6.1 Branch and Bound

Bei *Branch-and-Bound*-Algorithmen wird die Menge der möglichen Lösungen in Form eines Baumes untersucht, wobei in jeder Stufe eine Aufteilung in zwei oder mehr Teilmengen erfolgt ('Branching'). Durch Berechnung von unteren Schranken ('Bounds') können dann Teile des Suchbaumes von der weiteren Berechnung ausgeschlossen werden.

Im Verfahren von Little, Murty, Sweeney und Karel werden die unteren Schranken durch *Reduktion* der Kostenmatrix $C = (c_{ij})$ (deren Diagonalelemente auf ∞ gesetzt werden) berechnet: Da bei einer Tour aus jeder Zeile und jeder Spalte genau ein Element der Kostenmatrix vorkommt, verringert sich durch Subtraktion einer Konstante q von allen Elementen einer Zeile oder einer Spalte die Länge aller Touren ebenfalls um q und die optimale Tour bleibt auch weiterhin optimal. Führt man dies für jede Zeile und Spalte so durch, daß man jeweils mindestens eine Null und keine negativen Werte erhält, dann ist die Summe der subtrahierten Werte eine untere Schranke.

Die Aufteilung des Lösungsmenge erfolgt jeweils in eine Teilmenge von Touren ohne eine Kante (i, j) und eine mit Touren, die diese enthalten. Da im ersten Fall (i, j) nicht vorkommt, kann c_{ij} den Wert ∞ erhalten und von der Zeile i und der Spalte j jeweils wieder eine Konstante subtrahiert werden, wodurch man eine neue Kostenmatrix und eine vergrößerte untere Schranke erhält.

Tritt (i, j) jedoch in der Tour auf, so lassen sich die Zeile i und die Spalte j entfernen, da man von i zu keinem anderen Knoten als j gehen kann, und zu j nur von i aus gelangt. Ferner muß c_{ji} auf unendlich gesetzt werden, da diese Kante ebenfalls nicht mehr vorkommen kann. Man hat also jetzt eine Matrix mit einer um eins verringerten Dimension, auf die man das gleiche Verfahren der Reduktion und Aufspaltung anwendet. Auf diese Weise gelangt man schließlich bis zu 2×2 Matrizen, wodurch eine Tour festgelegt ist, deren Länge l man berechnen kann. Aus dem Suchbaum lassen sich nun alle die Bereiche ausschließen, für die man untere Schranken größer l erhalten hat, während man für die verbliebenen Matrizen den Algorithmus wiederholt.

Es bleibt noch zu klären, wie man die Kanten (i, j) auswählt, nach denen die Lösungsmenge aufgeteilt wird: Da das Ziel ist, möglichst viele 'große' Matrizen von der weiteren Rechnung auszuschließen, wird man (i, j) so bestimmen, daß die untere Schranke für die Teilmenge ohne (i, j) sich am stärksten vergrößert, d. h. man wählt die Null, deren Ersetzung durch ∞ die größte Subtraktion von der entsprechenden Zeile und Spalte erlaubt.

Bei Programm 6.1 handelt es sich um eine an die TSP-Unit angepaßte Version eines Programms aus *M. M. Syslo: Discrete Optimization Algorithms (Prentice-Hall, 1983).*