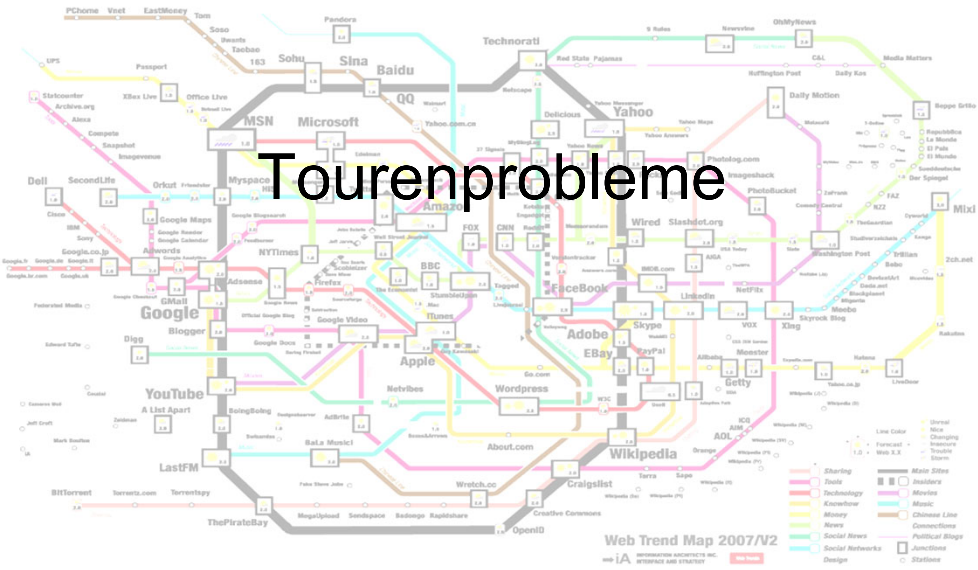


Tourenprobleme

Kantenbezogene Aufgaben

Eine geschlossene Kantenfolge, die jede Kante eines Graphen genau einmal enthält, heißt eine ***Eulertour****. Ein Graph, der eine Eulertour besitzt, heißt ein* ***eulerscher Graph****. Eine Kantenfolge, die jede Kante eines Graphen genau einmal enthält und nicht geschlossen ist, heißt ein* ***Eulerpfad****. Eine geschlossene Kantenfolge heißt ein* ***Eulerkreis****, wenn alle Kanten $e_1, \ldots, e_k$ voneinander verschieden sind und $v_0=v_k$ gilt.*

Ein ungerichteter Graph besitzt genau dann eine Eulertour, wenn jede Ecke einen geraden Grad besitzt.

Ein ungerichteter Graph besitzt genau dann einen Eulerpfad, wenn genau zwei Ecken einen ungeraden Grad besitzen. Diese beiden Ecken sind die erste und die letzte Ecke des Eulerpfads.

Ein gerichteter Graph besitzt genau dann eine Eulertour, wenn er stark zusammenhängend ist und für jeden Knoten sind Eingangsgrad und Ausgangsgrad gleich.

Kantenbezogene Aufgaben

Algorithmus von Fleury: *Gegeben sei ein eulerscher Graph G = (V,E). Ausgabe ist die Eulertour $W_{|E|}$.*

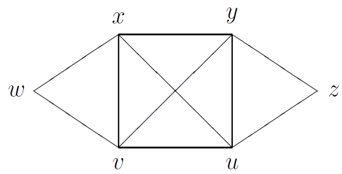
Schritt 1: *Man wähle eine beliebige Ecke v_0 in G und setze $W_0 = v_0$.*

Schritt 2: *Wenn der Kantenzug $W_i = v_0 e_1 v_1 \ldots e_i v_i$ gewählt worden ist (so dass alle $e_1 \ldots e_i$ unterschiedlich sind), wähle man eine von $e_1 \ldots e_i$ verschiedene Kante e_{i+1}, so dass*

1. e_{i+1} inzident mit v_i ist und

2. ausgenommen, es gibt keine Alternative, e_{i+1} keine Schnittkante des Teilgraphen $G \setminus \{e_1, \ldots, e_i\}$ ist.

Schritt 3: *Man beende den Algorithmus, wenn W_i jede Kante von G beinhaltet. Andernfalls ist Schritt 2 zu wiederholen.*



Kantenbezogene Aufgaben

Ein zusammenhängender Graph G ist dann und nur dann eulersch, wenn G die Vereinigung von kantendisjunkten Zyklen (Eulerkreisen) ist.

