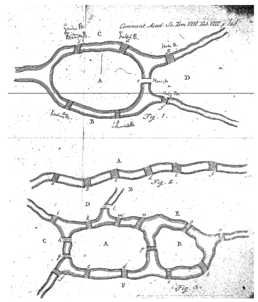
Graphentheoretische Konzepte und Algorithmen

Thema 06
Touren

Julia Padberg

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

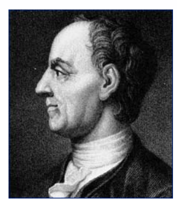


Aufgabenstellung Touren

befasst sich mit Reisen einer Person auf einem als Wegenetz aufgefaßten zusammenhängenden Graphen. Es wird dabei unterschieden, ob bei diesen Reisen alle Kanten oder alle Knoten genau einmal passiert werden müssen.

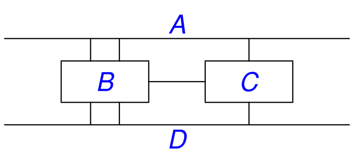
Das Königsberger Brückenproblem

Als Geburtsstunde der Graphentheorie gilt **Leonhard Eulers** Lösung des nachfolgenden Problems aus dem Jahre 1736.



Leonhard Euler (1707-1783)

Die Stadt Königsberg liegt auf beiden Ufern des Pregels, in dessen Mitte außerdem im Stadtgebiet noch zwei Inseln liegen. Diese vier Gebiete sind durch sieben Brücken miteinander verbunden:



Ist es möglich, von einem der vier Gebietsteile A, B, C oder D ausgehend, jede der Brücken genau einmal zu überqueren und sich am Ende dieser Überquerungen wieder am Ausgangspunkt einzufinden?

Definition (Eulertour und Eulerpfad)

- 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*.

Satz

- Ein ungerichteter Graph besitzt genau dann eine Eulertour, wenn jeder Knoten einen geraden Grad besitzt.
- Ein ungerichteter Graph besitzt genau dann einen Eulerpfad, wenn genau zwei Knoten einen ungeraden Grad besitzen. Diese beiden Knoten sind der erste und der letzte Knoten des Eulerpfads.

Beweisidee

Da bei einer Eulertour jeder Knoten und bei einem Eulerpfad jeder Knoten mit Ausnahme der ersten und letzten genauso oft erreicht wie verlassen werden muß, sind die angegebenen Bedingungen für die Existenz dieser Kantenfolgen notwendig. Daß sie dafür auch hinreichend sind, ergibt sich aus der weiter unten angegebenen Verfahrensvorschrift. q.e.d.

Praktische Ermittlung einer Eulertour

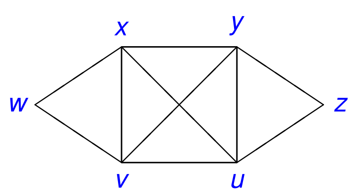
in einem Graphen ohne Knoten ungeraden Grades

1. Von einem beliebigen Knoten ausgehend wird der Graphen durchlaufen und jede passierte Kante wird entfernt bis der Ausgangsknoten wieder erreicht wird.
   Also liegt eine geschlossene Kantenfolge vor und an jedem Knoten dieser Kantenfolge wurde der Grad um eine gerade Zahl erniedrigt.
2. Falls diese Kantenfolge noch nicht alle Kanten enthält, werden die rstlichen Komponenten genauso durchlaufen.
   Es entsteht eine Menge von geschlossenen Kantenfolgen, die jede Kante genau einmal enthält.
3. Geschlossene Kantenfolgen mit einem gemeinsamen Knoten werden verschmolzen, indem beim Durchlaufen der einen Kantenfolge die andere „eingeschoben“ wird, sobald erstmals einer ihrer Knoten erreicht wird.
   Dann wird aus diesen geschlossenen Kantenfolgen eine Eulertour.

BSP

BSP:

Es sei folgender Graph gegeben:



Beim Durchlaufen der Kanten kann man etwa die geschlossenen Kantenfolgen erhalten: w, x, v, w; x, y, v, u, x; y, u, z, y die sich verschmelzen lassen zu $w, x, y, u, z, y, v, u, x, v, w$

Dem Zerfallen der Eulertour in verschiedene Kantenfolgen kann man entgehen, wenn man bei der Bildung der ersten Kantenfolge so weit wie möglich vermeidet, eine Schnittkante des jeweils verbliebenen Graphen auszuwählen.

Algorithmus von Fleury

Algorithmus

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

Schritt 1: Man wähle einen beliebigen Knoten v_0 in G und setze $W_0 = v_0$.

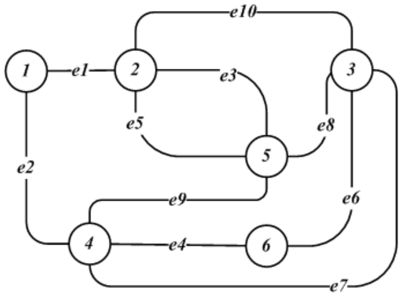
Schritt 2: Wenn der Kantenzug $W_i = v_0 e_1 v_1 \dots e_i v_i$ gewählt worden ist (so daß alle $e_1 \dots e_i$ unterschiedlich sind), wähle man eine von $e_1 \dots e_i$ verschiedene Kante e_{i+1}, so daß

