1 Approximative Lösungen NP-schwerer Optimierungsprobleme

"Kann man NP-schwere Optimierungsprobleme effizient beliebig genau approximieren?" Diese Frage wird neben anderen vom PCP-Theorem beantwortet. Zunächst soll aber an einem Beispiel genauer erläutert werden, was Approximation eigentlich ist.

Definition ($val(\varphi)$): Der *Wert* einer Formel φ in 3KNF mit n Klauseln ist definiert als die größte (rationale) Zahl $q \in [0,1]$, für die gilt: es existiert eine Belegung der Variablen von φ, die m Klauseln erfüllt, und $q = \frac{m}{n}$. Wir schreiben dafür $val(\varphi)$.
Offensichtlich gilt: φ ist erfüllbar $\Leftrightarrow val(\varphi) = 1$.

Definition (ρ-Approximation für MAX-3SAT): Das Optimierungsproblem MAX-3SAT besteht darin, zu einer gegebenen logischen Formel φ eine Belegung zu finden, die die Anzahl erfüllter Klauseln maximiert, also gerade $val(\varphi)\cdot n$ Klauseln wahr macht, wenn n die Anzahl der Klauseln in φ ist. MAX-3SAT ist NP-schwer, da 3SAT NP-vollständig ist.

Sei $\rho \in [0,1]$. Ein Algorithmus A heißt *ρ-Approximations-Algorithmus* für MAX-3SAT, wenn A zu jeder beliebigen eingegebenen Formel φ in 3KNF eine Belegung ausgibt, die mindestens $\rho \cdot val(\varphi) \cdot n$ Klauseln in φ erfüllt.

Bsp.: $\frac{1}{2}$-Approximation für MAX-3SAT: Zu einer gegebenen Formel φ in 3KNF mit n Klauseln und m Variablen lässt sich mit Hilfe des folgenden Greedy-Algorithmus eine $\frac{1}{2}$-Approximation für MAX-3SAT in polynomieller Zeit berechnen:
Allen Variablen von 1 bis m wird nacheinander ein Wert zugewiesen. Variable i erhält den Wert, der mindestens die Hälfte der Klauseln wahr macht, in denen diese Variable vorkommt. Danach werden alle Klauseln gestrichen, die erfüllt sind, und es wird mit der nächsten Variable fortgefahren.

Die erzeugte Belegung erfüllt mindestens die Hälfte aller Klauseln also auch mindestens halb so viele Klauseln wie maximal möglich.

2 Zwei Sichtweisen des PCP-Theorems

2.1 Die erste Sichtweise: lokal überprüfbare Beweise

Definition (PCP-System): Seien L eine Sprache und $q, r : \mathbb{N} \to \mathbb{N}$ zwei Funktionen. L hat ein $(r(n), q(n))$-*PCP-System*, wenn es einen polynomialzeitbeschränkten probabilistischen Algorithmus V gibt, der folgende Bedingungen erfüllt:

1. Effizienz: Bei Eingabe eines Wortes $x \in \{0,1\}^n$ und wahlfreiem Zugriff auf ein Wort $\pi \in \{0,1\}^*$ mit Länge höchstens $q(n)\cdot 2^{r(n)}$ (welches im Folgenden *Beweis* genannt wird) nutzt V höchstens $r(n)$ Münzwürfe und stellt höchstens $q(n)$ nicht-adaptive Anfragen an Zeichen in π. Dann gibt er 1 oder 0 aus. Im Folgenden bezeichne $V^\pi(x)$ die Zufallsvariable, die die Ausgabe von V bei Eingabe x und wahlfreiem Zugriff auf π repräsentiert.
2. Vollständigkeit: Wenn $x \in L$, dann existiert ein Beweis $\pi \in \{0,1\}^*$, sodass $\Pr[V^\pi(x) = 1] = 1$. Dieses Wort π wird als *korrekter Beweis* für x bezeichnet.
3. Korrektheit: Wenn $x \notin L$, dann ist $\Pr[V^\pi(x) = 1] \leq \frac{1}{2}$ für jedes $\pi \in \{0,1\}^*$.

