2 Heutige Lösungsverfahren

Die gemischt-ganzzahligen Modelle unterscheiden sich, wie in Kapitel 1.1 dargestellt, von den reinen LP-Modellen nur durch die Ganzzahligkeitsrestriktionen. Somit liegt der Gedanke nahe, das Modell ohne diese Restriktionen zu lösen und die fraktionellen Werte der Integer-Variablen nach dem Lösen des LPs zu runden. Folgendes Beispiel veranschaulicht, dass bei diesem Verfahren die gerundete Lösung weit entfernt von der eigentlichen optimalen Lösung liegen kann.

Beispiel:

$$\begin{aligned} \max \quad & x_1 + 0{,}64x_2 & & \\ & 50x_1 + 31x_2 & \le & 250 \\ & 3x_1 - 2x_2 & \ge & -4 \\ & x_1, x_2 \ge 0 \text{ und integer} & & \end{aligned}$$

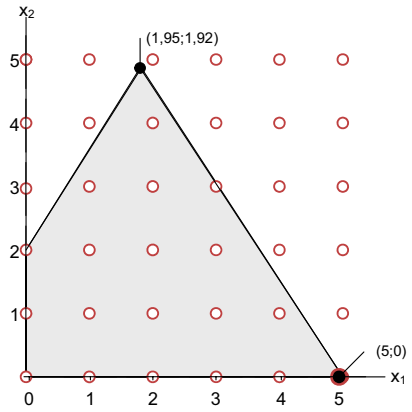

Abb. 2-1: Beispiel LP-Lösung runden

Eine optimale LP-Lösung lautet x_1=1,95, x_2=4,92 mit z=5,099. Wenn beide Werte gerundet werden (x_1=2, x_2=5) ergibt sich eine Lösung, die nicht mehr im Lösungsraum enthalten und somit unzulässig ist. Die an der LP-Lösung am nächsten liegende zulässige Integer-Lösung ist x_1=2, x_2=4 mit einem Funktionswert von z=4,56. Diese Lösung ist aber nicht optimal. Eine optimale IP-Lösung lautet x_1=5, x_2=0 mit z=5 und liegt weit von der LP-Lösung entfernt.

Somit muss es andere Lösungsverfahren für gemischt-ganzzahlige Modelle geben, welche die optimale IP-Lösung finden und dessen Optimalität beweisen.

Im Folgenden sind als geeignete Lösungsverfahren das Konzept des Branch-and-Bound-Verfahrens und der Branch-and-Cut-Ansatz dargestellt.

2.1 Branch-and-Bound-Verfahren

Das Branch-and-Bound-Verfahren stammt ursprünglich von Land and Doig (1960) [LaDo60] und war eines der ersten Methoden, welches gemischt-ganzzahlige Modelle bis zur Optimalität löst. Das Verfahren wurde u.a. von Dakin [Dak65], Beale und Small [BeSm65], Driebeck [Dri66], Forrest, Hirst und Tomlin [FoHiTo74] weiterentwickelt und wird in [LeMi01], [JoNeSa00], [LiSa99] beschrieben.

Ausgehend von dem IP-Modell in (1-2) ist die Idee des LP-basierten Branch-and-Bounds die Zerlegung des Lösungsraums F in viele kleinere disjunkte Lösungsräume F_k. Jeder Lösungsraum gehört zu einem Problem P_k, welches ein Unterproblem vom originalen Problem P ist: $P = P_1 \cup P_2 \cup ... \cup P_k \cup ... \cup P_n$. Diese Unterprobleme unterscheiden sich von

dem originalen Problem P durch zusätzliche Restriktionen, die die Bounds von Integer-Variablen einschränken. Für jedes P_k wird das dazugehörige relaxierte LP (LP_k) gelöst. Ein Problem ist relaxiert, wenn für dieses gemischt-ganzzahlige Problem die Ganzzahligkeitsbedingungen wegfallen.

Für das relaxierte Unterproblem LP_k mit $k \in [1,n]$ gilt $z(P_k) \geq z(LP_k)$, d.h. die Lösung des relaxierten LPs LP_k stellt eine Untergrenze für die Lösung von P_k dar. Jede Integer-Lösung einer LP-Relaxation ist auch für das Unterproblem P_k gültig, und eine unzulässige Lösung von LP_k ist auch für P_k unzulässig. Demzufolge ergibt sich eine optimale IP-Lösung für P aus einer der besten Integer-Lösungen der Unterprobleme, welche durch die LP-Relaxation berechnet wurden.