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

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

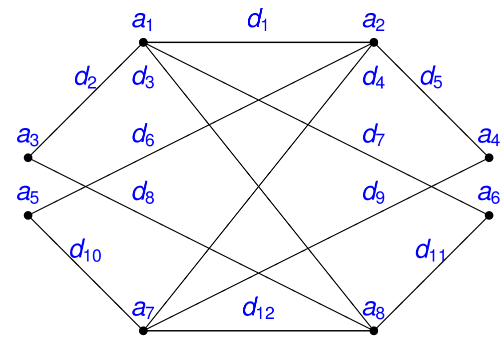
BSP: Algorithmus von Fleury

BSP:



e1–e10–e6–e4–e9–e5–e3–e8–e7–e2
Eulertour

Aufgabe 1:



Lösung

Eine mögliche Tour wäre (es sind hier nur die Kanten aufgeführt):

$$W_{12} = d_1 d_5 d_9 d_4 d_6 d_{10} d_{12} d_3 d_7 d_{11} d_8 d_2.$$

Zerlegung in in kantendisjunkte Kreise

Ein nicht leerer, zusammenhängender Graph G ist genau dann eulersch, wenn er in kantendisjunkte Kreise zerlegt werden kann, d.h. wenn man ihn als Vereinigung von kantendisjunkten Kreisen darstellen kann.

Beweis

$\Longrightarrow$ Ist $G = (V_G, E_G)$ eulersch, dann ist jeder Knotengrad $d(v)$ gerade. Damit besitzt G einen Kreis $C = (V_C, E_C)$, und in G' mit $E'_G = E_G - E_C$ sind wieder alle Knoten von geradem Grad. Die nicht-leeren Komponenten enthalten also wieder mindestens ein Kreis, der daraus gelöscht werden kann, usw.
Da der Graph endlich ist, gibt es eine Zerlegung von G in kantendisjunkte Kreise.

$\Longleftarrow$ Kann G in kantendisjunkte Kreise zerlegt werden, dann gehen durch einen Knoten v genau i kantendisjunkte Kreise, so gilt $d(v) = 2i$, also hat jeder Knoten eine geraden Grad. Da G zusammenhängend ist, ist auch Eulersch.

Algorithmus von Hierholzer

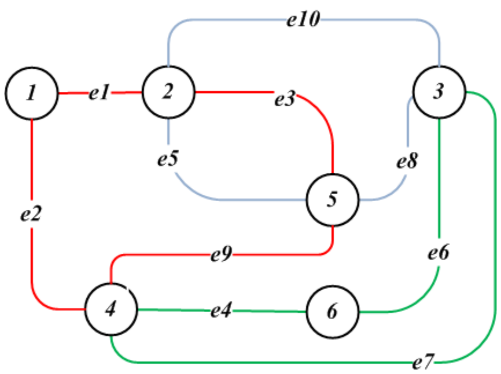
Algorithmus

Voraussetzung: Sei $G = (V, E)$ ein zusammenhängender Graph, der nur Knoten mit geradem Grad aufweist.

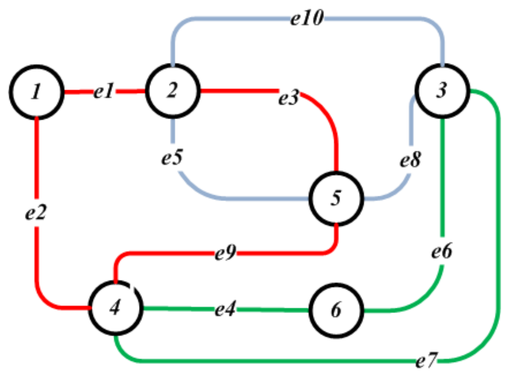
1. Wähle einen beliebigen Knoten v_0 des Graphen und konstruiere von v_0 ausgehend einen Kreis[a] K in G. Vernachlässige nun alle Kanten dieses Kreises.
2. Am ersten Knoten des ersten Kreises, dessen Grad größer 0 ist, lässt man nun einen weiteren Kreis entstehen. Erstelle so viele Kreise, bis alle Kanten von einem Kreis durchlaufen wurden.
3. Nun erhält man den Eulerkreis, indem man mit dem ersten Kreis beginnt und bei jedem Schnittpunkt mit einem anderen Kreis, den letzteren einfügt, und danach den nächsthöheren Kreis wieder bis zu einem weiteren Schnittpunkt oder dem Endpunkt fortsetzt.

[a] der keine Kante zweimal, aber ggf. Knoten mehrfach beinhaltet

BSP



Kreis 1: e1-e3-e9-e2
Kreis 2:e10-e8-e5
Kreis 3: e7-e4-e6



Eulerkreis:
e1-e10-e7-e4-e6-e8-e5-e3-e9-e2

Das Chinesische Briefträgerproblem

Der Name dieses Problems geht auf den Chinesen *Mei-ko Kwan* zurück, der es so formulierte:

Ein Postbote soll in seinem Zustellbezirk jede Straße (mindestens) einmal entlanggehen[1]. Insgesamt möchte er einen möglichst kurzen Weg zurücklegen (d.h. er möchte möglichst selten eine Straße zweimal durchlaufen müssen).
Wie soll er seine Tour durch den Zustellbezirk planen?

[1] Falls es in dem Zustellbezirk Straßen gibt, die nicht durchlaufen werden müssen, da an ihnen niemand wohnt, die aber durchlaufen werden dürfen, weil sie möglicherweise den Weg zwischen zwei bewohnten Gebieten verkürzen, wird die Aufgabenstellung als „Problem des Landbriefträgers“ bezeichnet. Dieses Problem ist schwieriger zu lösen als das vorliegende.

