Der Approximationsalgorithmus von Christofides

Problem: Traveling Salesman
Input: Ein Graph $G = (V, E)$ mit einer Distanzfunktion $d : E \to \mathbb{Q}_{\geq 0}$.

Aufgabe: Finde eine Tour, die alle Knoten des Graphen G genau einmal besucht und deren Länge bezüglich d möglichst klein ist, oder stelle fest, dass keine solche Tour existiert.

Anwendungsbeispiele des TSP:

- Routenplanungs-Probleme
- Bestückung von Platinen mit elektronischen Bauelementen
- Steuerung von Schweißrobotern
- optimale Anordnung von Leitungen auf einem Halbleiter-Chip

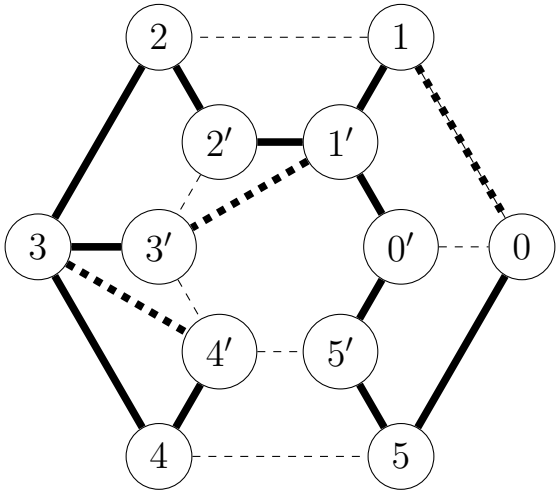
Komplexität: $\mathcal{NP}$-vollständig

Approximierbarkeit: Sie kennen bereits ein Nicht-Approximierbarkeitsresultat. Approximationsalgorithmen gibt es dennoch für spezielle Problemklassen. Hier betrachten wir einen Approximationsalgorithmus für das **metrische TSP**, bei dem die Distanzfunktion d der Dreiecksungleichung genügt.

Approximationsfaktor von Christofides: $\frac{3}{2}$

Grundidee:

- Konstruiere eine Eulertour in einem aufspannenden Subgraphen von G (d.h. eine Tour, die alle Kanten des Subgraphen abfährt). Eventuell muss die Existenz einer solchen Tour durch einen Trick gesichert werden.
- Verkürze die Tour zu einer TSP-Tour (hier geht die Dreiecksungleichung ein).



Traveling Salesman – Nearest-Insert-Heuristik

Problem: Traveling Salesman
Input: Ein Graph $G = (V, E)$ mit einer Distanzfunktion $d : E \to \mathbb{Q}_{\geq 0}$.

Aufgabe: Finde eine Tour, die alle Knoten des Graphen G genau einmal besucht und deren Länge bezüglich d möglichst klein ist, oder stelle fest, dass keine solche Tour existiert.

Anwendungsbeispiele des TSP:

- Routenplanungs-Probleme
- Bestückung von Platinen mit elektronischen Bauelementen
- Steuerung von Schweißrobotern
- optimale Anordnung von Leitungen auf einem Halbleiter-Chip

Komplexität: $\mathcal{NP}$-vollständig

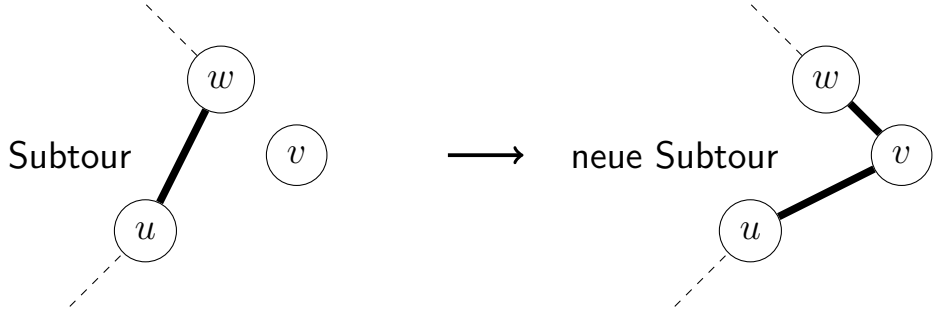
Approximierbarkeit: Sie kennen bereits ein Nicht-Approximierbarkeitsresultat. Approximationsalgorithmen gibt es dennoch für spezielle Problemklassen. Hier betrachten wir einen Approximationsalgorithmus für das **metrische TSP**, bei dem die Distanzfunktion d der Dreiecksungleichung genügt.