Eine Integer-Lösung zeichnet sich dadurch aus, dass alle Integer-Variablen einen ganzzahligen Wert haben. Hat eine Integer-Variable keinen ganzzahligen Wert, so ist diese Variable fraktionell. Der Wert der fraktionellen Integer-Variablen x_j kann zerlegt werden in

$$\bar{x}_j = \lfloor \bar{x}_j \rfloor + f_j \text{ für } j \in J_I . \tag{2-1}$$

Der Term $\lfloor \bar{x}_j \rfloor$ stellt den ganzzahligen Teil von x_j dar, während f_j die Fraktionalität der Variablen angibt. Bei einer fraktionellen Variablen ist $0<f_j<1$. Je dichter f_j an 0,5 liegt, desto fraktioneller ist die Variable.

Hat ein Unterproblem P_k fraktionelle Integer-Variablen, dann wird aus dieser Menge an Variablen eine fraktionelle Variable x_j mit $j \in J_I$ gewählt. Durch das Branchen, d.h. das Zerlegen des Wertebereichs einer fraktionellen Variable in zwei Teile, unterteilt sich der Lösungsraum F_k in zwei neue disjunkte Lösungsräume F_{k+1} und F_{k+2}. Die dazugehörigen Probleme P_{k+1} und P_{k+2} unterscheiden sich von dem Problem P_k durch das Einfügen jeweils einer zusätzlichen Branching-Bedingung, die die relaxierte LP-Lösung LP_k von dem Problem P_k abschneidet:

$$P_{k+1} = P_k \cap \left\{ x_j \leq \lfloor \bar{x}_j \rfloor \right\} \text{ und } P_{k+2} = P_k \cap \left\{ x_j \geq \lceil \bar{x}_j \rceil \right\} \tag{2-2}$$

Eine bewährte Darstellung dieses Branch-and-Bound-Verfahrens ist eine Baumstruktur, wobei die einzelnen Unterprobleme die Knoten des Baums darstellen und das originale Problem P den Ausgangsknoten (Supernode).

Im schlechtesten Fall wird ein Problem mit z.B. n 01-Variablen in 2^n Knoten gelöst. Damit nicht der gesamte Baum abgearbeitet werden muss, um die Optimalität einer IP-Lösung zu beweisen, kann ein Teilproblem P_k durch folgende Gegebenheiten vom Baum abgetrennt werden:

- *Unzulässigkeit:* Durch die zusätzliche Restriktion in P_k wird der Wertebereich der Branchingvariablen x_j so eingeschränkt, dass das Problem P_k unzulässig wird. Alle nachfolgenden Knoten von P_k sind auch unzulässig, und somit müssen diese nicht betrachtet werden.
- *Integer-Lösung gefunden***:** Das relaxierte LP LP_k ist Integer und somit ist das dazugehörige Problem P_k optimal gelöst.
- *$z(LP_k) > zip$***:** Die Lösung des relaxierten LPs ist schlechter als die aktuelle beste Integer-Lösung zip. Da für die nachfolgenden Teilprobleme P_{k+1} und P_{k+2} $P_k = P_{k+1} \cup P_{k+2}$

und $P_{k+1} \cap P_{k+2} = \phi$ gilt und durch das Einfügen der Restriktionen die Lösung von LP_k für diese beiden Probleme nicht zulässig ist, ist $\min\{z(LP_{k+1}), z(LP_{k+2})\} \geq z(LP_k)$.

Demzufolge ist der Funktionswert eines Nachfolgeknotens nie besser als der Wert des Vorgängerknotens. Wenn der Funktionswert eines Knotens schlechter ist als die beste IP-Lösung, dann wird kein Nachfolgeknoten von diesem Knoten eine bessere IP-Lösung produzieren.