Graphentheoretische Formulierung

- Zustellbezirk als Graphen $G = (E, V)$
- in dem die Kanten E Straßen und die Knoten V Kreuzungen, Einmündungen und Endpunkte von Sackgassen darstellen.
- Jede Kante wird mit der Länge des entsprechenden Straßenabschnitts bewertet.
- Graphen in einem Zug durchlaufbar durch Verdoppeln der Kante, d.h. durch zwei parallele Kanten ersetzen
- Die Aufgabe ist,
  in diesem Graphen $G = (V, E)$ eine Kantenmenge mit minimaler Kantenbewertungssumme zu finden, die durch deren Verdoppelung der Graph eulersch wird.

Polynomialer Algorithmus von Edmonds

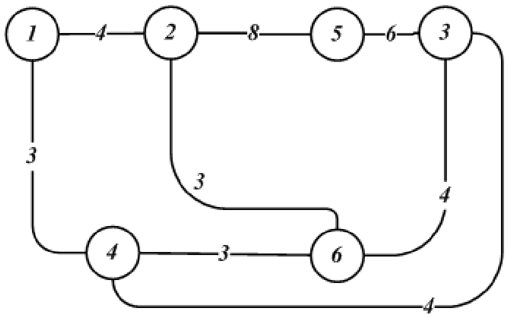
Algorithmus

Gegeben ein Graph G.

1. Finde die Menge U aller Knoten ungeraden Grades in G und bestimme für je zwei dieser Knoten u_i, u_j die Länge w_{ij} des kürzesten Weges zwischen ihnen.
2. Konstruiere den vollständigen Graphen $K_{|U|}$ mit Knotenmenge U und Kantenbewertungen $max - w_{ij}$, wobei max eine hinreichend große Zahl ist.
3. Bestimme in diesem vollständigen Graphen eine Paarung M mit maximaler Kantengewichtssumme.
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 einem beliebigen Knoten ausgehend „in einem Zug“ durchlaufen werden.

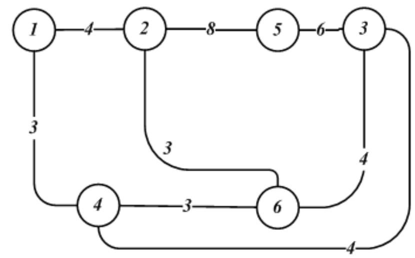
BSP

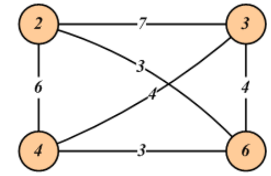
Gegeben ein Graph G.



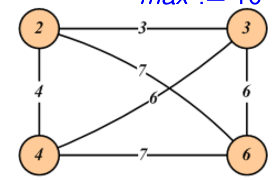
BSP

1. Finde die Menge U aller Knoten ungeraden Grades in G und bestimme für je zwei dieser Knoten u_i, u_j die Länge w_{ij} des kürzesten Weges zwischen ihnen.
2. Konstruiere den vollständigen Graphen $K_{|U|}$ mit Knotenmenge U und Kantenbewertungen $max - w_{ij}$, wobei max eine hinreichend große Zahl ist.



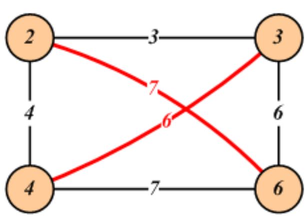


$max := 10$



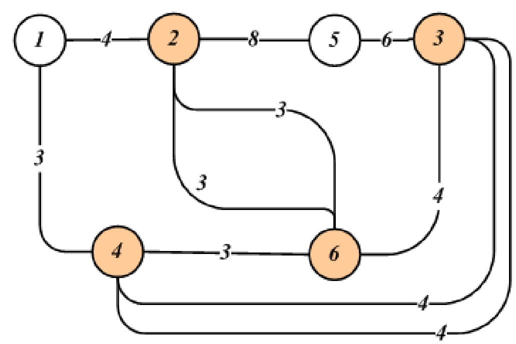
BSP

3. Bestimme in diesem vollständigen Graphen eine Paarung M mit maximaler Kantengewichtssumme.



BSP

4. Für jede Kante $u_i u_j \in M$ 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 einem beliebigen Knoten ausgehend „in einem Zug“ durchlaufen werden.



Hamiltonkreis

Definition

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

Hamiltonsche Graphen

Satz

Sei G ein schlichter Graph mit n Knoten für $3 \leq n \in \mathbb{N}^+$, und der Minimalgrad von G betrage $\delta(G) \geq \frac{n}{2}$, dann ist G hamiltonsch.

Definition

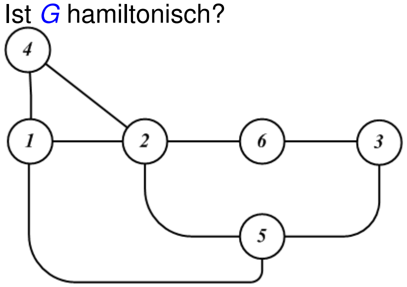
Gegeben sei ein schlichter Graph G_0 mit n Knoten. Solange es zwei nicht adjazente Knoten v_1, v_2 in G_i gibt, so daß $d(v_1) + d(v_2) \geq n$ in G_i ist, verbinde man diese beiden Knoten in einem neuen Obergraphen G_{i+1}. Der letzte Obergraph, der so erhalten wird, heißt **Hamiltonabschluss**(auch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

Satz

Ein schlichter Graph G ist dann und nur dann hamiltonsch, wenn seine Hülle $c(G)$ hamiltonsch ist.

