Näherungsalgorithmen

(Approximationsalgorithmen)

WiSe 2024/25 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

19. November 2024

Näherungsalgorithmen Gesamtübersicht

- Organisatorisches

- Einführung / Motivation

- Grundtechniken für Näherungsalgorithmen

- Approximationsklassen (Approximationstheorie)

Grundtechniken

- Greedy-Verfahren
- Partitionsprobleme
- Lokale Suche
- Lineares Programmieren
- Dynamisches Programmieren

Maximum Independent Set (MIS)

I: Graphen $G = (V, E)$

S: $\{U \subseteq V \mid \underbrace{E(G[U]) = \emptyset\}}_{\text{„unabh. Menge"}}$

m: $|U|$

opt: max

Mitteilung: MIS_D ist NP-vollständig.

Heuristische Idee: günstig, kleingradige Knoten zu wählen.
$\rightsquigarrow$ GreedyIndependantSet($G = (V, E)$):

1. Setze $U := \emptyset$; $V' := V$.

2. Solange $V' \neq \emptyset$, tue:

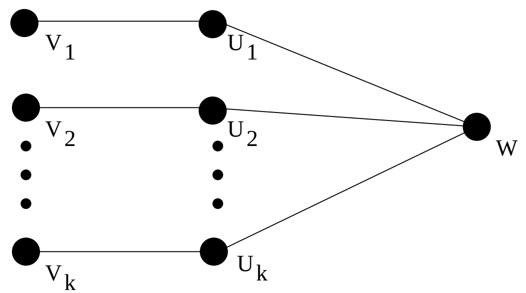
   2a: x sei Knoten kleinsten Grades in $G[V']$;

   2b: $U := U \cup \{x\}$;

   2c: $V' := V' \setminus N[x]$.

3. Liefere U

Der Greedy-Algorithmus kann beliebig schlechte Ergebnisse liefern.
Beispiel: Betrachte die folgenden Graphenschar: G_k



$V(G_k) = \{v_1, \ldots, v_k, u_1, \ldots, u_k, w\}$.
Kanten: (1) $G[\{v_1, \ldots, v_k\}] = K_k$ (*vollständiger Graph* mit k Knoten)
(2) $G[\{v_1, \ldots, v_k, u_1, \ldots, u_k\}]$ bildet (ohne Berücksichtigung der Kanten aus (1)) einen *vollständigen paaren Graphen* $K_{k,k}$ mit k Knoten auf jeder Seite.
(3) $G[\{u_1, \ldots, u_k, w\}] = K_{k,1}$
$\Rightarrow \quad E(G[\{u_1, \ldots, u_k\}]) = \emptyset$. $\rightsquigarrow \exists U \subseteq V, |U| = k, E(U) = \emptyset$
GreedyIndependentSet liefert aber erst w und dann z.B. v_1.

Mathematischer Exkurs:

Lemma 1: Betrachte $x_1, \ldots, x_n \geq 0$; setze $\overline{x} = \frac{\sum x_i}{n}$. Dann gilt:

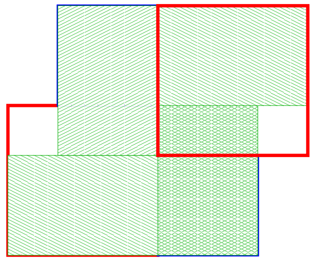
$$\sum {x_i}^2 \geq n \cdot \overline{x}^2$$

<u>Beweis:</u> Für $\Delta_i := x_i - \overline{x}$ gilt

$$\sum \Delta_i = \sum (x_i - \overline{x}) = \sum x_i - n \cdot \overline{x} = 0$$

$$\rightsquigarrow \sum {x_i}^2 = \sum (\overline{x} + \Delta_i)^2 = \sum \overline{x}^2 + 2\overline{x} \underbrace{\sum \Delta_i}_{=0} + \sum \Delta_i^2 \geq \sum \overline{x}^2 = n \cdot \overline{x}^2$$

Mathematischer Exkurs:



Lemma 2: $\forall x, y > 0 : \frac{x}{y} + \frac{y}{x} \geq 2$. 1. geometr. Bew. (o.E. $x > y$):

Beweis: 2. algebraischer Beweis:

$$\begin{aligned} \frac{x}{y} + \frac{y}{x} &= \frac{x^2 + y^2}{x \cdot y} \overset{\text{L 1}}{\geq} \frac{2 \cdot (\frac{x+y}{2})^2}{x \cdot y} \\ &= \frac{(x+y)^2}{2xy} = \frac{x^2 + 2xy + y^2}{2xy} \\ &= \frac{1}{2}\left(\frac{x}{y} + \frac{y}{x}\right) + 1. \end{aligned}$$

Satz 2: Betrachte Graph $G = (V, E)$, $n = |V|$, $m = |E|$. $\delta = \frac{m}{n}$ sei die *Dichte* von G. Der Wert $m_{Gr}(G)$, den GreedyIndependentSet(G) liefert, ist wenigstens $\frac{n}{2\delta+1}$.