Man definiert neue Sprachklassen: Eine Sprache L ist in $\mathrm{PCP}(r(n), q(n))$, wenn Konstanten $c, d > 0$ existieren, sodass L ein $(c\cdot r(n), d\cdot q(n))$-PCP-System hat.

PCP-Theorem (erste Sichtweise): NP = PCP($\log n$, 1).

Anmerkung: Allgemein gilt PCP($r(n), q(n)$) $\subseteq$ NTIME($2^{O(r(n))}q(n)$).
Eine Richtung des PCP-Theorems ergibt sich damit als Spezialfall:
PCP($\log n$, 1) $\subseteq$ NTIME($2^{O(\log n)}$) = NP.

Beispiel für ein PCP-System: Um eine bessere Vorstellung davon zu bekommen, wie ein PCP-System funktioniert, wird gezeigt, dass die Sprache GNI aller Paare nicht-isomorpher Graphen in PCP($poly(n), 1$) ist.

Die Eingabe seien zwei beschriftete Graphen G_0 und G_1 mit jeweils n Knoten. Der Beweis π enthält zu jedem beschrifteten Graphen H mit n Knoten ein Bit. Es ist $\pi[H] = 0$, falls H isomorph zu G_0 und $\pi[H] = 1$, falls H isomorph zu G_1. Ist H isomorph zu beiden Graphen oder zu keinem von beiden, ist $\pi[H]$ beliebig.

Das System erzeugt eine Zufallszahl $b \in \{0,1\}$ und eine zufällige Permutation der Zahlen von 1 bis n. Dann wendet es diese Permutation auf G_b an und erhält einen isomorphen Graphen H. Schließlich akzeptiert es gdw. $\pi[H] = b$.

Die drei Bedingungen sind erfüllt:

1. Für b wird nur ein Münzwurf benötigt und die Permutation lässt sich mit einer polynomiellen Anzahl an Münzwürfen erzeugen (zum Beispiel mit $O(n \log n)$). Es wird genau eine Anfrage an den Beweis gestellt.
2. Wenn G_0 und G_1 nicht isomorph sind, dann wird das System bei korrekt konstruiertem π in $\pi[H]$ stets das erwartete b vorfinden und akzeptieren.
3. Sind G_0 und G_1 dagegen isomporph, kann die Wahrscheinlichkeit, dass in $\pi[H]$ das erwartete b steht, für kein π größer als $\frac{1}{2}$ sein.

2.2 Die zweite Sichtweise: Approximationsschwere

PCP-Theorem (zweite Sichtweise): Es gibt ein $\rho < 1$, sodass für alle $L \in$ NP eine in polynomieller Zeit berechenbare Funktion f existiert, die Wörter auf (Repräsentationen von) Formeln in 3KNF abbildet, sodass gilt:

1. $x \in L \Rightarrow val(f(x)) = 1$
2. $x \notin L \Rightarrow val(f(x)) < \rho$

Korollar: Es existiert ein $\rho < 1$, sodass aus der Existenz eines polynomialzeitbeschränkten ρ-Approximations-Algorithmus für MAX-3SAT folgt, dass P = NP.

2.3 Äquivalenz der beiden Sichtweisen

Zwei weitere Definitionen und ein Theorem dienen als Hilfsmittel für den Äquivalenzbeweis.

Definition (CSP) : Sei $q \in \mathbb{N}$. Eine qCSP-Instanz φ ist eine Menge von Funktionen $\varphi_1, \ldots, \varphi_m$ (diese nennt man *constraints*) von $\{0,1\}^n$ nach $\{0,1\}$, wobei für jedes $i \in \{1, \ldots, m\}$ Indizes $j_1, \ldots, j_q \in \{1, \ldots, n\}$ und eine Funktion f von $\{0,1\}^q$ nach $\{0,1\}$ existieren, sodass für alle $u \in \{0,1\}^n$ gilt: $\varphi(u) = f(u_{j_1}, \ldots, u_{j_q})$.