Aufgabe 2:

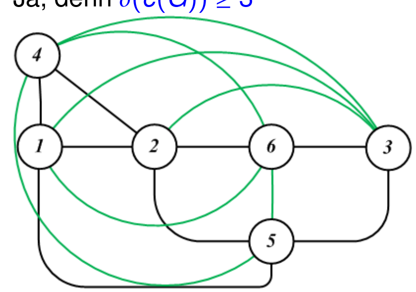
1. Warum ist die Länge jeder Rundreise mindestens so groß wie die Kantengewichtssumme eines Minimalgerüsts?
2. Wieviele Hamiltonische Kreise besitzt ein vollständiger, ungerichteter Graph mit n Knoten?
3. Ist G hamiltonisch?



Lösung von Aufgabe 2

Lösung

1. Jede Rundreise wird durch Weglassen einer beliebigen Kante zu einem Gerüst.
2. Es gibt $(n-1)!$ Hamiltonische Kreise.
3. Ja, denn $\delta(c(G)) \geq 3$



Finden von Hamiltonkreisen

- Ob ein Graph hamiltonsch ist, kann natürlich durch Ausprobieren aller Möglichkeiten („Brute_Force“) entschieden werden.
- Hat der Graph n Knoten, so ist die Anzahl der möglichen Hamiltonkreise nach oben beschränkt durch $(n-1)!$.

Komplexität des Hamiltonproblems

- HAMILTON ist NP-vollständig.
- Die EULER- bzw. HAMILTON-Probleme sind in der Formulierung fast identisch (Knotenwege statt Kantenwege), sie sind dennoch völlig unterschiedlich in ihrer inneren Komplexität.

Komplexität

Hintergründe für Phänomene dieser Art gibt die Komplexitätstheorie.
Die folgende Tabelle gibt einen Vorgeschmack auf die Konsequenzen unterschiedlicher Funktionen für den Rechenaufwand.

$log_2 n$	$n * log_2 n$	n	n^2	n^3	2^n	$n!$
3.3	33	10	100	1000	1024	$3.6 \cdot 10^6$
6.6	660	10^2	10^4	10^6	$1.3 \cdot 10^{30}$	$9,3 \cdot 10^{157}$
13.3	$1.3 \cdot 10^5$	10^4	10^8	10^{12}	$2 \cdot 10^{3010}$	$2.8 \cdot 10^{35659}$
20	$2 \cdot 10^7$	10^6	10^{12}	10^{18}	$9.9 \cdot 10^{301029}$	$< error >$

Problem des Handlungsreisenden

Traveling **S**alesman **P**roblem http://de.wikipedia.org/wiki/Problem_des_Handlungsreisenden

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 $w_{ij} \geq 0$. In welcher Reihenfolge soll er die Orte besuchen, damit die insgesamt zurückgelegte Entfernung minimal wird?

- Wahl einer Reihenfolge mehrere Orte für kürzeste Rundreisestrecke
- erste Erwähnung als mathematisches Problem im Jahre 1930
- Variante der Hamiltonschen Kreise

Anwendungen des TSP

- Tourenplanung
- Design von Mikrochips
- Verteilung von Waren
- Planung von Touren eines Kunden- oder Pannendienstes
- Genom-Sequenzierung

siehe z.B. http://www.tsp.gatech.edu/apps/genome.html

Modellierung

- Modellierung mit Hilfe eines Graphen
- Knoten repräsentieren Städte
- Kante (i, j) zwischen zwei Knoten i und j repräsentiert die Verbindung dazwischen
- Länge $c_{ij} \geq 0$ repräsentiert geographische Länge einer Verbindung, Reisezeit oder Kosten einer Reise.
- Tour ein Kreis, der jeden Knoten genau einmal enthält
- Ziel ist eine möglichst kurze Tour
- Vereinfachung: vollständiger Graph

Asymmetrisches und symmetrisches TSP

Asymmetrisches TSP

erlaubt Kanten, die in Hin- und Rückrichtung unterschiedliche Längen haben; wird mit Hilfe eines gerichteten Graphen modelliert.

Symmetrisches TSP

erfordert für alle Knotenpaare (i, j) identische Kantenlängen in beiden Richtungen; halbiert also die Anzahl der möglichen Touren und wird mit Hilfe eines ungerichteten Graphen modelliert.