Approximationsfaktor der Nearest-Insert-Heuristik: 2

Ähnliche Heuristik: Cheapest-Insert-Heuristik

Grundidee:

- Die Heuristik baut nach und nach eine Tour auf. Sie baut dabei sukzessive Knoten in eine bestehende Subtour im Graphen ein und wählt immer einen Knoten aus, der möglichst nahe („nearest“) an der bereits bestehenden Tour liegt.
- Die dabei gestrichene Kante wird so gewählt, dass die neue Subtour möglichst kurz bleibt.



Traveling Salesman – Cheapest-Insert-Heuristik

Problem: Traveling Salesman
Input: Ein Graph $G = (V, E)$ mit einer Distanzfunktion $d : E \to \mathbb{Q}_{\geq 0}$.

Aufgabe: Finde eine Tour, die alle Knoten des Graphen G genau einmal besucht und deren Länge bezüglich d möglichst klein ist, oder stelle fest, dass keine solche Tour existiert.

Anwendungsbeispiele des TSP:

- Routenplanungs-Probleme
- Bestückung von Platinen mit elektronischen Bauelementen
- Steuerung von Schweißrobotern
- optimale Anordnung von Leitungen auf einem Halbleiter-Chip

Komplexität: $\mathcal{NP}$-vollständig

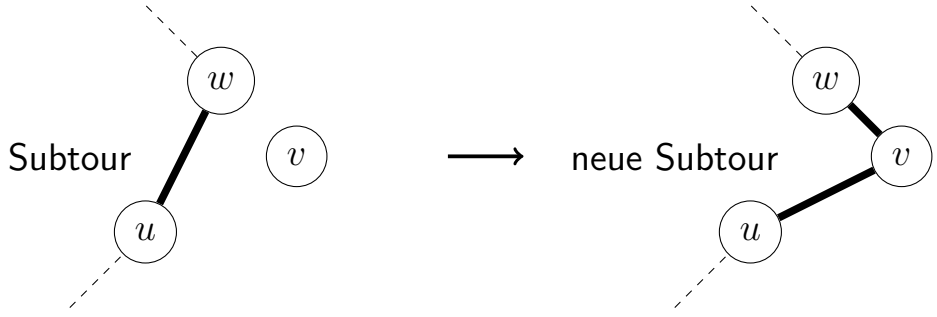
Approximierbarkeit: Sie kennen bereits ein Nicht-Approximierbarkeitsresultat. Approximationsalgorithmen gibt es dennoch für spezielle Problemklassen. Hier betrachten wir einen Approximationsalgorithmus für das **metrische TSP**, bei dem die Distanzfunktion d der Dreiecksungleichung genügt.

Approximationsfaktor der Cheapest-Insert-Heuristik: 2

Ähnliche Heuristik: Nearest-Insert-Heuristik

Grundidee:

- Die Heuristik baut nach und nach eine Tour auf. Sie baut dabei sukzessive Knoten in eine bestehende Subtour im Graphen ein. Die Wahl des einzubauenden Knotens erfolgt so, dass die Kosten der neuen Subtour so niedrig wie möglich bleiben („cheapest“).
- Die für eine neue Subtour gestrichene Kante wird so gewählt, dass die neue Subtour möglichst kurz bleibt.



