ETH zürich
Departement Informatik

Approximations- und Online-Algorithmen

Dr. Hans-Joachim Böckenhauer
Dr. Dennis Komm

courses.ite.inf.ethz.ch/approx_online_alg_22

Lösungsvorschläge – Blatt 4

Zürich, 24. März 2022

Lösung zu Aufgabe 6

Anpassung des FPTAS. Wir passen nur zwei Aspekte an.

1. Der Algorithmus berechnet auf der Ursprungsinstanz zuerst eine 2-Approximation[1] und bestimmt deren Kosten α.
2. Der Algorithmus wählt den Faktor t zum Verkleinern der Kosten neu als $t = \frac{1}{n} \cdot \frac{\varepsilon}{1+\varepsilon} \cdot \alpha$.

Laufzeit. Die optimalen Kosten sind höchstens 2α (das Doppelte der 2-Approximation) für die Ursprungsinstanz, für die modifizierte Instanz also höchstens $2\alpha/t = 2n\frac{1+\varepsilon}{\varepsilon}$. Somit hat die Anwendung von DPKP eine Laufzeit von $O\left(n \cdot 2n\frac{1+\varepsilon}{\varepsilon}\right)$, wie gewünscht quadratisch in n.

Approximationsgüte. Bezeichnen Alg und Opt die Kosten der Approximation des angepassten FPTAS und des Optimums. Wie für das ursprünglichen FPTAS gilt $\text{Opt} \leq \text{Alg} + nt$.[2] Die Approximationsgüte lässt sich also abschätzen durch:

$$\frac{\text{Opt}}{\text{Alg}} \leq \frac{\text{Opt}}{\text{Opt} - nt} = \frac{\text{Opt}}{\text{Opt} - \frac{\varepsilon}{1+\varepsilon}\alpha} \leq \frac{\text{Opt}}{\text{Opt} - \frac{\varepsilon}{1+\varepsilon}\text{Opt}} = \frac{1}{1 - \frac{\varepsilon}{1+\varepsilon}} = 1 + \varepsilon.$$

(bitte wenden)

[1] Dies geht sogar in Zeit $O(n \log n)$ wie folgt. Der Algorithmus nimmt die bessere von zwei Lösungen: Einerseits die Lösung mit einem einzelnen Gegenstand mit maximalen Kosten, andererseits die Greedy-Lösung, die bei Einpacken der Gegenstände mit absteigender Dichte entsteht. Falls es einen Gegenstand gibt, der mindestens die Hälfte der Kosten des Optimums hat, haben wir sicher eine 2-Approximation. Ansonsten betrachten wir die Greedy-Lösung und den ersten Gegenstand, der nimmt mehr reingepasst hat. Könnten wir von diesem Gegenstand Bruchstück einpacken, das den Rucksack genau füllen würde, wäre die entstehende Lösung mindestens so gut wie das Optimum, weil der Rucksack dann mit der bestmöglichen Dichte vollgefüllt ist. Nehmen wir das Bruchstück wieder heraus, so verlieren wir höchstens die Hälfte des Optimums.

[2] Die Modifikation der Kostenwerte entspricht einem Abrunden um höchstens t für jeden der n Gegenstände. Eine optimale Lösung der Ursprungsinstanz hat nach solchem Abrunden noch Kosten von mindestens $\text{Opt} - nt$ und Alg ist als Optimum der modifizierte Instanz mindestens so gross.

Lösung zu Aufgabe 7

(a) Sei $U \in \mathcal{NPO}$ ein Optimierungsproblem, bei dem für alle Probleminstanzen x alle möglichen Lösungswerte nur positive ganze Zahlen sein können und die Kosten $\text{cost}(\text{Opt}(x))$ der optimalen Lösung durch ein Polynom p über den beiden Variablen $|x|$ und Max-Int(x) beschränkt sind. Es gelte für die Schwellenwertsprache $Lang_U$, dass sie stark $\mathcal{NP}$-schwer ist.

Wir wollen zeigen, dass kein FPTAS für U existieren kann, wenn $\mathcal{P} \neq \mathcal{NP}$. Aus der Vorlesung wissen wir, dass es für kein Optimierungsproblem U mit einer stark $\mathcal{NP}$-schweren Schwellenwertsprache $Lang_U$ einen Pseudo-Polynomzeit-Algorithmus geben kann.