Metrisches TSP

liegt vor, wenn zusätzlich die Kantenlängen die Dreiecksungleichung erfüllen; also die direkte Verbindung von i nach j ist nie länger als der Weg von i nach j über einen dritten Knoten k: $c_{ij} \leq c_{ik} + c_{kj}$

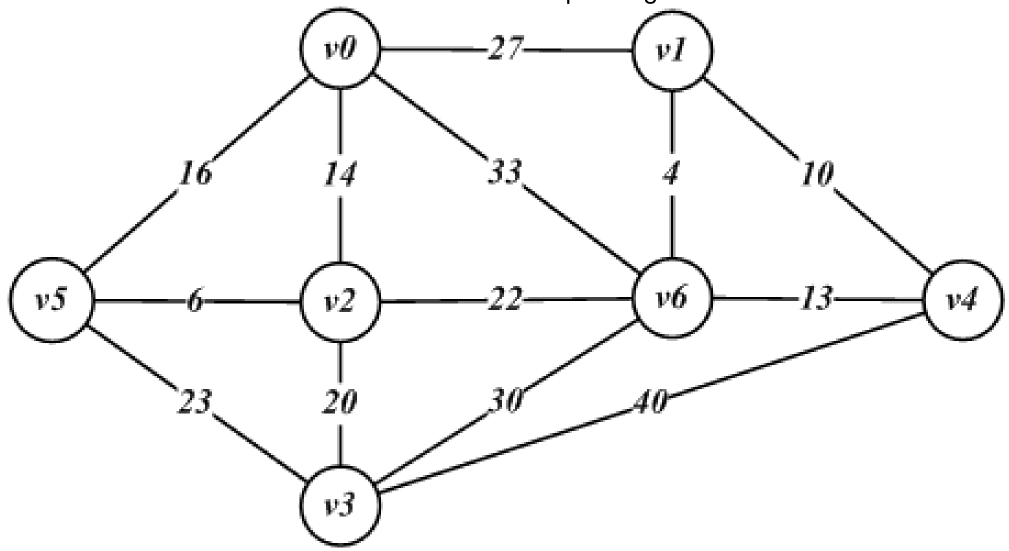
Minimales Gerüst zur Lösung eines TSP

Minimum-Spanning-Tree-Heuristik

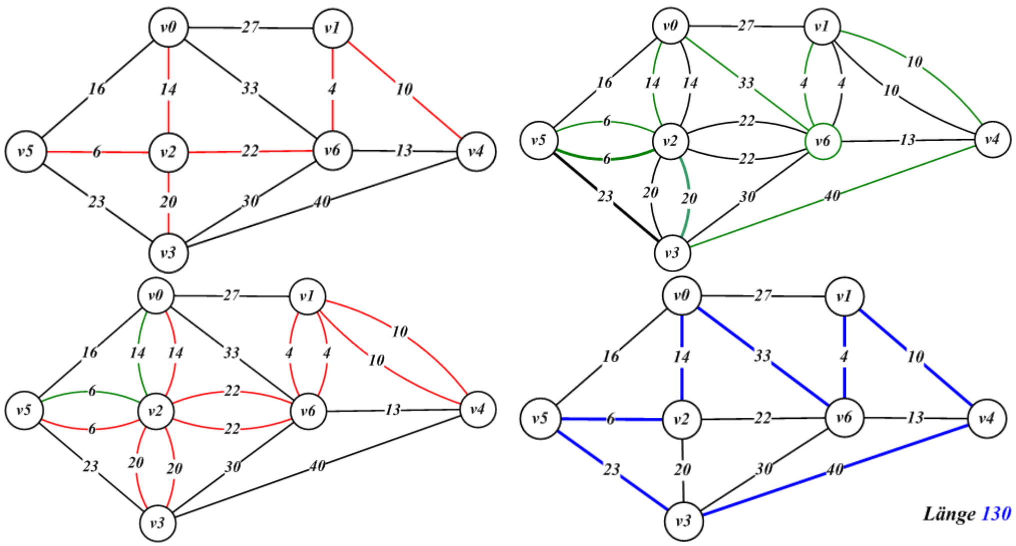
- Berechnung eines minimalen Gerüstes
- Konstruktion einer Tour, durch Verdopplung aller Baumkanten und dann Suche einer Eulertour.
- Abkürzung durch direkte Kanten, falls Knoten doppelt besucht werden falls metrisches TSP ohne Kontrolle, sonst mit.
- höchstens doppelt so lang ist wie eine kürzeste Tour.

Aufgabe 3:

Berechnen Sie bitte das TSP mit der Minimum-Spanning-Tree-Heuristik:



Lösung



Methode der Einführung des nächstgelegenen Knoten

nearest insertion algorithm

- gleiche Laufzeit ($O(n^2)$)
- und gleiche Qualität
  (Resultat besser als das doppelte vom optimalen Minimum)
  wie die Lösung mit Minimalgerüst
- in [KM99] *Methode der Einführung der dichtesten Ecke*
- Vorraussetzung
  - Entfernung eines Knoten v zu einen Menge von Knoten W in vollständigem Graph
    $d(v, W) = min\{l_{uv} \mid u \in W\}$
    also die kürzeste Kante von v zu einem Knoten in W
  - Ein Knoten v wird als **nächstgelegener** zu W bezeichnet, wenn für alle $x \in V \setminus W$ gilt
    $d(v, W) \leq d(x, W)$ gilt.

Algorithmus

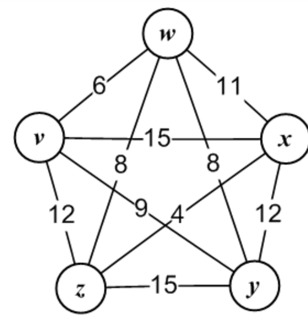
Gegeben sei ein Graph $K_n = (V, E)$ mit $V = \{v_1, v_2, \ldots v_n\}$. Der aktuelle gewählte Kreis W_i wird stets neu numeriert und ist gegeben durch $W_i = u_1 u_2 \ldots u_i u_1$. Diese neue Numerierung verändert nicht die Reihenfolge der gewählten Knoten v_i!