Eine *Belegung* $u \in \{0,1\}^n$ *erfüllt* constraint φ_i, wenn $\varphi_i(u) = 1$. Der Anteil der durch die Belegung u erfüllten constraints ist $(\sum_{i=1}^{m} \varphi_i(u))/m$ und $val(\varphi)$ bezeichne das Maximum dieses Wertes für alle $u \in \{0,1\}^n$. Wenn $val(\varphi) = 1$, heißt φ *erfüllbar*.

Definition (Gap CSP): Zu $q \in \mathbb{N}$ und $\rho \in (0,1)$ sei ρ-GAPqCSP das Problem, zu einer gegebenen qCSP-Instanz φ zu bestimmen, ob $val(\varphi) = 1$, dann heißt φ JA-Instanz von ρ-GAPqCSP, oder $val(\varphi) < \rho$, dann heißt φ NEIN-Instanz.

Das Problem ρ-GAPqCSP ist NP-schwer für jede Sprache $L \in$ NP, wenn eine in polynomieller Zeit berechenbare Funktion f existiert, die Wörter auf (Repräsentationen von) qCSP-Instanzen abbildet, sodass gilt:

1. Vollständigkeit: $x \in L \Rightarrow val(f(x)) = 1$
2. Korrektheit: $x \notin L \Rightarrow val(f(x)) < \rho$

Theorem 3: Es existieren $q \in \mathbb{N}$ und $\rho \in (0,1)$, sodass ρ-GAPqCSP NP-schwer ist.

Im Folgenden wird gezeigt, dass Theorem 3 sowohl zu Theorem 1 als auch zu Theorem 2 äquivalent ist. Damit ist dann auch gezeigt, dass Theorem 1 und 2 äquivalent sind. Der Beweis, dass überhaupt eines dieser Theoreme gilt, ist sehr kompliziert und wird an dieser Stelle nicht angegeben. Siehe dazu das als Quelle angegebene Buch.

Theorem 1 impliziert Theorem 3
Angenommen NP $\subseteq PCP(\log n, 1)$. Es wird 3SAT reduziert auf $\frac{1}{2}$-GAPqCSP für eine Konstante q. Gemäß der Annahme gibt es ein PCP-System V für 3SAT, das $c \log n$ Münzen wirft (für eine Konstante c) und eine konstante Anzahl, welche mit q bezeichnet werde, an Anfragen stellt. Zu jeder Eingabe x und jeder Folge von Wurfergebnissen $r \in \{0,1\}^{c \log n}$ bezeichne $V_{x,r}$ die Funktion, die bei Eingabe eines Beweises π 1 ausgibt, wenn V den Beweis π bei Eingabe x unter Nutzung der Münzwürfe r akzeptiert. Da jedes $V_{x,r}$ von höchstens q Stellen abhängt, ist für jedes $x \in \{0,1\}^n$ die Menge $\varphi = \{V_{x,r}\}_{r \in \{0,1\}^{c \log n}}$ eine qCSP-Instanz polynomieller Größe. Da V in polynomieller Zeit läuft, kann auch die Transformation von x nach φ in polynomieller Zeit berechnet werden. Da das PCP-System vollständig ist, ergibt sich für $x \in L$ wie gewünscht $val(\varphi) = 1$. Aus der Korrektheit folgt zudem für $x \notin L$ wie benötigt $val(\varphi) \leq \frac{1}{2}$.

Theorem 3 impliziert Theorem 1
Angenommen ρ-GAPqCSP ist NP-schwer für zwei Konstanten q, ρ. Dann lässt sich zu jeder Sprache $L \in$ NP ein PCP-System V erzeugen, das logarithmisch viele Münzwürfe benötigt, maximal q Anfragen stellt und die Schranke ρ einhält. Zu einer Eingabe x berechnet V die $qCSP$ $\varphi = \{\varphi_i\}_{i=1}^m = f(x)$. Als Beweis erwartet der Algorithmus eine Belegung der Variablen in φ, wählt ein zufälliges $i \in \{1, \ldots, m\}$ und überprüft mit q Anfragen, ob $\varphi_i = 1$. Wenn $x \in L$, wird V den (korrekten) Beweis mit Wahrscheinlichkeit 1 akzeptieren. Ist dagegen $x \notin L$, so wird er mit einer Wahrscheinlichkeit von höchstens ρ akzeptieren. Ist $\rho > \frac{1}{2}$, führt man das Verfahren mehrfach durch, um die Korrektheit sicherzustellen (dies erzeugt lediglich einen konstanten Vorfaktor für die Anzahl an Münzwürfen und Anfragen).