Beweis: Es sei x_i der in der i-ten Iteration der Solange-Schleife in Schritt 2a ausgewählten Knoten und d_i bezeichne den Grad von x_i. In Schritt 2c werden x_i und seine Nachbarn entfernt, d.h. insgesamt $d_i + 1$ viele Knoten. Dadurch werden wenigstens $\frac{d_i(d_i+1)}{2}$ viele Kanten entfernt, denn

- x_i ist ein Knoten minimalen Grades im aktuellen Graphen und
- von jedem der insgesamt $d_i + 1$ vielen entfernten Knoten gehen insgesamt wenigstens $d_i(d_i + 1)$ viele (nicht notwendig verschiedene) Kanten aus; im schlimmsten Fall —wenn nämlich jeder der $d_i + 1$ Knoten aus der Nachbarschaft $N[x_i]$ mit jedem anderen Knoten aus $N[x_i]$ verbunden ist und sonst mit keinem anderen Knoten außerhalb von $N[x_i]$— sind es $\frac{d_i(d_i+1)}{2}$ viele verschiedene Kanten.

Summieren über alle $m_{Gr}(G)$ viele Iterationen des Programmes ergibt

(1) $\sum_{i=1}^{m_{Gr}(G)} \frac{d_i(d_i+1)}{2} \leq m = \delta n$.

Da der Algorithmus hält, wenn alle Knoten entfernt worden sind, gilt außerdem:

(2) $\sum_{i=1}^{m_{Gr}(G)} (d_i + 1) = n$.

Addieren wir (2) und (zweimal) (1), so bekommen wir: $\sum_{i=1}^{m_{Gr}(G)} (d_i + 1)^2 \leq n(2\delta + 1)$.
Der Mittelwert der $(d_i + 1)$ ist $\frac{\sum (d_i+1)}{m_{Gr}(G)} \overset{(2)}{=} \frac{n}{m_{Gr}(G)}$ und Lemma 1 liefert

$$n(2\delta + 1) \geq \sum (d_i + 1)^2 \geq m_{Gr}(G) \cdot \frac{n^2}{(m_{Gr}(G))^2} = \frac{n^2}{m_{Gr}(G)}.$$

Also gilt:

$$m_{Gr}(G) \geq \frac{n}{2\delta + 1}.$$

Satz 3: Mit den Bezeichnungen des vorigen Satzes gilt: $\frac{m^*(G)}{m_{Gr}(G)} \leq \delta + 1$.

Beweis: Es sei V^* ein MIS. Es bezeichne k_i die Anzahl der Knoten aus V^*, die unter den $d_i + 1$ im i-ten Schritt des Algorithmus entfernten Knoten sind. Natürlich gilt jetzt:
(3) $\sum_{i=1}^{m_{Gr}(G)} k_i = |V^*| = m^*(G)$
Wir wollen jetzt Abschätzung (1) verbessern.
Per Definition sind alle Knoten einer unabhängigen Menge nicht untereinander verbunden. Daher gehen nicht nur $\frac{d_i(d_i+1)}{2}$ viele paarweise verschiedene Kanten mindestens von Knoten in $N[x_i]$ aus, sondern noch „zusätzlich“ wenigstens $\frac{k_i(k_i-1)}{2}$ viele; $\rightsquigarrow$

(1’) $\sum_{i=1}^{m_{Gr}(G)} \frac{d_i(d_i+1)+k_i(k_i-1)}{2} \leq \delta n$

Jetzt addieren wir (2), (3) und (zweimal) (1’), um

$$\sum_{i=1}^{m_{gr}(G)} (d_i + 1)^2 + \sum_{i=1}^{m_{Gr}(G)} k_i^2 \leq n(2\delta + 1) + m^*(G)$$

zu erhalten.