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

Schritt 2: Aus der Zahl der $n - i$ Knoten, die bisher noch nicht gewählt worden sind, ermittle man einen Knoten $u_{i+1} = v_k$, die am dichtesten zu W_i liegt. Sei $W_i = u_1 u_2 \ldots u_i u_1$. Man bestimme dann, welche der Kantenfolgen (Kreise) $u_1 u_{i+1} u_2 u_3 \ldots u_i u_1$, $u_1 u_2 u_{i+1} u_3 \ldots u_i u_1$, $\ldots$ $u_1 u_2 u_3 \ldots u_i u_{i+1} u_1$ die kürzeste ist. Es sei W_{i+1} die kürzeste Kantenfolge. Man kennzeichne sie, wenn nötig, neu als $u_1 \ldots u_{i+1} u_1$. Man setze $i := i + 1$.

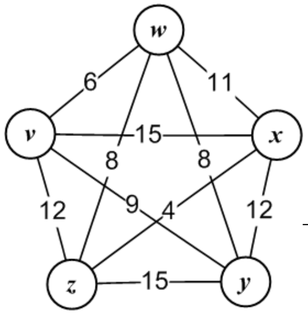
Schritt 3: Wenn W_i alle Knoten beinhaltet, beende den Algorithmus, sonst führe Schritt 2 aus.

BSP



Schritt	Ablauf
1	$u_1 := v$
2	$u_2 := w$, so daß $W_2 = vwv = 12$
3	$u_3 := y$, so daß $W_3 = vywv = 23$
4	$u_4 := z$. Folgende Zyklen werden betrachtet:
	$u_1 u_4 u_2 u_3 u_1 = vzywv = 41$
	$u_1 u_2 u_4 u_3 u_1 = vyzwv = 38$
	$u_1 u_2 u_3 u_4 u_1 = vywzy = 37$
	Wähle $W_4 = vywzy := u_1 u_2 u_3 u_4 u_1$

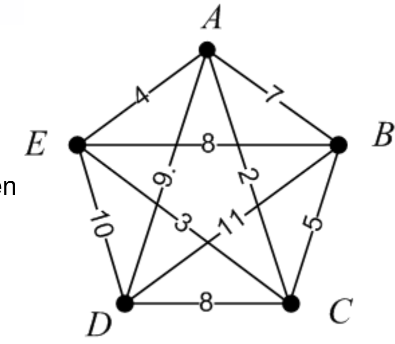
BSP (ff)



Schritt	Ablauf
5	$u_5 := x$. Folgende Zyklen werden betrachtet:
	$u_1 u_5 u_2 u_3 u_4 u_1 = vxywzv = 55$
	$u_1 u_2 u_5 u_3 u_4 u_1 = vyxwzv = 52$
	$u_1 u_2 u_3 u_5 u_4 u_1 = vywxzv = 54$
	$u_1 u_2 u_3 u_4 u_5 u_1 = vywzxv = 54$
	Wähle $W_5 = vyxwzv := u_1 u_2 u_3 u_4 u_5 u_1$

Aufgabe 4:

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.

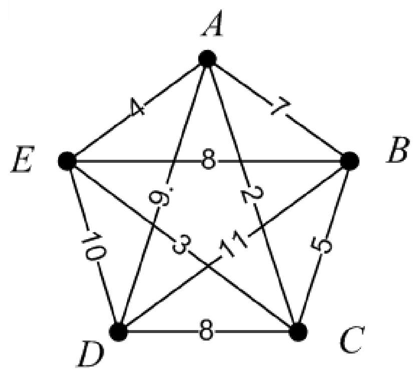


Schritt 1: Man wähle eine beliebige Knoten $u_1 = v_j \in V$ als Startknoten und setze $W_1 = u_1$.

Schritt 2: Aus der Zahl der $n - i$ Knoten, die bisher noch nicht gewählt worden sind, ermittle man einen Knoten $u_{i+1} = v_k$, die am dichtesten zu W_i liegt. Sei $W_i = u_1 u_2 \dots u_i u_1$. Man bestimme dann, welche der Kantenfolgen (Kreise) $u_1 u_{i+1} u_2 u_3 \dots u_i u_1$, $u_1 u_2 u_{i+1} u_3 \dots u_i u_1$, $\dots$ $u_1 u_2 u_3 \dots u_i u_{i+1} u_1$ die kürzeste ist. Es sei W_{i+1} die kürzeste Kantenfolge. Man kennzeichne sie, wenn nötig, neu als $u_1 \dots u_{i+1} u_1$. Man setze $i := i + 1$.

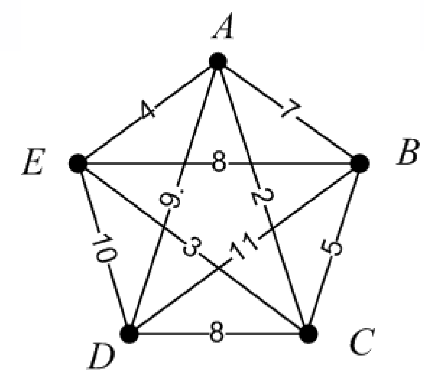
Schritt 3: Wenn W_i alle Knoten beinhaltet, beende den Algorithmus, sonst führe Schritt 2 aus.