Theorem 2 impliziert Theorem 3
Da Formeln in 3KNF ein Spezialfall von 3CSP-Instanzen sind, impliziert die zweite Sichtweise des PCP-Theorems das Theorem 3.

Theorem 3 impliziert Theorem 2
Angenommen es gibt $\epsilon > 0$ und $q \in \mathbb{N}$, sodass $(1-\epsilon)$-GAPqCSP NP-schwer ist. Sei φ eine qCSP-Instanz mit n Variablen und m constraints. Jeder constraint φ_i von φ kann als logische Formel in KNF mit höchstens 2^q Klauseln ausgedrückt werden, die jeweils höchstens q Literale enthalten. Nun bezeichne φ' die Menge von höchstens $m2^q$ Klauseln, die allen constraints in φ entsprechen. Wenn φ eine JA-Instanz von $(1-\epsilon)$-GAPqCSP (also erfüllbar) ist, dann existiert eine Belegung, die alle Variablen in φ' wahr macht. Ist φ dagegen eine NEIN-Instanz von $(1-\epsilon)$-GAPqCSP, dann verletzt jede Belegung wenigstens einen Anteil ϵ der constraints von φ und damit mindestens einen Anteil $\frac{\epsilon}{2^q}$ der Klauseln in φ'. Jede der $m2^q$ Klauseln mit jeweils höchstens q Literalen lässt sich durch Einführung von maximal q neuen Variablen in maximal q Klauseln überführen, die zusammen eine zur Ausgangsklausel äquivalente Formel in 3KNF ergeben. φ'' bezeichne die Menge von höchstens $qm2^q$ Klauseln über höchstens $n+qm2^q$ Variablen, die man in dieser Weise aus φ' erhält. φ'' die Konjunktion über alle diese Klauseln ist eine Formel in 3KNF. Die gesuchte Abbildung besteht darin, zunächst ein Eingabewort x auf eine qCSP abzubilden und dieses anschließend wie beschrieben auf eine Formel in 3KNF. Dies lässt sich in polynomieller Zeit berechnen, denn das qCSP ist in polynomieller Zeit berechenbar und die Umformungen sind polynomiell (nur die Konstante q tritt als Exponent auf). Wenn die Vollständigkeit für φ gilt, so gilt sie aufgrund der äquivalenten Umformung auch für φ' und φ''. Wenn jede Belegung mindestens einen Anteil ϵ der constraints in φ verletzt, so verletzt jede Belegung mindestens einen Anteil $\frac{\epsilon}{2^q}$ der Klauseln in φ' und schließlich verletzt damit jede Belegung mindestens einen Anteil $\frac{\epsilon}{q2^q}$ in φ''.

3 Zusammenfassung

Die eingangs aufgeworfene Frage, ob NP-schwere Optimierungsprobleme effizient beliebig genau approximiert werden können, muss höchstwahrscheinlich (nämlich wenn P $\neq$ NP) in vielen Fällen mit "nein " beantwortet werden. Als weitere Erkenntnis, die interessanterweise äquivalent dazu ist, ergibt sich, dass es für gewisse Beweise ausreicht, sie nur stichprobenartig zu lesen, um Fehler trotzdem mit hoher Wahrscheinlichkeit zu finden.
Es gibt viele weitere PCP-Theoreme, die für manche Probleme noch ernüchterndere Ergebnisse implizieren als für MAX-3SAT. So ist zum Beispiel jede approximative Bestimmung einer maximalen unabhängigen Knotenmenge in einem Graphen NP-schwer.

Quelle: Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 1st edition, 2009.