variable fixing for capacitated network design. In Proceedings of the 10th Annual European Symposium on Algorithms (ESA-2002), Springer, LNCS 2461, pages 845-858, Rome, Italy, 2002.
- Georg Kliewer. Integrating market modeling and fleet assignment. In Proceedings of the 2nd international Workshop on the Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CP-AI-OR'00), Paderborn, Germany, 2000.

Internationale Konferenzen und Workshops:

- Georg Kliewer. A Branch-and-Cut System for Capacitated Network Design. In Proceedings of the 18th International Symposium on Mathematical Programming (ISMP-2003), Copenhagen, Denmark, 2003.
- Georg Kliewer, Achim Koberstein, and Meinolf Sellmann. Capacitated network design. In Proceedings of the the sixteenth triennial conference of the International Federation of Operational Research Societies (IFORS-2002), Edinburgh, Scotland, UK, 2002.
- Georg Kliewer and Achim Koberstein. Solving the capacitated network design problem in parallel. In Proceedings of the third meeting of the EURO-PAREO working group on Parallel Processing in Operations Research (PAREO), Guadeloupe, France, 2002.
- Georg Kliewer. Network design: Modeling and solving in an airline alliance context. In Proceedings of the 14. Meeting of the European Chapter on Combinatorial Optimization (ECCO XIV), Bonn, Germany, 2001.
- Georg Kliewer. Cooperative approaches for market modeling and fleet assignment. In Proceedings of the 17th International Symposium on Mathematical Programming (ISMP-2000), Atlanta, GA, USA, 2000.
- Silvia Götz, Sven Grothklags, Georg Kliewer, and Stefan Tschöke. Solving the weekly fleet assignment problem for large airlines. In Proceedings of the Third Metaheuristics International Conference (MIC-1999), pages 241-246, Angra dos Reis, Brasil, 1999.