Lösung von Aufgabe 4



Schritt	Ablauf
1	$u_1 := A$
2	$u_2 := C$, so daß $W_2 = ACA = 4$
3	$u_3 := E$, so daß $W_3 = AECA = 9$
4	$u_4 := B$. Folgende Zyklen werden betrachtet:
	$u_1 u_4 u_2 u_3 u_1 = ABECA = 20$
	$u_1 u_2 u_4 u_3 u_1 = AEBCA = 19$
	$u_1 u_2 u_3 u_4 u_1 = AECBA = 19$
	Wähle $W_4 = AEBCA := u_1 u_2 u_3 u_4 u_1$

Lösung von Aufgabe 4



Schritt	Ablauf
5	$u_5 := D$. Folgende Zyklen werden betrachtet:
	$u_1 u_5 u_2 u_3 u_4 u_1 = ADEBCA = 34$
	$u_1 u_2 u_5 u_3 u_4 u_1 = AEDBCA = 32$
	$u_1 u_2 u_3 u_5 u_4 u_1 = AEBDCA = 33$
	$u_1 u_2 u_3 u_4 u_5 u_1 = AEBCDA = 34$
	Wähle $W_5 = AEDBCA := u_1 u_2 u_3 u_4 u_5 u_1$

Damit ist der ziemlich kurze Kreis $W_5 = AEDBCA$ mit den Kosten von 32 gefunden.

Aufgabe 5:

Zeigen Sie, daß das Hamiltonkreisproblem in einem vorgegebenen ungerichteten Graphen $G(V, E)$ ein Spezialfall des TSP für vollständige Graphen ist.

Lösung

Sie vervollständigen G durch Einfügen weiterer Kanten in den vollständigen Graphen $K_{|V|}$, für alle Kanten $v_i v_j \in E$ setzen $w_{ij} := 1$ und für alle übrigen Kanten $w_{ij} := 2$ und dann für den so bewerteten vollständigen Graphen das TSP lösen.

Lösung

Falls das TSP eine Lösung mit der Länge $|V|$ besitzt, hat der ursprüngliche Graph einen Hamiltonkreis, sonst nicht.

Aufgabe 6:

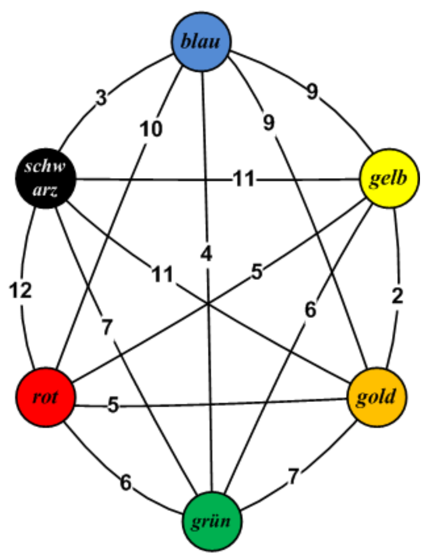
In der Druckerei Müller werden sechs unterschiedliche Farben der Reihe nach in einer Druckmaschine verwendet. Die Maschine muß vor jedem Farbenwechsel gereinigt werden, wobei die Reinigungszeit von den beiden aufeinanderfolgenden Farben abhängig ist. Die Druckerei möchte für den 6-Farbendruck eine Reihenfolge ermitteln, beginnend mit Blau, so daß die für die Reinigung benötigte Gesamtzeit minimal ist.
Die Reinigungsdauern (in Minuten) sind in der folgenden (symmetrischen) Tabelle angegeben:

Sorte	Blau	Schwarz	Gelb	Rot	Gold	Grün
Blau	0	3	9	10	9	4
Schwarz	3	0	11	12	11	7
Gelb	9	11	0	5	2	6
Rot	10	12	5	0	4	8
Gold	9	11	2	4	0	7
Grün	4	7	6	8	7	0

Bitte erläutern Sie Ihr Lösungsverfahren zur Minimierung der Reinigungszeit und führen Sie es durch.

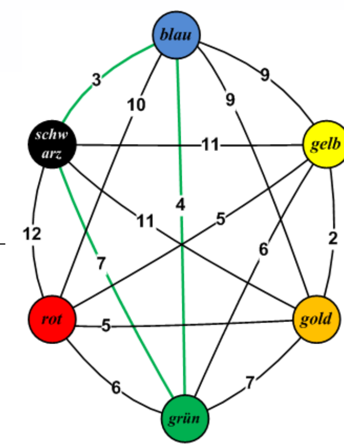
Lösung von Aufgabe 6

Zeichnen Sie die Tabelle als vollständigen bewerteten Graphen, wobei die Farben die Namen der Knoten sind und die Kanten mit den Reinigungszeiten bewertet werden.
Führen Sie einen Algorithmus für das Problem des Handlungsreisenden auf diesen vollständigen bewerteten Graphen durch.