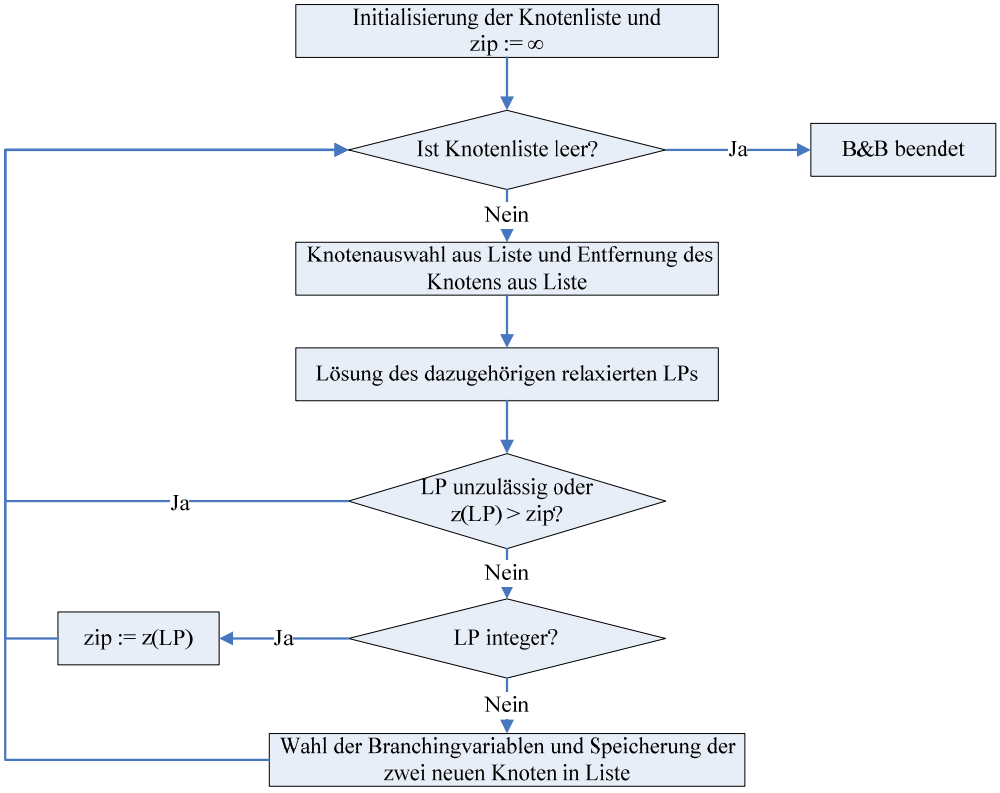

Abb. 2-2: Ablaufschema des Branch-and-Bound-Prozesses

Wie oben beschrieben, gibt es eine Obergrenze zip für alle Knoten. Diese Grenze berechnet sich aus der besten bisher gefundenen Integer-Lösung und ist für den ganzen Lösungsprozess global gültig. Wenn keine Integer-Lösung gefunden bzw. keine Obergrenze von außen angegeben wurde, dann ist die Grenze unendlich. Die Obergrenze der Funktionswerte der relaxierten LPs ist somit unbeschränkt. Des Weiteren gibt es eine globale Untergrenze für die Funktionswerte der Knoten. Diese Grenze wird mit dem Zielfunktionswert des relaxierten LPs nach dem Supernode Processing initialisiert. Dieser Wert berechnet sich während des Branch-and-Bound-Prozesses aus dem besten Funktionswert aller aktiven Knoten: $zlb = \min_k \{z(LP_k)\}$. Da jeder Nachfolgeknoten keinen besseren Funktionswert annehmen kann als der Vorgängerknoten, steigt der Wert von zlb monoton an. Immer wenn eine neue Integer-Lösung gefunden wird, sinkt der Wert von zip. Somit streben zlb und zip einander zu. Die Optimalität einer Integer-Lösung ist bewiesen, wenn der Gap zwischen zlb und zip hinreichend klein ist.

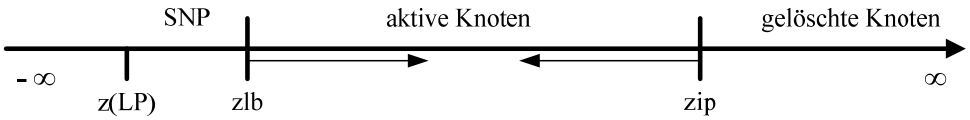

Abb. 2-3: Darstellung von zlb und zip

Bei dem Branch-and-Bound-Prozess gibt es zwei Auswahlentscheidungen, die getroffen werden müssen:

1. Welches Teilproblem, d.h. welcher Knoten, soll als nächstes untersucht werden?
2. Welche fraktionelle Variable soll an einem Knoten verzweigt werden?

Mit diesen beiden zentralen Fragestellungen und mit der Umsetzung des Branch-and-Bound-Prozesses in eine Software-Implementation beschäftigen sich die Kapitel 4 und 5.

Beispiel zur Verdeutlichung des Branch-and-Bound-Verfahrens:

$$\begin{aligned} \max \quad & 17x_1 + 12x_2 \\ & 10x_1 + 7x_2 \leq 40 \\ & x_1 + x_2 \leq 10 \\ & x_1, x_2 \geq 0 \text{ und integer} \end{aligned}$$

Eine optimale LP-Lösung ist $x_1 = 1.66$, $x_2 = 3.33$ mit $z(LP) = 68.33$

Zu diesem Problem wird folgender Baum erstellt:

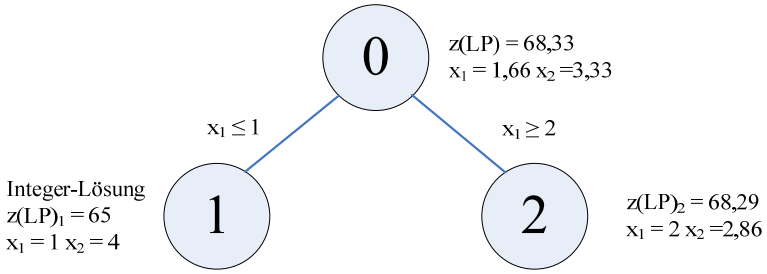

Abb. 2-4: Branch-and-Bound-Beispiel1

Am Ausgangsknoten 0 wird die fraktionelle Variable x_1 als Branchingvariable gewählt. Der Knoten 1 hat eine Integer-Lösung und wird demzufolge nicht weiterentwickelt. Der Knoten 2 ist fraktionell, zulässig und $z(LP)_2 > zip$. Demzufolge kann dieser Knoten weiterentwickelt werden. Die einzige fraktionelle Variable x_2 ist Branchingvariable.

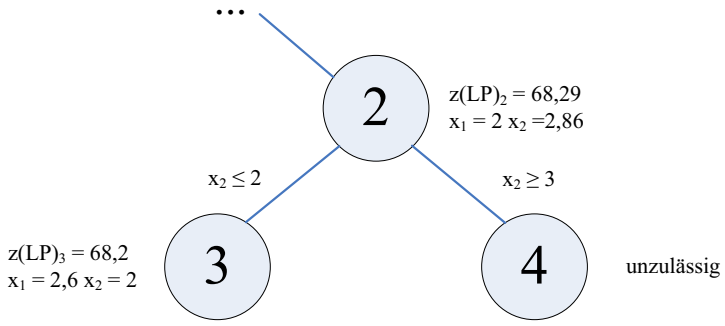

Abb. 2-5: Branch-and-Bound-Beispiel2

Durch das Branchen nach der Variablen x_2 entsteht der fraktionelle Knoten 3 und der unzulässige Knoten 4, der nicht weiterentwickelt wird. Bei dem Knoten 3 ist die einzige fraktionelle Variable x_1 Branchingvariable.

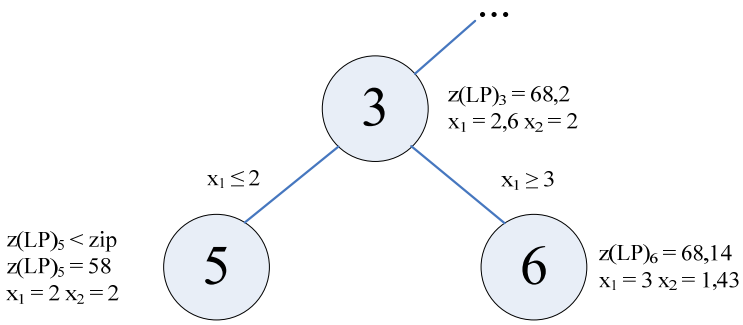

Abb. 2-6: Branch-and-Bound-Beispiel3

Der Lower Branch der Variable x_1 führt zu einer Integer-Lösung, die allerdings schlechter ist als zip. Demzufolge wird dieser Knoten nicht weiter betrachtet. Der andere entwickelte Knoten ist fraktionell mit x_2 als Branchingvariable.

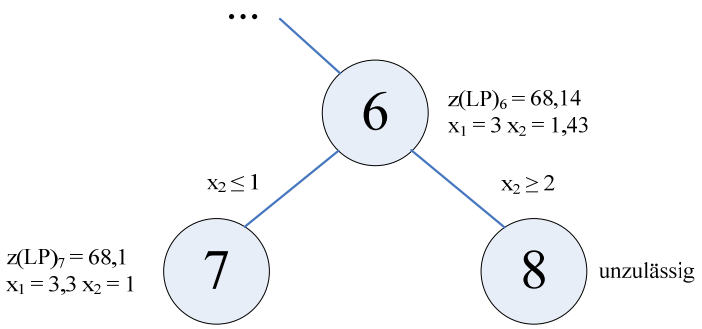

Abb. 2-7: Branch-and-Bound-Beispiel4

Das neue Teilproblem 7 ist fraktionell und muss weiterentwickelt werden. Nur die Variable x_1 kommt als Branchingvariable in Frage. Das andere neu entwickelte Teilproblem 8 ist unzulässig und wird nicht weiter betrachtet.

