ILP,LP-Relaxierungen, Cutting planes und Branch and Bound

Das Matching-Polytop

Sei ein Graph $G = (V, E)$ mit nicht-negativen Kantengewichten w_e für alle Kanten e gegeben. Wir möchten in G ein Matching finden, das unter allen Matchings maximales Gewicht hat. Falls G ein bipartiter Graph ist, kennen wir einen Algorithmus, der ein Matching maximalen Gewichts berechnet. Dieser Algorithmus funktioniert jedoch nicht, wenn G ein beliebiger Graph ist. Wir formulieren unser Problem deshalb zuerst einmal als *ganzzahliges lineares Programm (ILP: integer linear program)*:

Wir führen für jede Kante e eine Variable x_e ein, die den Wert 1 annehmen soll, wenn die Kante e im Matching maximalen Gewichts enthalten ist. Wenn e nicht in diesem Matching enthalten ist, soll die Variable x_e den Wert 0 annehmen. Nun müssen wir noch die Bedingung beschreiben, dass jeder Knoten v von höchstens einer Matchingkante überdeckt wird. Dazu definieren wir $\delta(V) := \{e \in E : v \in e\}$ als die Menge aller Kanten, die mit dem Knoten v benachbart sind, und erhalten für jeden Knoten $v \in V$ die Bedingung $\sum_{e\in\delta(v)} x_e \leq 1$. Insgesamt erhalten wir also folgendes ILP:

$$\begin{array}{lll} \text{maximize} & \sum_{e\in E} w_e x_e & \\ \text{subject to} & \sum_{e\in\delta(v)} x_e \leq 1 & \text{für alle } v \in V, \\ & x_e \in \{0,1\} & \text{für alle } e \in E. \end{array}$$

Relaxierung. Das obige Programm ist *kein* lineares Programm, die Bedingung $x_e \in \{0, 1\}$ führt nämlich dazu, dass wir nur ganz bestimmte Lösungen zulassen, nämlich die ganzzahligen.

Falls G ein bipartiter Graph ist, können wir die Bedingung $x_e \in \{0, 1\}$ durch $x_e \geq 0$ ersetzen und erhalten ein lineares Programm, das wir mit dem Simplex-Algorithmus lösen können. Das so entstandene lineare Programm nennt man *Relaxierung* des ursprünglichen Problems. Für den bipartiten Fall kann man leicht zeigen, dass alle Lösungen, die wir erhalten, trotzdem ganzzahlig sind. Im allgemeinen Fall jedoch, können die Nebenbedingungen $x_e \in \{0, 1\}$ nicht einfach durch $x_e \geq 0$ ersetzt werden, wie man leicht an folgendem Beispiel sieht: Nehmen wir als Graph ein Dreieck, indem jede Kante das Gewicht 1 hat. Ein Matching maximalen Gewichts hat dann Gewicht 1, weil jedes Matching in dem Dreieck aus genau einer Kante besteht. Die optimale Lösung des relaxierten linearen Programms hat hingegen den Wert $3/2$, weil alle Kantenvariablen auf den Wert $1/2$ gesetzt werden können.

Ein lineares Programm für Matching. Wie können wir nun unser Matching-Problem mit linearer Programmierung lösen? Gibt es überhaupt ein lineares Programm, das das Matching-Problem in allgemeinen Graphen beschreibt? Die Antwort ist "ja". Durch Hinzufügen von neuen Ungleichungen können wir die LP-Relaxierung so verändern, dass alle Lösungen des neuen LPs ganzzahlig sind. Wir verwenden dazu die folgende Beobachtung:

Sei $A \subseteq V$ eine Menge einer ungeraden Anzahl Knoten und sei $E_A \subseteq E$ die Menge aller Kanten, die beide Endpunkte in A haben. Dann enthält jedes Matching höchstens $(|A| - 1)/2$

Kanten aus E_A. Diese Beobachtung führt zu den folgenden linearen Ungleichungen:

$$\sum_{e \in E_A} x_e \leq \frac{|A|-1}{2} \quad \text{für alle } A \subseteq V, |A| \text{ ungerade.}$$

Edmonds hat gezeigt, dass durch Hinzufügen dieser Gleichungen zu **??** ein lineares Programm entsteht, dessen Lösungen genau die Matchings maximalen Gewichts sind.

Das TSP-Polytop

Im Falle des Matching-Problems sind wir in der glücklichen Situation, dass wir ein lineares Programm angeben können, dessen Lösungen den Matchings entsprechen. Wie sieht die Situation für das TSP aus?

Beschreiben wir das TSP zuerst als ILP. Auch hier führen wir wieder für jede Kante e eine Variable x_e ein, die 1 sein soll, wenn die Kante e in der optimalen Tour enthalten ist und sonst 0.

In einer gültigen Tour darf jeder Knoten nur genau einmal besucht werden, insgesamt gibt es somit zwei Kanten innerhalb der Tour, die mit dem Knoten verbunden sind (die Kante, über die man zum Knoten kommt, und die Kante, über die man ihn wieder verlässt); wir erhalten also

$$\sum_{e: v \in e} x_e = 2 \quad \text{für alle } v \in V.$$

Diese linearen Ungleichungen allein reichen jedoch nicht: eine Menge von disjunkten Kreisen wäre immer noch eine gültige Lösung. Wir müssen lineare Ungleichungen finden, die solche "Subtouren" verbieten, d.h. wir müssen verbieten, dass es innerhalb einer echten Teilmenge $\emptyset \neq A \subsetneq V$ von Knoten eine Tour der Länge $|A|$ gibt:

$$\sum_{e \in E_A} x_e \leq |A| - 1 \quad \text{für alle } \emptyset \neq A \subsetneq V$$

