Technische Universität München
Zentrum Mathematik

Diskrete Optimierung: Fallstudien aus der Praxis

Barbara Wilhelm | Michael Ritter

Feasibility Pump

Feasibility Pump ist ein Algorithmus, der zulässige ganzzahlige Lösungen für ILPs generiert. Der Algorithmus eignet sich damit gut für die Startphase eines Branch and Bound-Ansatzes, um eine erste ganzzahlige Lösung zu gewinnen. (Erst mit einer Lösung hat man ja die Möglichkeit, Äste des Branch and Bound-Baums abzuschneiden.) Wir erläutern hier kurz die Idee von Feasibility Pump.

1 Problemstellung

Wir gehen zunächst von einem 0-1-ILP in folgender Form aus:

$$\begin{aligned} \max\ & c^T x \\ & Ax \leq b \\ & x \in \{0,1\}^n \end{aligned}$$

Wir bezeichnen den zulässigen Bereich der LP-Relaxation mit P:

$$P := \{x \in \mathbb{R}^n : Ax \leq b,\ 0 \leq x \leq 1\}$$

Für einen Vektor $x \in \mathbb{R}^n$ sei $[x] \in \{0,1\}^n$ der komponentenweise gerundete Vektor. Außerdem benötigen wir die L_1-Norm zur Abstandsmessung, die als

$$\Delta(x,y) := \|x-y\|_1 := \sum_{i=1}^{n} |x_i - y_i|$$

definiert ist.

2 Grundidee

1. Bestimme ein zulässiges $x^* \in P$ und den zugehörigen gerundeten Vektor $\tilde{x} := [x^*]$.
2. Falls $\tilde{x} \in P$ gilt, haben wir eine ganzzahlige, zulässige Lösung $\tilde{x}$ gefunden.
3. Bestimme einen neuen Punkt $x^* \in P$, der möglichst nahe an $\tilde{x}$ liegt, d. h., für gegebenes $\tilde{x}$ löse das Problem $x^* := \operatorname{argmin}\{\Delta(x,\tilde{x}) : x \in P\}$. Wenn x^* ganzzahlig ist, so haben wir mit x^* eine ganzzahlige Lösung gefunden. Sonst ersetze $\tilde{x}$ durch den neuen gerundeten Vektor $\tilde{x} := [x^*]$ und gehe zu 2.
4. Wiederhole diese Schritte, bis eine ganzzahlige Lösung gefunden ist oder bis ein anderes Abbruchkriterium erreicht ist, z. B. ausreichend lange erfolglose Iteration.

3 Algorithmus

Beim beschriebenen Vorgehen ergeben sich zwei Probleme: Zum einen kann es sehr leicht zum Zykeln kommen, wenn der gleiche gerundete Vektor ein zweites Mal im Lauf des Algorithmus berechnet wird. Zum anderen sollte Konvergenz des Algorithmus sichergestellt werden, d. h., man muss dafür sorgen, dass der $\Delta(x^*, \tilde{x})$-Wert mit jeder Iteration geringer wird. Beides versucht man durch *reverse rounding* zu garantieren, d. h., man rundet bei Bedarf nicht wie gewöhnlich auf die nächstliegende ganze Zahl, sondern nimmt stattdessen die weiter entfernte der beiden Alternativen 0 und 1. Im Detail sieht *feasibility pump* wie im Algorithmus 1 aus.

Algorithm 1: Feasibility Pump

Input: $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, $c \in \mathbb{Z}^n$, $MaxIter \in \mathbb{N}$ (maximale Anzahl Iterationen), $minFlip \in \mathbb{N}$ (minimale Anzahl Variablen, die jeweils „geflippt“ werden)

Output: Eine ganzzahlige zulässige Lösung $x^* \in P \cap \{0,1\}^n = \{x : Ax \leq b,\ 0 \leq x \leq 1\} \cap \{0,1\}^m$ oder die Meldung, dass keine solche Lösung gefunden wurde.

Initialisiere: $x^* := \operatorname{argmax}\left\{c^T x : x \in P\right\}$, $\tilde{x} := [x]$, $nIter := 0$
while $nIter < MaxIter$ **do**
  $nIter := nIter + 1$
  if $\tilde{x} \in P$ **then**
    Abbruch mit Lösung $\tilde{x}$
  end
  $x^* := \operatorname{argmin}\{\Delta(x, \tilde{x}) : x \in P\}$
  if x^* *ganzzahlig* **then**
    Abbruch mit Lösung x^*
  end
  Berechne „flip score“ $\sigma_j := |x^*_j - \tilde{x}_j|$ für alle $j \in \{1, \ldots, n\}$
  Setze $J := \{j \in \{1, \ldots, n\} : \sigma_j > 1/2\}$
  Setze $\tilde{x}_j := 1 - \tilde{x}_j$ für alle $j \in J$
  if $|J| < minFlip$ **then**
    Setze $\tilde{x}_j := 1 - \tilde{x}_j$ für die $minFlip - |J|$ Indizes $j \notin J$, die den höchsten flip score σ_j haben
  end
end
if $\tilde{x} \in P$ **then**
  Abbruch mit Lösung $\tilde{x}$
else
  Meldung: „erfolglos“
end

Im Vergleich zur Grundidee kommen also noch zwei Elemente hinzu: Die Iteration wird nach einer vorgegebenen Anzahl Versuchen abgebrochen ($MaxIter$), auch wenn bis dahin kein Erfolg erzielt wurde. Außerdem wird das Runden über einen „flip“ realisiert. Dabei wird zunächst überprüft, an wie vielen Positionen sich die neue ganzzahlige Lösung von der bisherigen ganzzahligen Lösung unterscheiden würde, wenn man alle Variablenwerte von x^* standardmäßig

runden würde. Ist dieser Wert geringer als $minFlip$, so werden durch *reverse rounding* zusätzliche Variablen in der neuen Lösung auf andere Werte gesetzt als in der bisherigen ganzzahligen Lösung. Auf diese Weise soll das Zykeln des Verfahrens vermieden werden.

Das Verfahren lässt sich auf beliebige ganzzahlige lineare Programme verallgemeinern (nicht nur auf 0-1-Programme).

4 Fragen

In den folgenden Fragen befassen Sie sich eingehender mit den eben erarbeiteten Methoden. Diskutieren Sie die Fragen in der Gruppe und gehen Sie in Ihrer Präsentation der Verfahren kurz auf Ihre Ergebnisse ein.

1. Im Laufe des Algorithmus ist das Optimierungsproblem

   $$x^* := \operatorname{argmin} \{\Delta(x, \tilde{x}) : x \in P\}$$

   zu lösen. Um was für ein Optimierungsproblem handelt es sich (linear, nichtlinear, kombinatorisch, etc.)? Welchen Algorithmus schlagen Sie vor, um das Problem zu lösen und warum?

2. Die vorgestellte Methode lässt sich sehr ähnlich auch für allgemeine ganzzahlige lineare Programme einsetzen (nicht nur für 0-1-ILPs). An welchen Stellen des Algorithmus sind Änderungen notwendig, wenn Variablen $x \in \mathbb{N}_0^n$ berücksichtigt werden sollen? Welche Änderungen schlagen Sie jeweils vor?