Steinerbaum – der KMB-Algorithmus

Problem: Steinerbaum

Input: Ein Graph $G = (V, E)$ mit einer Kostenfunktion $c : E \to \mathbb{Q}_{\geq 0}$, eine Terminalmenge $T \subseteq V$.

Aufgabe: Finde einen zusammenhängenden Subgraphen von G, der alle Knoten aus T enthält und bezüglich c möglichst geringe Kosten besitzt, oder stelle fest, dass kein solcher Subgraph existiert.

Anwendungsbeispiele des Steinerbaumproblems:

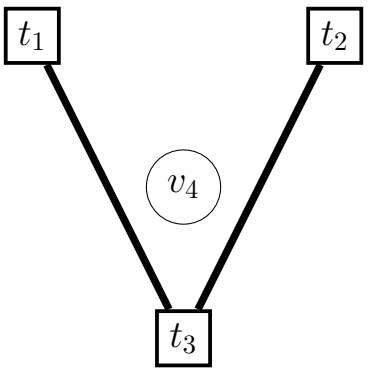
- Telekommunikationsnetze
- Gasversorgungsleitungen
- gemietete Netzleitungen für eine Konferenzschaltung mit mehreren Standorten

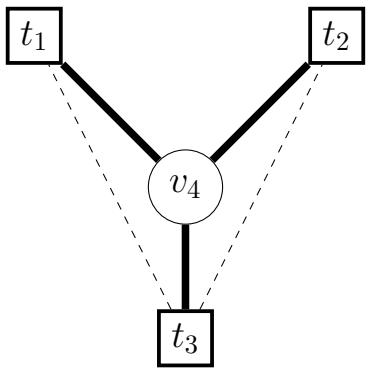
Komplexität: $\mathcal{NP}$-vollständig

Approximationsfaktor des Kou-Markovsky-Berman-Algorithmus: 2

Grundidee:

- Der Algorithmus stellt zunächst den Distanzgraphen G_T auf. Dabei handelt es sich um einen vollständigen Graphen auf der Knotenmenge T, dessen Kantengewichte sich aus Kürzeste-Wege-Berechnungen ergeben.
- In G_T wird dann ein minimaler spannender Baum gesucht, der anschließend mittels der entsprechenden kürzesten Wege in einen Subgraphen G_S von G zurückübersetzt wird.
- Geeignetes Ausdünnen dieses Subgraphen liefert schließlich den gesuchten Steinerbaum.





Knapsack

Problem: Knapsack

Input: Eine Zahl $n \in \mathbb{N}$, Gewichte $w \in \mathbb{N}^n$ und Werte $v \in \mathbb{N}^n$ sowie eine Kapazität $K \in \mathbb{N}$.

Aufgabe: Finde eine Teilmenge $I \subseteq \{1, \ldots, n\}$ mit $\sum_{i \in I} w_i \leq K$, die größtmöglichen Gesamtwert besitzt.

Anwendungsbeispiele des Knapsack-Problems:

- Packen von Versandkartons oder Containern
- Beladung von Flugzeugen, LKWs und Schiffen
- Hilfsproblem bei vielen ganzzahligen Programmen und Verfahren (z. B. Schnittebenen-Verfahren)