Algorithmus von Hierholzer: *Gegeben sei ein eulerscher Graph G = (V,E). Ausgabe ist die Eulertour K.*

Schritt 1: *Man wähle einen beliebigen Knoten v_0 in G und konstruiere von v_0 ausgehend einen Unterkreis (Eulerkreis) K in G, der keine Kante in G zweimal durchläuft.*

Schritt 2: *Wenn K eine Eulertour ist, gehe zu 4. Andernfalls: gehe zu Schritt 3.*

Schritt 3: *1. Vernachlässige alle Kanten des Eulerkreises K.*

2. Eine beliebige Ecke von K, zu der nicht vernachlässigte Kanten inzident sind, startet man einen weiteren Eulerkreis K' (analog Schritt 1).

3. Füge in K den zweiten Kreis K' ein, indem die Startecke von K' in K durch alle Ecken von K' ersetzt wird. Gehe nun zu Schritt 2.

Schritt 4: *Gebe K als Eulertour aus.*

Chinesische Briefträgerproblem

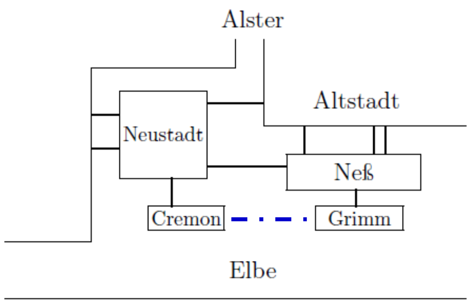
Ein Postbote soll in seinem Zustellbezirk jede Straße (mindestens) einmal entlanggehen. Insgesamt möchte er einen möglichst kurzen Weg zurücklegen (d.h. er möchte möglichst selten eine Straße zweimal durchlaufen müssen).

Algorithmus von Edmonds *Gegeben sei ein Graph G.*

1. *Finde die Menge U aller Ecken ungeraden Grades in G und bestimme für je zwei dieser Ecken u_i, u_j die Länge w_{ij} des kürzesten Weges zwischen ihnen (zB Floyd-Warshall Algorithmus).*
2. *Konstruiere den vollständigen Graphen $K_{|U|}$ mit Eckenmenge U und Kantenbewertungen max − w_{ij}, wobei max eine hinreichend große Zahl ist.*
3. *Bestimme in diesem vollständigen Graphen ein Matching M mit maximaler Kantengewichtssumme (nicht behandelt).*
4. *Für jede Kante $u_iu_j \in M$ verdoppele in G die Kanten eines kürzesten Weges von u_i nach u_j.*
5. *Der so geänderte Graph G ist jetzt eulersch und kann von einer beliebigen Ecke ausgehend „in einem Zug" durchlaufen werden (zB Algorithmus von Fleury).*

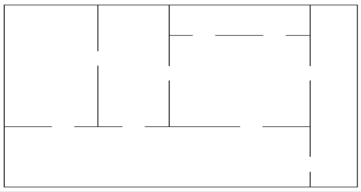
Aufgaben

1. Zu Beginn des 13. Jahrhunderts, als Hamburg noch etwas übersichtlicher war als heute, erstreckte sich das Stadtgebiet über Teile des Alsterufers und mehrere Inseln in der Alstermündung. Die verschiedenen Gebietsteile waren durch insgesamt 9 Brücken untereinander verbunden. Wäre es damals möglich gewesen, auf einem Sonntagsspaziergang alle Brücken genau einmal zu passieren? Wie ändert sich die Lage, wenn man zwischen Cremon und Grimm auf einem Boot übersetzen kann?



Aufgaben

2. In dieser Aufgabe betrachten wir eine Verallgemeinerung der Domino-Aufgabe: Auf den Spielsteinen stehen die Zahlen 0 bis k (k ∈ N) und es gibt für jedes Zahlenpaar einen Spielstein i/j (i ≤ j), also 0/0, 0/1, . . . 0/k, 1/1, 1/2 . . . 1/k, 2/2, . . . k/k.
   a) Begründen Sie, dass die Aufgabe, sämtliche Steine in einer geschlossenen Kette unterzubringen, nur für gerade Zahlen k lösbar ist.
   b) Wenn k ungerade ist, werden die Steine 0/1, 2/3, 4/5, . . . , (k − 1)/k aus dem Dominospiel entfernt. Begründen Sie, dass die Domino-Aufgabe dennoch lösbar ist.
3. Gegeben sei nachfolgender Grundriss einer Wohnung. Ist es möglich in dieser Wohnung einen Rundgang zu machen, so dass man durch jede Tür genau einmal hindurch kommt ?