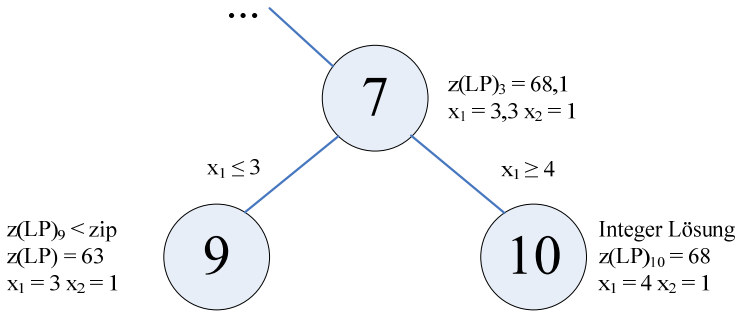

Abb. 2-8: Branch-and-Bound-Beispiel5

Der Lower Branch nach x_1 führt zu einem Knoten 9 mit einer Integer-Lösung schlechter als zip und zu einem Knoten 10 mit einer Integer-Lösung besser als zip (zip = $z(LP)_{10}$). Beide Knoten können nicht weiterentwickelt werden. Demzufolge sind alle Knoten des Baums abgearbeitet und der optimale Lösungswert ist zip = 68.

2.2 Branch-and-Cut-Verfahren

Dieses Kapitel gibt einen kurzen Überblick über das Branch-and-Cut-Verfahren. Zur Vertiefung dieser Materie ist folgende Literatur zu empfehlen [Pad01], [Mitch01], [Coretal99], [Maretal02].

Eine Schnittebene (Cut) ist eine gültige lineare Ungleichung, die nicht Bestandteil des aktuellen relaxierten Modells ist und die aktuelle LP-Lösung verletzt.

Beispiel:

$$\begin{aligned} \min \quad & 6x_1 - 5x_2 \\ & 3x_1 + x_2 \le 11 \\ & -x_1 + 2x_2 \le 5 \\ & x_1, x_2 \ge 0 \text{ und integer} \end{aligned}$$

Eine relaxierte LP-Lösung lautet: x_1=2,429; x_2=3,571

Ein möglicher Cut wäre $x_1+x_2 \le 5$. Dieser Cut verletzt die aktuelle LP-Lösung. Da noch alle zulässigen IP-Lösungen im relaxierten Lösungsraum enthalten sind, ist dieser Cut gültig.

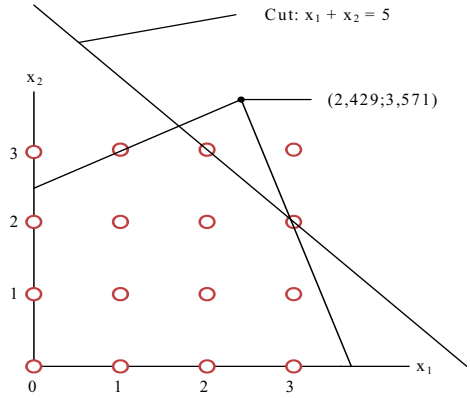

Abb. 2-9: Beispiel Schnittebenen

Das Branch-and-Cut-Verfahren ist eine Kombination aus Schnittebenen einfügen und dem Branch-and-Bound-Algorithmus (siehe Kapitel 2.1). Mit diesem Verfahren können wie beim Branch-and-Bound-Verfahren gemischt-ganzzahlige Modelle gelöst werden. Das Einfügen von Cuts verschärft die LP-Relaxation und das Branch-and-Bound-Verfahren sucht in dem eingeschränkten Lösungsraum nach einer optimalen IP-Lösung.

Die Schnittebenen können entweder am Ausgangsknoten, bei jedem Knoten oder nach einer bestimmten Anzahl von Knoten eingefügt werden.

Ein wichtiger Punkt des Branch-and-Cut-Verfahrens ist das Management der Cuts. Es muss festgelegt werden, wie oft, welche und wie viele Cuts abgeleitet werden und welche der schon generierten Cuts aktiv sein sollen. An einem Knoten können Cuts abgeleitet werden, welche diesen Knoten nicht verletzen, aber vielleicht die nachfolgenden Knoten. Diese Knoten werden in einem „Cut-Pool“ gespeichert und an bestimmten Knoten aktiviert bzw. deaktiviert.

Diese Einstellungen beeinflussen das Lösen der Modelle erheblich. So können z.B. sehr viele Cuts das Modell in weniger Knoten lösen. Durch die große Anzahl der Cuts ist aber die Anzahl der Zeilen und demzufolge der Nichtnullelemente gestiegen, so dass die relaxierten LPs dadurch langsamer gelöst werden. Werden weniger Cuts abgeleitet, dann werden die relaxierten LPs schneller gelöst, aber es werden wahrscheinlich mehr Knoten zum Lösen des Modells benötigt. Demzufolge sind die Einstellungen für die Cut-Generierung ein Trade-Off zwischen Lösungszeit der relaxierten LPs und Anzahl der zu lösenden Knoten.

Bei den Cuts wird zwischen globalen und lokalen Cuts unterschieden. Globale Cuts sind für den gesamten Suchbaum gültig, während lokale Cuts nur an dem Knoten, an dem sie generiert wurden, und an dessen Nachfolgeknoten gültig sind. Mit Hilfe eines „Liftings“ können lokale Cuts so verändert werden, dass diese für alle Teilprobleme global gültig sind [Pad01], [BaCeCo93], [Maretal02].