Der Mittelwert der $(d_i + 1)$ ist $\frac{n}{m_{Gr}(G)}$
und der der k_i ist $\frac{\sum k_i}{m_{Gr}(G)} = \frac{m^*(G)}{m_{Gr}(G)}$, sodass jetzt Lemma 1 ergibt:

$$n(2\delta + 1) + m^*(G) \geq \frac{n^2 + (m^*(G))^2}{m_{Gr}(G)}$$

$$\begin{aligned}
\rightsquigarrow m_{Gr}(G) \quad & \geq \quad \frac{n^2 + (m^*(G))^2}{n(2\delta + 1) + m^*(G)} \\
& = \quad m^*(G) \cdot \frac{(\frac{n}{m^*(G)}) + (\frac{m^*(G)}{n})}{2\delta + 1 + (\frac{m^*(G)}{n})} \\
& \overset{m^*(G) \leq n \text{ \& L 2}}{\geq} \quad m^*(G) \cdot \frac{2}{2\delta + 1 + 1\frac{n}{n}} = m^*(G) \cdot \frac{1}{\delta + 1}.
\end{aligned}$$

Bemerkungen zu MIS und verwandten Problemen

1. Für manche Graphenklassen, z.B. die der Bäume, liefert GreedyIndependentSet stets eine optimale Lösung. Man sieht also, dass die genaue Spezifikation der Klasse I zulässiger Instanzen wichtig ist.

2. GreedyIndependentSet enthält in Schritt 2 einen gewissen (durch beliebige Wahl aufgelösten) Nichtdeterminismus. Selbst wenn man für Graphen weiß, dass es eine Wahlmöglichkeit in Schritt 2a gibt, sodass seine Variante von GreedyIndependentSet für diese Graphen die optimale Lösung findet, ist das auf solche Graphen eingeschränkte MIS$_D$-Problem NP-vollständig.

3. Das Auffinden größter Cliquen in einem Graphen (MaxClique) ist genauso schwer —auch im approximativen Sinne— wie das Auffinden kleinstmöglicher unabhängiger Mengen, denn:

$C \subseteq V$ ist maximale Clique in $G = (V, E)$ genau dann, wenn C ist maximale unabhängige Menge in $G^C = (V, E^C)$, $E^C = \{\{v, v'\} \mid v, v', v \in V, v \neq v', \{v, v'\} \notin E\}$, dem *Komplementgraphen*. $\leadsto$ heuristische Idee für MaxClique, möglichst großgradige Knoten zu wählen.

4. Unser Approximationsalgorithmus liefert für die Klasse G_d der Graphen mit Maximalgrad d eine maximale unabhängige Menge der Mindestgröße $\frac{n}{d+1}$. Der „Restgraph“ (erzeugt durch Entfernen der maximalen unabhängigen Menge) hat Maximalgrad $d-1$ (denn sonst könnte man einen Knoten mit Grad d noch in die unabhängige Menge nehmen, im Widerspruch zur Maximalität), sodass sich auf ihn wiederum der Approximationsalgorithmus anwenden ließe. Auf diese Weise kann man eine $(d+1)$-Färbung des Ursprungsgraphen erzeugen.

Ein kleines Beispiel: 

Optimaler Reiseweg eines Handelsreisenden durch die 15 größten Städte Deutschlands. Der Weg ist der kürzeste von $15! > 10^{13}$ oder auch $15! > 2^{40}$ möglichen.

Minimum Traveling Salesperson MTS (Handelsreisendenproblem)

I : vollständiger gerichteter Graph $G = (V, E), V = \{v_1, \ldots, v_n\}$
mit positiven Kantengewichten. (Abstandsfunktion; $D(v_i, v_j)$-Matrix über $\mathbb{Q}^+$)

S : $\{(v_{i_1}, \ldots v_{i_n}) \in V^n \mid [\forall j = 1, \ldots, n-1 : (v_{i_j}, v_{i_{j+1}}) \in E]$
und $(i_1, \ldots, i_n)$ ist Permutation von $(1, \ldots, n)\}$
$\hookrightarrow$ alternative Schreibweise: Wort der Länge n über V

m : $D(v_{i_1}, \ldots v_{i_n}) = \sum_{j=1}^{n-1} D(v_{i_j}, v_{i_{j+1}}) + D(v_{i_n}, v_{i_1})$ *Länge der Tour*

opt: min

MTS ist sehr bekannt. *G. Reinelt, The Traveling Salesman, Computational Solutions for TSP Applications, LNCS 840, Springer 1994,* ist ganz MTS gewidmet, ebenso: *D.L. Applegate, R.E. Bixby, V. Chvatal, W.J. Cook, The Traveling Salesman Problem – A Computational Study, Princeton Series in Applied Mathematics 2006* und *W.J. Cook, In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation, Princeton University Press 2012.*

Bezeichnung: Für Tour $f \in V^*$ ist $D(f) = D(E(f))$ wegen
$E(f) = \{(u, v) \in V^2 \mid \exists x, z \in V^* : f = xu \cdot vz$ oder $f = vxu\}$.