Insgesamt wird das TSP also durch folgendes ILP beschrieben:

$$\begin{array}{rll} \text{maximize} & \sum_{e \in E} w_e x_e & \\ \text{subject to} & \sum_{e: v \in e} x_e = 2 & \text{für alle } v \in V, \\ & \sum_{e \in E_A} x_e \leq |A| - 1 & \text{für alle } \emptyset \neq A \subsetneq V, \\ & x_e \in \{0, 1\} & \text{für alle } e \in E. \end{array}$$

Natürlich würden wir auch das TSP gerne mittels linearer Programmierung lösen. Wenn wir aber die Bedingung $x_e \in \{0, 1\}$ durch $0 \leq x_e \leq 1$ ersetzen, um die LP-Relaxierung zu erhalten, erhalten wir aber wieder wie beim Matching-Problem ungewünschte nicht-ganzzahlige Lösungen. Beim Matching-Problem konnten wir durch Hinzufügen neuer Ungleichungen trotzdem ein lineares Programm erhalten, was unser Problem genau beschreibt. Für das TSP sind solche Ungleichung jedoch *nicht bekannt*! Wie können wir TSP also lösen?

Cutting planes

Die Grundidee der Cutting-plane-Verfahren ist einfach. Wenn die optimale Lösung x^* der LP-Relaxierung eines ILP nicht ganzzahlig ist, kann x^* durch eine Hyperebene von den ganzzahligen Lösungen des ILPs getrennt werden, d.h.es gibt einen Vektor d und eine Zahl f, so dass

$$d^\top x^* > f \quad \text{und} \quad d^\top x \leq f \text{ für alle ganzzahligen Lösungen } x.$$

Diese neue Ungleichung kann nun zur LP-Relaxierung hinzugefügt werden und wir beginnen von neuem. Es kann gezeigt werden, dass dieses Verfahren terminiert. Dass Problem ist, dass es im allgemeinen schwierig ist, solch eine trennende Hyperebene zu finden. Eine Klasse von solch trennenden Hyperebenen sind die sogenannten *Gomory-Cuts*, für viele spezielle Problemklassen sind jedoch bessere "Cuts" bekannt, die die spezielle Problemstruktur verwenden, um mit einem "cut" möglichst viele nicht-ganzzahlige Lösungen wegzuschneiden.

Branch and bound

Bei der Lösung eines ILPs mit Hilfe von Cutting planes beschränkt man sich häufig auf ganz bestimmte Klassen von trennenden Hyperebenen. Wenn mit diesen Hyperebenen die Lösung nicht mehr weiter verbessert werden kann, verwendet man Suchverfahren um die aktuelle Lösung weiter zu verbessern.

Der Ansatz des "Branch and bound" löst ein ILP, indem alle Lösungen aufgezählt werden, und die jeweils beste gefundene Lösung zurückgegeben wird. Die Aufzählung der Lösungen geschieht jedoch so, dass möglichst viele Lösungen gar nicht erst betrachtet werden müssen.

Gegeben das ILP

$$\begin{array}{rl} \text{maximize} & c^\top x \\ \text{subject to} & Ax \leq b, \\ & x \geq 0, \\ & x \in \mathbb{Z}^n, \end{array}$$

lösen wir zunächst die LP-Relaxierung, indem wir die Nebenbedingung $x \in \mathbb{Z}^n$ einfach fallen lassen. Wenn die Lösung x^* der LP-Relaxierung ganzzahlig ist, sind wir fertig. Ansonsten gibt es eine Koordinate i, so dass $x_i^* \notin \mathbb{Z}$. In diesem Fall, teilen wir das ursprüngliche Problem in zwei Teilprobleme auf und geben die bessere der beiden Lösungen zurück:

$$\begin{array}{rl} \text{maximize} & c^\top x \\ \text{subject to} & Ax \leq b, \\ & x_i \leq \lfloor x_i^* \rfloor, \\ & x \geq 0, \\ & x \in \mathbb{Z}^n. \end{array} \qquad \begin{array}{rl} \text{maximize} & c^\top x \\ \text{subject to} & Ax \leq b, \\ & x_i \geq \lfloor x_i^* \rfloor + 1, \\ & x \geq 0, \\ & x \in \mathbb{Z}^n. \end{array}$$

Die wichtige Beobachtung ist, dass jede ganzzahlige Lösung des ursprünglichen Problems in genau einem der beiden Teilprobleme enthalten ist—wir können also keine Lösung übersehen. Die Aufteilung des ursprünglichen Problems in Teilprobleme führt zu einem Berechnungsbaum, dessen Knoten jeweils einem ILP entsprechen und dessen Kinder die beiden im Aufteilungsschritt entstandenen ILPs sind.

Der wirklich wichtige Schritt beim "Branch and bound" ist der folgende: Während der Berechnung des Baumes speichern wir in der Variable z_I den besten Wert einer ganzzahligen

Lösung. Wenn nun innerhalb eines Knotens v die optimale Lösung z_{LP} der LP-Relaxierung berechnet wird, können wir den Wert z_{LP} mit z_I vergleichen. Da im Fall $z_{LP} < z_I$ die optimalen Lösungen der ILPs aller Nachfahren von v immer kleiner sind als x_{LP}, müssen die Kinder von v gar nicht erst berechnet werden.

Die Hoffnung ist nun, dass der Berechnungsbaum klein bleibt, weil für viele Knoten v die Kinder gar nicht erst berechnet werden müssen. Die Effizienz des Branch and bound wird natürlich durch die beiden folgenden Faktoren erheblich beeinflusst:

- die Reihenfolge, in welcher die Knoten des Baumes bearbeitet werden,
- die Auswahl des Index i, für den $x_i^* \notin \mathbb{Z}$.