Das Ermitteln von Cuts während des Branch-and-Bounds erfolgt nachdem die LP-Relaxation gelöst wurde und das Ergebnis nicht integer, unzulässig oder schlechter als die bisher beste IP-Lösung ist. Anhand der relaxierten LP-Lösung werden Cuts ermittelt, die diese Lösung verletzen oder einfach nur in den „Cut-Pool“ aufgenommen werden können. Die eventuell generierten aktiven Cuts werden in das Problem eingefügt. Sind Cuts im „Cut-Pool“ enthalten, dann wird überprüft, ob von den Cuts welche aktiv werden können. Bei neu eingefügten Cuts wird die LP-Relaxation erneut gelöst. Es wird solange nach Cuts gesucht, bis keine mehr gefunden werden oder ein bestimmtes Limit an Cuts erreicht ist. Danach wird der Branch-and-Bound-Algorithmus fortgesetzt. Die folgende Abbildung Abb. 2-10 zeigt den Spezialfall des Branch-and-Cut-Algorithmus, bei dem aktive globale Cuts an einem Knoten abgeleitet und dem Problem zugefügt werden.

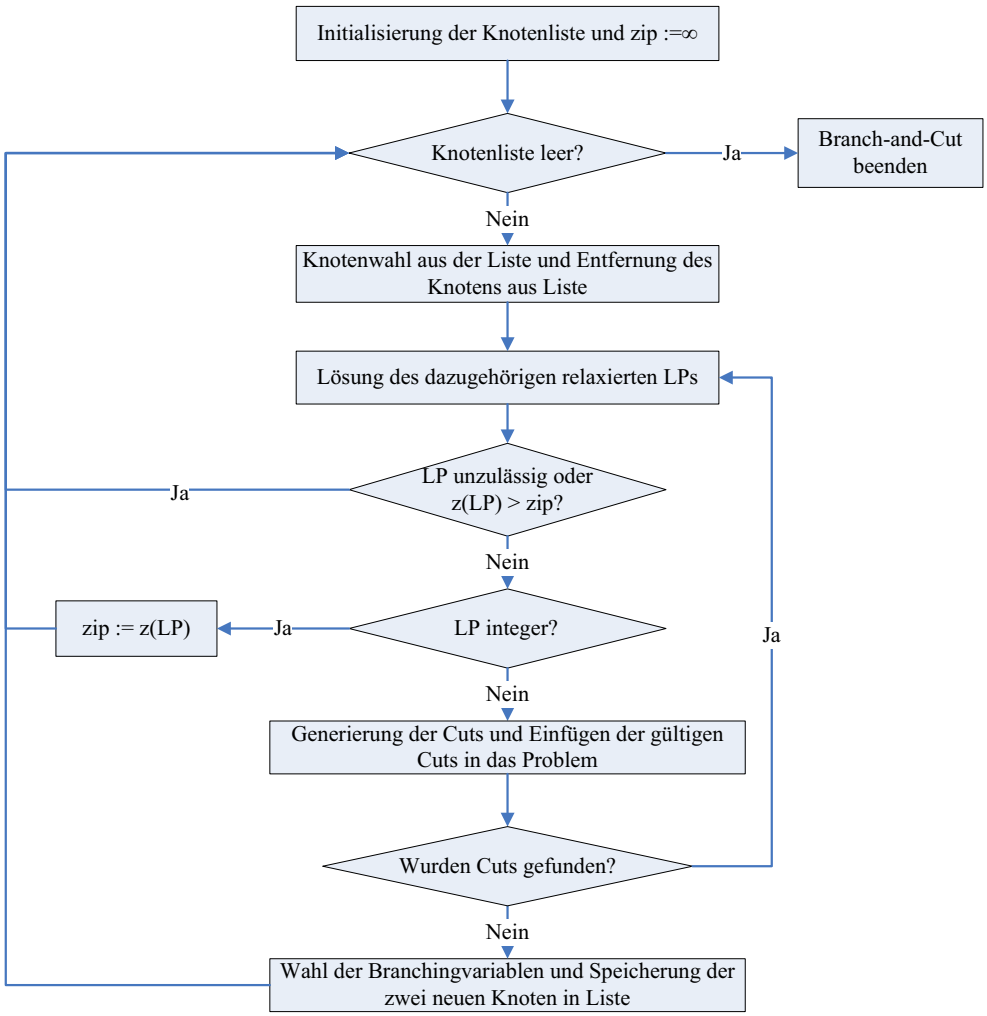

Abb. 2-10: Ablaufschema des Branch-and-Cut-Verfahrens

Es gibt eine Reihe von verschiedenen Cuts. Die folgende Auflistung mit dazugehörigen Referenzen stellt einige davon dar:

- Implication Cuts: [SuSz94]
- Clique Cuts: [JoPa83]
- Gomery Mixed-Integer Cuts: [Gom60]
- Mixed-integer Rounding Cuts: [NeWo88]
- Flow Cover Cuts: [Paetal85], [RoyWol86], [RoyWol87]
- Lifted Cover Cuts: [GuNeSa98]

2.3 Branch-and-Bound vs. Branch-and-Cut

Bei dem klassischen Branch-and-Bound-Verfahren werden außer den Einschränkungen der Variablengrenzen keine zusätzlichen Restriktionen in das Problem eingefügt. Alleine durch das Verzweigen wird der Lösungsraum des Modells eingeschränkt. Bei dem Branch-and-Cut-Verfahren werden zusätzliche Restriktionen, die die optimale LP-Lösung verletzen, in das Modell eingefügt, so dass der Lösungsraum durch Anhebung der globalen Lower Bound verkleinert wird. Das Suchen nach und die Generierung von Cuts kostet Zeit, auch wenn im Endeffekt kein verletzender Cut gefunden wird. Durch diese Cuts wird der Lö-

sungsraum zwar kleiner, aber es ist nicht sichergestellt, dass eine optimale Lösung schneller gefunden wird als beim Branch-and-Bound-Verfahren, denn durch das Einfügen der Cuts wird das Modell immer größer und dementsprechend schwerer zu lösen. Durch die Vor- und Nachteile des Branch-and-Cut-Verfahrens ist nicht gegeben, dass eines der beiden Verfahren besser ist als das andere. Bevor ein Modell gelöst wird, ist nicht erkennbar, welches der beiden Lösungsverfahren dieses schneller löst.

2.4 Strenge LP-Relaxierung

Das Konzept der strengen LP-Relaxierung geht ursprünglich auf E. L. Johnson in einem Forschungsprojekt von General Motors und IBM Research zur Lösung reiner 0-1-Probleme im Zeitraum 1978-1980 zurück [JoKoSu85].

Durch die strenge LP-Relaxierung wurde in den letzten Jahren das Lösen der IP-Modelle stark verbessert und beschleunigt. Es baut auf folgendem Prinzip auf [SuSz94]:

Das Problem P und das Problem P’ sind integer äquivalent, wenn sie die gleichen Integer-Lösungen besitzen. Das Problem P’ hat eine strengere LP-Relaxation LP’ als die LP-Relaxation LP von P, wenn der Lösungsraum F(LP’) von LP’ eine Teilmenge des Lösungsraums F(LP) von LP ist. Je kleiner der Lösungsraum eines Modells ist, desto wahrscheinlicher ist es, schneller eine optimale IP-Lösung zu finden. Demzufolge wird versucht ein Problem P in ein integer äquivalentes Problem P’ zu verändern, so dass der Lösungsraum des relaxierten Problems F(LP’) möglichst klein ist.

Der Lösungsraum eines IP-Problems wird durch die globale Upper Bound zip, die sich aus der besten Integer-Lösung ergibt, und der globalen Lower Bound z(LP), die sich aus der optimalen LP-Lösung ergibt, eingeschränkt. Die strenge LP-Relaxierung verbessert die globale Lower Bound z(LP’) des Problems P, so dass der Abstand zwischen den Bounds sich verringert.

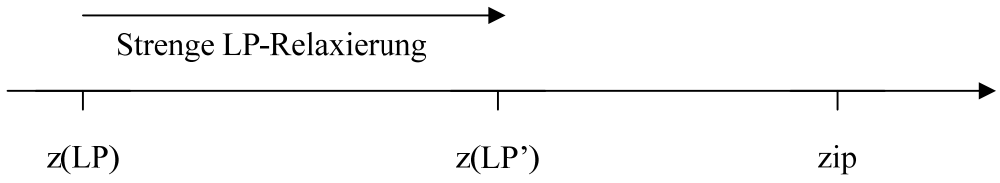

Abb. 2-11: Strenge LP-Relaxierung

Demzufolge ist ein Gütekennzeichen der strengen LP-Relaxierung die Größe von z(LP’).

Das Problem P kann durch eine automatische und eine manuelle strenge Relaxierung in P’ verändert werden.