Spezialfall: metrisches MTS (kurz: MMTS), d.h. D ist *Metrik*, also:

- $\forall u, v : D(u, v) = D(v, u)$ (also „ungerichteter Graph“)
- $\forall u, v, w : D(u, w) \leq D(u, v) + D(v, w)$ ($\triangle$-*Ungleichung*)
- $\forall u, v : D(u, v) = 0 \Leftrightarrow u = v$

Aus der $\triangle$-Ungleichung folgt unmittelbar: Ist
$f' = (v_{i_{j_1}}, \ldots v_{i_{j_k}})$ eine Teilfolge von $f = (v_{i_1}, \ldots v_{i_n})$, so gilt $D(f') \leq D(f)$.

Mitteilung MTS_D und MMTS_D sind NP-vollständig.

Greedy-Algorithmen für Minimierungsprobleme

Erinnerung: Idee für Überdeckungsprobleme:

Starte mit Gesamtmenge X

(ist zulässige Lösung, denn $f(X) = 1$)

Entferne sukzessive Elemente, sodass immer noch die Zulässigkeit

(modelliert durch $f(X') = 1$ für $X' \subseteq X$) gewährleistet ist.

Erinnerung: Überdeckungseigenschaft ist „monoton".

Alternative Idee:

Geeigneter Begriff einer „zulässigen Teillösung".

Dann: $\emptyset$ ist zulässig, und sukzessiver Aufbau „von unten" ist möglich.

Abgebrochen wird, falls die Teillösung auch eine Lösung ist.

$\rightsquigarrow$ nachfolgendes MMTS-Beispiel

Ein Approximationsalgorithmus für (M)MTS

MMTS$_{NN}(G = (V, E), D)$

1. Setze $P := v$ für irgendein $v \in V$ und $P_{end} := v$.

2. Setze $V' := V \setminus \{v\}$.

3. Solange $V' \neq \emptyset$ tue:

   3a. v sei ein Knoten aus V', sodass
   $\forall u \in V' : D(P_{end}, v) \leq D(P_{end}, u)$

   3b. $P_{end} := v$; $P := P \cdot P_{end}$.
   [$\cdot$ steht für die Konkatenation.]

   3c. $V' := V' \setminus \{v\}$.

4. Liefere P.

Durch die Verwaltung der Menge V' ist klar, dass P ein zu keinem Zeitpunkt zweimal denselben Knoten beinhaltet. M.a.W: das zurückgelieferte P ist in jedem Fall zulässig.

Lemma 3: Ist (G, D) eine Instanz von MMTS, und sei $G = (V, E)$, $n = |V|$. Existiert eine Abbildung $\ell : V \to \mathbb{Q}$ mit

1. $\forall u, v \in V, u \neq v : D(u, v) \geq \min(\ell(u), \ell(v))$
2. $\forall v \in V : \ell(v) \leq \frac{1}{2} m^*(G, D)$,

so gilt:

$$\sum_{v \in V} \ell(v) \leq \frac{1}{2}(\lceil \log n \rceil + 1) \cdot m^*(G, D)$$

Beweis: Es seien $v_1, \ldots, v_n$ absteigend gemäß ihrer ℓ-Werte sortiert. Wir beweisen zunächst:

$$\forall 1 \leq k \leq n : m^*(G,D) \geq 2 \cdot \sum_{i=k+1}^{\min(2k,n)} \ell(v_i).$$

Dazu sei P^* eine bestmögliche Tour der Länge $m^*(G,D)$.
Betrachte $V_k = \{v_i \in V \mid 1 \leq i \leq \min(2k,n)\}$.
P_k sei diejenige Teiltour von P^*, die entsteht, wenn man nur die „Städte" aus V_k in der durch P^* vorgegebenen Reihenfolge durchläuft. Wegen obiger Bemerkung ist

$$D(P_k) \leq D(P^*) = m^*(G,D).$$

Wegen der ersten Voraussetzung gilt ferner:

$$D(P_k) \geq \sum_{(u,v)\in E(P_k)} \min(\ell(u),\ell(v)) = \sum_{v_i \in V_k} \alpha_i \ell(v_i)$$

für gewisse $\alpha_i \in \{0,1,2\}$.