Beweis durch Widerspruch: Wir wollen zeigen, dass, wenn für das Problem U ein FPTAS A existiert, auch ein Pseudo-Polynomzeit-Algorithmus A' für U existiert. Sei A ein FPTAS für U. Es existiert der folgende Algorithmus A'.

Gegeben ist die Eingabeinstanz x, sei $\varepsilon = \dfrac{1}{p(|x|, \text{Max-Int}(x)) + 1}$.

Man wendet nun A auf x und ε an, um eine Lösung für x zu bekommen in einer Zeit, die polynomiell in $|x|$ und $1/\varepsilon = p(|x|, \text{Max-Int}(x)) + 1$ ist. Falls U ein Maximierungsproblem ist, gilt für die berechnete Lösung $A'(x)$:

$$\begin{aligned} |\text{cost}(A'(x)) - \text{cost}(\text{Opt}(x))| &= \text{cost}(\text{Opt}(x)) - \text{cost}(A'(x)) \\ &\leq (1+\varepsilon)\,\text{cost}(A'(x)) - \text{cost}(A'(x)) \\ &\leq \varepsilon\,\text{cost}(\text{Opt}(x)) \\ &= \frac{\text{cost}(\text{Opt}(x))}{p(|x|, \text{Max-Int}(x)) + 1}\ . \end{aligned}$$

Dabei haben wir verwendet, dass A ein Approximationsschema ist und daher $\text{cost}(\text{Opt}(x)) \leq (1+\varepsilon)\,\text{cost}(A'(x))$ gilt. Da ausserdem $\text{cost}(\text{Opt}(x)) \leq p(|x|, \text{Max-Int}(x))$ ist, können wir nun folgern, dass $|\text{cost}(A'(x)) - \text{cost}(\text{Opt}(x))| < 1$. Falls U ein Minimierungsproblem ist, können wir analog zeigen, dass $|\text{cost}(A'(x)) - \text{cost}(\text{Opt}(x))| = \text{cost}(A'(x)) - \text{cost}(\text{Opt}(x)) < 1$. Da aber alle möglichen Lösungswerte positive ganze Zahlen sind, muss in beiden Fällen die approximative Lösung optimal sein. Es folgt $\text{cost}(A'(x)) = \text{cost}(\text{Opt}(x))$. Damit ist $A'(x)$ ein Pseudo-Polynomzeit-Algorithmus für U.

Das ist ein direkter Widerspruch zur Annahme, dass $Lang_U$ stark $\mathcal{NP}$-schwer ist.

(b) Betrachten wir das TSP auf einem Graphen mit n Knoten und Kantenkosten 1 und 2 (das $\mathcal{NP}$-schwer ist, wie wir aus der Vorlesung wissen) und addieren $n!$ auf alle Kantenkosten. Daraus resultiert also ein Graph als Eingabe mit Kantenkosten $n! + 1$ und $n! + 2$.

Das so definierte Problem ist noch immer $\mathcal{NP}$-schwer, da es genauso schwer wie im ursprünglichen Problem ist, eine optimale Lösung zu finden. Die Kosten jeder Lösung (auch der optimalen) sind nun stets zwischen $n \cdot n! + n$ und $n \cdot n! + 2n$, und die Approximationsgüte R_{ALG} eines beliebigen Algorithmus ALG ist immer beschränkt durch

$$1 \leq R_{\text{ALG}} \leq \frac{n \cdot n! + 2n}{n \cdot n! + n} \leq 1 + \frac{1}{n!}\,.$$

Für $\varepsilon \geq \frac{1}{n!}$ hat jeder zulässige Algorithmus die geforderte Approximationsgüte, insbesondere auch der Greedy-Algorithmus, der in Polynomzeit bezüglich n arbeitet.

Für jedes $\varepsilon < \frac{1}{n!}$ können wir wiederum einen Algorithmus ALG mit der geforderten Approximationsgüte konstruieren, der einfach alle Knotenpermutationen ausprobiert, um einen billigsten Hamiltonkreis zu finden; dies kann mit einer Zeitkomplexität in $\mathcal{O}(n!)$ getan werden und somit gilt $\text{Time}_{\text{ALG}}(n, \varepsilon^{-1}) \leq p(n, \varepsilon^{-1})$ für ein Polynom p. Es folgt, dass die Approximationsgüte von ALG sogar 1 ist und der Algorithmus ferner eine Laufzeit besitzt, die polynomiell in n und ε^{-1} ist.

Somit ist ALG ein FPTAS für die vorgestellte TSP-Version.