Eckenbezogene Aufgaben

Ein ***Hamiltonscher Weg*** *in einem Graphen G ist ein Weg, der jede Ecke von G enthält. Ein* ***Hamiltonscher Kreis*** *(oder Hamiltonischer Zyklus) in einem Graphen G ist ein Kreis, der jede Ecke von G enthält. Ein Graph heißt* ***hamiltonsch****, wenn er einen hamiltonschen Kreis enthält.*

Satz von Dirac: Wenn G ein schlichter Graph mit n Ecken, mit 3 ≤ n ∈ N, ist und der Grad jeder Ecke v von G d(v) ≥ n/2 beträgt, dann ist G hamiltonsch. (positives Kriterium)

Löscht man in einem hamiltonschen Graphen n Ecken, so zerfällt der Graph in höchstens n Teilgraphen. (Ausschlusskriterium)

Gegeben sei ein schlichter Graph G_0 mit n Ecken. Solange es zwei nicht adjazente Ecken v_1, v_2 in G_i gibt, so dass $d(v_1) + d(v_2) \geq n$ in G_i ist, verbinde man diese beiden Ecken, um einen neuen Obergraphen G_{i+1} zu bilden. Der letzte Obergraph, der so erhalten wird, heißt ***Hülle*** *oder auch* ***Hamiltonabschluss*** *von G_0 und wird mit $c(G_0)$ bezeichnet.*

Eckenbezogene Aufgaben

Satz vo Bondy und Chvatal: Ein schlichter Graph G ist dann und nur dann hamiltonsch, wenn seine Hülle c(G) hamiltonsch ist.

***naiver Algorithmus**: Vorausgesetzt es gibt einen hamiltonschen Kreis, können wir ihn wie folgt finden:*

Wenn wir eine Ecke in den möglichen Kreis aufgenommen haben, löschen wir bei dem Vorgänger (nicht für die Startecke) alle nicht benutzten und mit dem Vorgänger inzidenten Kanten (, da wir diese Ecke nicht mehr besuchen werden).

Wenn eine der nachfolgenden Situationen eintritt (außer am Ende der Suche, wenn es sich nicht mehr vermeiden lässt), müssen wir wieder zum Vorgänger zurück und eine andere Fortsetzung wählen (Backtracking):

- *Es dürfen keine Ecken mit Grad 1 entstehen.*
- *Es darf kein nicht zusammenhängender Graph entstehen.*

Rundreiseproblem

Ein Handlungsreisender im Ort v_1 will Kunden in den Orten $v_2, \ldots, v_n$ besuchen und dann nach Hause zurückkehren. Die Entfernung zwischen den Orten v_i und v_j sei jeweils $w_{ij} \geq 0$.

Es gelte für beliebige drei unterschiedliche Zahlen i, j, k = 1, …, n: $w_{ij} \leq w_{ik} + w_{kj}$. Dann lässt sich das TSP näherungsweise mit einem polynomialen Algorithmus lösen. Der einfachste Algorithmus dieser Art bestimmt eine Rundreise, die höchstens das Doppelte der minimalen Länge aufweist (max ≤ min + min = 2min):

1. *Finde ein Minimalgerüst T in dem vollständigen Graphen K_n. (Die Summe der Kantenbewertungen des Minimalgerüsts ist höchstens so groß wie die Kantenbewertungssumme einer kürzesten Rundreise.)*
2. *Verdoppele alle Kanten aus T. Dadurch wird T eulersch. Bestimme eine Eulertour in T und notiere die zugehörige Folge der Ecken. (Die Kantenbewertungssumme der Eulertour ist genau doppelt so groß wie die des Minimalgerüsts.)*
3. *Streiche in dieser Folge alle Ecken mit Ausnahme ihres ersten Auftretens und mit Ausnahme der letzten Ecke. Die verbleibende Eckenfolge definiert eine Rundreise im Graphen K_n. (Aufgrund der Dreiecksungleichung ist ihre Kantenbewertungssumme nicht größer als die der Eulertour.)*