Genauer gilt:

$$\alpha_i = |\{v_j \in V_k \mid \{v_i, v_j\} \in E \wedge j < i\}|\,,$$

denn die v_i werden als absteigend sortiert angenommen.
Außerdem ist

$$\sum_{v_i \in V_k} \alpha_i = |V_k| \leq 2k.$$

Wegen der absteigenden Sortierung der v_i können wir

$$\sum_{v_i \in V_k} \alpha_i \ell(v_i) \geq 2 \cdot \sum_{i=k+1}^{\min(2k,n)} \ell(v_i)$$

abschätzen, indem wir $\alpha_1 = \ldots = \alpha_k = 0$ und $\alpha_{k+1} = \ldots = \alpha_{\min(2k,n)} = 2$ annehmen. $\leadsto$

$$m^*(G, D) \geq D(P_k) \geq 2 \cdot \sum_{i=k+1}^{\min(2k,n)} \ell(v_i).$$

Summieren wir diese Ungleichung für

$$k = 2^j, \quad j = 0, \ldots, \lceil \log n \rceil - 1,$$

so erhalten wir:

$$\lceil \log n \rceil m^*(G, D) = \sum_{j=0}^{\lceil \log n \rceil - 1} m^*(G, D)$$

$$\geq \sum_{j=0}^{\lceil \log n \rceil - 1} \left(2 \cdot \sum_{i=2^j+1}^{\min(2^{j+1}, n)} \ell(v_i) \right) = 2 \cdot \sum_{i=2}^{n} \ell(v_i)$$

Mit der zweiten Voraussetzung, aus der $m^*(G, D) \geq 2 \cdot \ell(v_1)$ folgt, ergibt sich die Behauptung des Lemmas. □

Satz 4: Es sei $(G = (V, E), D)$ eine Instanz von MMTS, $|V| = n$. Ist $m_{NN}(G, D)$ die Länge der Lösung der von MMTS$_{NN}(G, D)$ gelieferte Tour, so gilt:

$$\frac{m_{NN}(G, D)}{m^*(G, D)} \leq \frac{1}{2}(\lceil \log n \rceil + 1)$$

Bem.: Logarithmische, nicht konstante Approximationsgüte!
Später besser!

Beweis: Wir wollen das vorige Lemma anwenden. Dazu: für die von MMTS$_{NN}$ gelieferte Tour $P = v_{i_1}, \dots v_{i_n}$ legen wir fest:

$$\ell(v_{i_j}) = \begin{cases} D(v_{i_j}, v_{i_{j+1}}), & 1 \leq j < n \\ D(v_{i_n}, v_{i_1}), & j = n \end{cases}$$

Augenscheinlich ist

$$m_{NN}(G, D) = \sum_{j=1}^{n} \ell(v_j)$$

Warum dürfen wir das Lemma anwenden, um Satz 4 zu zeigen?

Zu Voraussetzung 1: Es seien v_{i_j}, v_{i_k} zwei Städte der von MMTS$_{NN}$ gelieferten Tour.
Ist $j < k$, dann ist

$$\ell(v_{i_j}) = D(v_{i_j}, v_{i_{j+1}}) \leq D(v_{i_j}, v_{i_k}),$$

denn sonst wäre v_{i_k} statt $v_{i_{j+1}}$ vom Algorithmus MMTS$_{NN}$ gewählt worden.
Ist $j > k$, dann gilt entsprechend

$$\ell(v_{i_k}) = D(v_{i_k}, v_{i_{k+1}}) \leq D(v_{i_k}, v_{i_j}).$$

Zu Voraussetzung 2: Betrachte einen Knoten $v \in V$.
Nach Definition von ℓ gibt es einen „benachbarten“ Knoten $u \in V$ auf der MMTS$_{NN}$-Tour mit $\ell(v) = D(v, u)$.
Gemäß einer optimalen Tour P^* gibt es zwischen v und u zwei disjunkte Wege;
jeder dieser Wege hat wegen der $\triangle$-Ungleichung mind. die Länge $D(v, u)$. Daher ist:

$$m^*(G, D) = D(P^*) \geq 2 \cdot D(v, u) = 2 \cdot \ell(v).$$

Hinweis: Popularität von TSP ermisst sich leicht im Internet.

Oft werden hier auch *Metaheuristiken* getestet, wie genetische Algorithmen.