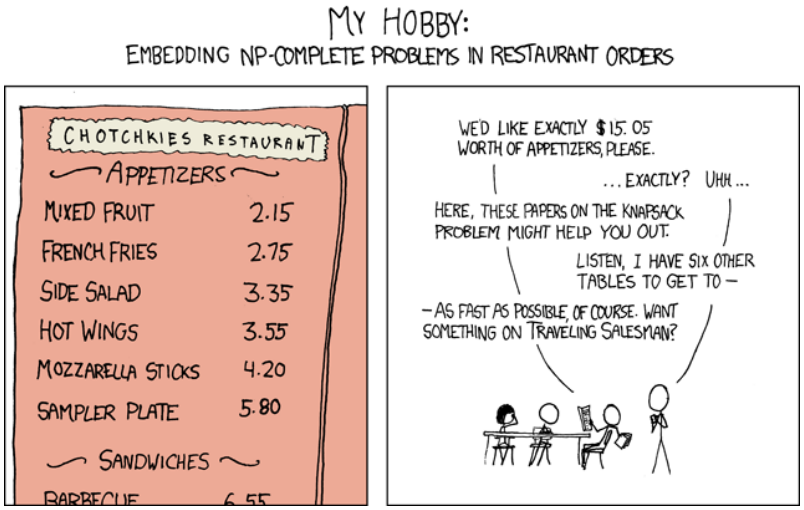
Komplexität: $\mathcal{NP}$-vollständig

Approximierbarkeit: pseudopolynomieller exakter Algorithmus, beliebig genaue polynomielle Approximation

Approximationsfaktor: $1 - \varepsilon$ für beliebiges $\varepsilon > 0$

Grundidee:

- Modifikation des pseudopolynomiellen Algorithmus (dynamische Optimierung).
- Die auftretenden Zahlen (die zu pseudopolynomieller Laufzeit führen) werden gerundet. Je nach Größenordnung der Rundung verringert sich die Laufzeit, entsprechend wird aber auch die Genauigkeit des Algorithmus schlechter.
- Zu vorgegebener Genauigkeit ε kann ein Algorithmus mit polynomieller Laufzeit konstruiert werden, ein sogenanntes *Approximationsschema* (genauer: ein *FPTAS*, d. h. ein *fully polynomial time approximation scheme*.)



k-Matching – Randomisierte Methoden

Problem: k-Matching
Input: Ein Graph $G = (V, E)$, eine Zahl $k \in \mathbb{N}$.

Aufgabe: Finde ein k-Matching in G, das möglichst viele Kanten enthält. Ein k-Matching ist eine Kantenmenge $M \subseteq E$ mit der Eigenschaft, dass jeder Knoten höchstens mit k Kanten aus M inzident ist.

Anwendungsbeispiele des k-Matching-Problems:

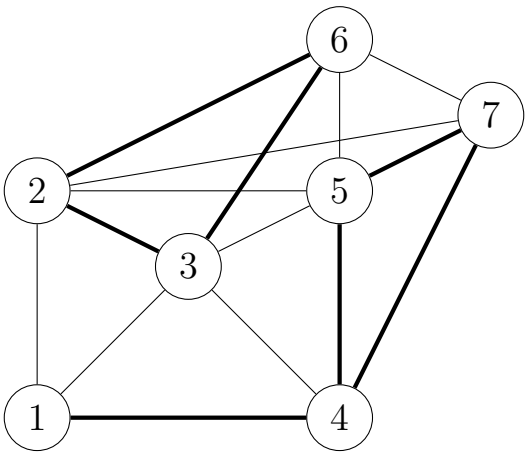
- Maschinenbelegung
- Zuordnung von Kunden- und Lieferanten-Standorten in der Logistik
- Erstellen von Serviceplänen

Komplexität: $\mathcal{NP}$-vollständig

Approximationsfaktor der randomisierten Rundungsheuristik: Falls k für einen gegebenen Parameter ε ausreichend groß ist (mindestens $k(\varepsilon)$), findet die Heuristik mit hoher Wahrscheinlichkeit eine $(1 - \varepsilon)$-Approximation.

Grundidee:

- Schreibe das Problem als LP.
- Berechne eine optimale Lösung x^* für die LP-Relaxation.
- Runde alle fraktionellen Komponenten x_i^* von x^* mit einer Wahrscheinlichkeit von $(1 - \varepsilon/2)x_i^*$ auf 1 (bzw. mit Wahrscheinlichkeit $1 - (1 - \varepsilon/2)x_i^*$ auf 0).
- Vorsicht: Die Lösung muss dann nicht unbedingt zulässig sein. Der Algorithmus muss also eventuell mehrmals wiederholt werden.



2-Matching