Rundreiseproblem

Der Abstand einer Ecke v von einer Kantenfolge W ist definiert als $d(v,W) = \min\{d(v,u) \mid u$ ist eine Ecke von $W\}$, wobei $d(v,u)$ die Kosten dieser Kante sind. Die Ecke v, die nicht in W liegt, wird als ***dichteste Ecke*** *zu W bezeichnet, wenn für jede andere nicht in W befindliche Ecke x $d(v,W) \leq d(x,W)$ gilt.*

Methode der Einführung der dichtesten Ecke*: Gegeben sei ein Graph $K_n = (V,E)$ mit $V= \{v_1, v_2, \ldots, v_n\}$. Der aktuelle gewählte Kreis W_i wird stets neu nummeriert und ist gegeben durch $W_i = u_1u_2 \ldots u_iu_1$, $u_j \in V$. Diese neue Nummerierung verändert nicht die Reihenfolge der bisher gewählten Ecken!*

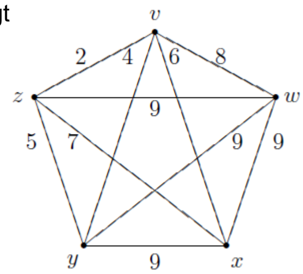
Schritt 1: Man wähle eine beliebige Ecke $u_1 = v_j \in V$ als Startecke und setze $W_1 = u_1$.

Schritt 2: Aus der Zahl der $n-i$ Ecken, die bisher noch nicht gewählt worden sind, ermittle man eine Ecke $u_{i+1} = v_k$, die am dichtesten zu W_i liegt. Sei $W_i = u_1u_2 \ldots u_iu_1$. Man bestimme dann, welche der Kantenfolgen (Kreise) $u_1u_{i+1}u_2u_3 \ldots u_iu_1$, $u_1u_2u_{i+1}u_3 \ldots u_iu_1, \ldots, u_1u_2u_3 \ldots u_iu_{i+1}u_1$ die kürzeste ist. Es sei W_{i+1} diese kürzeste Kantenfolge. Man nummeriere sie, wenn nötig, neu als $u_1u_2 \ldots u_{i+1}u_1$. Man setze $i := i + 1$.

Schritt 3: Wenn W_i alle Ecken beinhaltet, beende den Algorithmus, sonst führe Schritt 2 aus.

Aufgaben

1. Überzeugen Sie sich davon, dass die Kantenbewertungen in dem Graphen K_5 die Dreiecksungleichung erfüllen, und geben Sie eine Rundreise an, deren Länge höchstens das Doppelte der minimalen Länge beträgt



2. Warum ist die Länge jeder Rundreise mindestens so groß wie die Kantengewichtssumme eines Minimalgerüsts?

Aufgaben

3. Zeichnen Sie alle (bis auf Isomorphie gleiche) einfachen ungerichteten hamiltonschen Graphen (a) mit drei Ecken, (b) mit vier Ecken. Warum ist der Zusatz „einfach" in dieser Aufgabe wichtig ?.Begründen Sie, warum es jeweils keine anderen Graphen gibt, die die geforderte Bedingung erfüllen!

4. In der Druckerei Müller werden sechs unterschiedliche Farben der Reihe nach in einer Druckmaschine verwendet. Die Maschine muss vor jedem Farbenwechsel gereinigt werden, wobei die Reinigungszeit von den beiden aufeinander folgenden Farben abhängig ist. Die Druckerei möchte für den 6-Farbendruck eine Reihenfolge ermitteln, beginnend mit Blau, so dass die für die Reinigung benötigte Gesamtzeit minimal ist. Die Reinigungsdauern (in Minuten) sind in der nachfolgenden (symmetrischen) Tabelle angegeben. Zeichnen Sie die Tabelle als vollständigen bewerteten Graphen, wobei die Farben die Namen der Ecken sind und die Kanten mit den Reinigungszeiten bewertet werden. Führen Sie den Algorithmus der Einfügung der dichtesten Ecke für das Problem des Handlungsreisenden auf diesen vollständigen bewerteten Graphen (bis zum Abbruch!) durch.

Aufgaben

3.

Sorte	Blau	Schwarz	Gelb	Rot	Gold	Grün
Blau	0	3	12	10	10	6
Schwarz	3	0	17	15	15	20
Gelb	12	17	0	5	5	19
Rot	10	15	5	0	15	18
Gold	10	15	5	15	0	11
Grün	6	20	19	18	11	0