2.4.1 Automatische Verschärfung der LP-Relaxierung

Bevor der Branch-and-Bound-Algorithmus startet, wird in der „Supernode“, d.h. im Ausgangsknoten, das Problem P in ein möglichst streng relaxiertes Problem P’ verändert. Dazu werden im so genannten Supernode-Processing (SNP) eine Reihe von Techniken angewandt, die das Ziel haben, möglichst viele Variablen zu fixieren, Bounds einzuschränken, mit Cuts den Lösungsraum zu verkleinern oder gar Unzulässigkeiten aufzudecken. Die

aufgelisteten Techniken werden u.a. in [Sa1994], [SuSz94], [RoyWol87], [CrJoPa83] und [HoPa91] detaillierter erklärt und vervollständigt:

- *Euklidean Reduction:* Wenn eine Restriktion nur Integer-Variablen mit rationalen Koeffizienten beinhaltet und für eine kleine Zahl e $a_{ij}' = a_{ij}*e$ integer ist, werden alle Koeffizienten a_{ij}' und die rechte Seite bl bzw. bu durch den größten gemeinsamen Teiler aller Koeffizienten a_{ij}' geteilt. Wenn bei einer Ungleichung die veränderten Restriktionsgrenzen bl' bzw. bu' nicht integer sind, dann kann bl' ab- und bu' aufgerundet werden. Sind bei einer Gleichung bl'=bu' kontinuierlich, dann kann für dieses Modell keine Integer-Lösung gefunden werden.
- *Zulässigkeitstests und gegebenenfalls Fixierungen:* Durch die Zulässigkeitstests wird überprüft, ob das Problem P eine zulässige Lösung haben kann. Dazu werden die minimale und maximale Zeilensumme der Restriktionen bestimmt und überprüft, ob die rechte Seite diese zulässt. Ist zum Beispiel bei einer ≤-Restriktion i mit der rechten Seite bu_i die minimale Zeilensumme L_i größer als bu_i, dann ist die Restriktion nicht zulässig, da keine IP-Lösung diese Restriktion erfüllen kann. Wenn allerdings die entsprechende Zeilensumme gleich der rechten Seite ist, z.B. $L_i=bu_i$, dann können alle Variablen in der Restriktion fixiert werden.
- *Koeffizientenreduktion:* Durch Setzen einer 01-Variablen auf Null oder Eins, wird überprüft, ob die Restriktion inaktiv wird. Ist dies der Fall, dann kann der Koeffizient der Variablen in der Restriktion reduziert werden, so dass keine Inaktivität mehr vorkommt.
- *Probing:* Beim Probing werden 01-Variablen nacheinander auf Null oder Eins gesetzt und überprüft, welche Variablen durch diese Setzung fixiert werden können. Das Probing kann eine Unzulässigkeit des Problems aufdecken und Implikationen herleiten.
- *Cuts:* U.a. die Implication- ([SuSz94]), die Clique- ([JoPa83], [NeWo88]), die Gomery Mixed-Integer ([Gom60]), die Mixed-Integer Rounding- ([NeWo88]), die Cover- ([Paetal85], [RoyWol86], [RoyWol87], [GuNeSa98]) und Disjunctive-Cuts ([Ba1979], [BaCeCo93], [BaCeCo96]) können im Supernode Processing abgeleitet werden und tragen erheblich zur Verschärfung der LP-Relaxation bei.

Ein Beispiel, welches den Einfluss des Preprocessings auf die Lösungszeit des Modells zeigt, befindet sich in [Suh00].

2.4.2 Manuelle Verschärfung der LP-Relaxierung

Die Formulierung eines Problems beeinflusst dessen Lösbarkeit. Demzufolge ist das Ziel, ein Problem so zu formulieren, dass es möglichst schnell zu lösen ist. Unterschiedliche Formulierungen führen zu gleich bleibenden IP-Lösungen. Sie haben aber einen unterschiedlich großen Lösungsraum, da der LP-Zielfunktionswert je nach Formulierung variiert.

Einer Optimierungssoftware wird ein Modell übergeben, welches je nach Können bzw. Aufwand des Entwicklers schwach oder scharf formuliert ist. In der Software ist es momentan noch nicht möglich, das gegebene Modell im Einzelnen zu analysieren und es je nach Art der Restriktionen komplett schärfer zu reformulieren.

Eine mögliche Variante der manuellen Verschärfung ist die Disaggregation, welche an dem folgenden Beispiel veranschaulicht wird.

Beispiel:

Die schwache aggregierte Formulierung $x_1+x_2\leq 2y$ mit der 01-Variablen y und den zwei positiven kontinuierlichen Variablen $x_1\geq 0$ und $x_2\leq 1$ kann in die strenge disaggregierte LP-Relaxation $x_1\leq y$ und $x_2\leq y$ überführt werden. Die disaggregierte Formulierung ist schärfer, da bei einer relaxierten LP-Lösung von z.B. $y'=0,7$ die beiden Variablen x_1 und x_2 durch $x_1\leq 0,7$ und $x_2\leq 0,7$ eingeschränkt werden können. Bei der aggregierten Formulierung sind die Wertebereiche der kontinuierlichen Variablen $x_1\leq 1,4$ und $x_2\leq 1$.