Lösung von Aufgabe 6

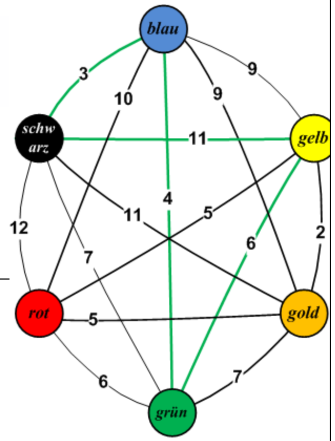
Nächstgelegener Knoten Algorithmus



Schritt	Ablauf
1	$u_1 :=$ blau
2	$u_2 :=$ schwarz,
	so daß $W_2 =$ blau schwarz blau $= 6$
3	$u_3 :=$ grün,
	so daß $W_3 =$ blau grün schwarz blau $= 14$
4	$u_4 :=$ gelb. Folgende Zyklen werden betrachtet:
	$u_1 u_4 u_2 u_3 u_1 =$ blau gelb grün schwarz blau $= 25$
	$u_1 u_2 u_4 u_3 u_1 =$ blau grün gelb schwarz blau $= 24$
	$u_1 u_2 u_3 u_4 u_1 =$ blau grün schwarz gelb blau $= 31$
	Wähle $W_4 =$blau grün gelb schwarz blau$:= u_1 u_2 u_3 u_4 u_1$

Lösung von Aufgabe 6

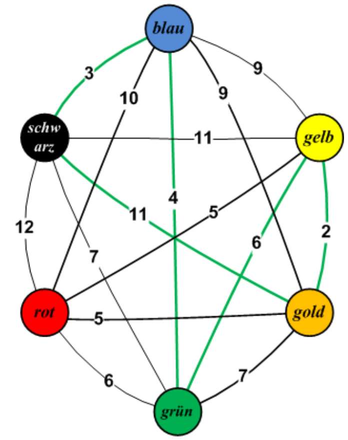
Nächstgelegener Knoten Algorithmus



Schritt	Ablauf
5	u_5 :=gold. Folgende Zyklen werden betrachtet: $u_1 u_5 u_2 u_3 u_4 u_1 =$ blau gold grün gelb schwarz blau $= 36$ $u_1 u_2 u_5 u_3 u_4 u_1 =$ blau grün gold gelb schwarz blau $= 27$ $u_1 u_2 u_3 u_5 u_4 u_1 =$ blau grün gelb gold schwarz blau $= 26$ $u_1 u_2 u_3 u_4 u_5 u_1 =$ blau grün gelb schwarz gold blau$= 41$ Wähle W_5 =blau grün gelb gold schwarz blau $:= u_1 u_2 u_3 u_4 u_5 u_1$

Lösung von Aufgabe 6

Nächstgelegener Knoten Algorithmus



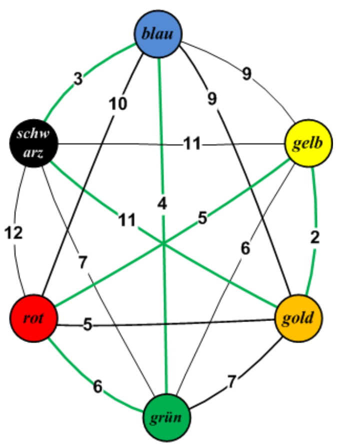
6	u_6 :=rot. Folgende Zyklen werden betrachtet: $u_1 u_6 u_2 u_3 u_4 u_5 u_1 =$ blau rot grün gelb gold schwarz blau$= 38$ $u_1 u_2 u_6 u_3 u_4 u_5 u_1 =$ blau grün rot gelb gold schwarz blau $= 31$ $u_1 u_2 u_3 u_6 u_4 u_5 u_1 =$ blau grün gelb rot gold schwarz blau$= 33$ $u_1 u_2 u_3 u_4 u_6 u_5 u_1 =$ blau grün gelb gold rot schwarz blau $= 32$ $u_1 u_2 u_3 u_4 u_5 u_6 u_1 =$ blau grün gelb gold schwarz rot blau $= 45$ Wähle $W_6 =$ blau grün rot gelb gold schwarz blau$:= u_1 u_2 u_3 u_4 u_5 u_6 u_1$

Lösung von Aufgabe 6

Nächstgelegener Knoten Algorithmus

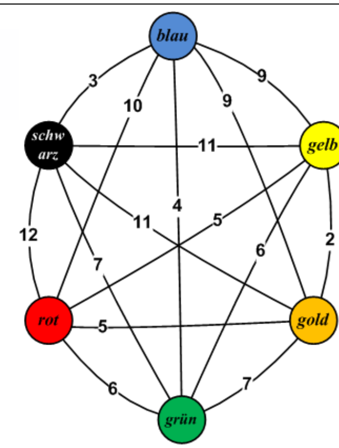
Ergebnis:

$W_6 =$ blau grün rot gelb gold schwarz blau $= 31$

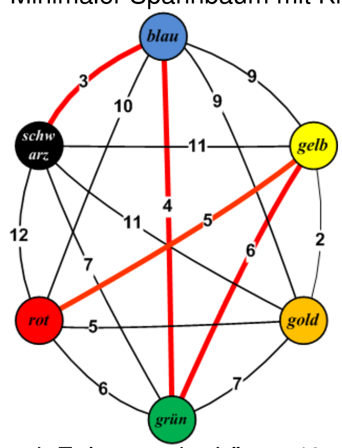


Lösung von Aufgabe 6

Minimaler-Spannbaum-Heuristik



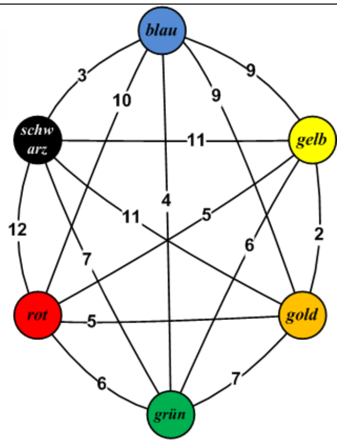
Minimaler Spannbaum mit Kruskal



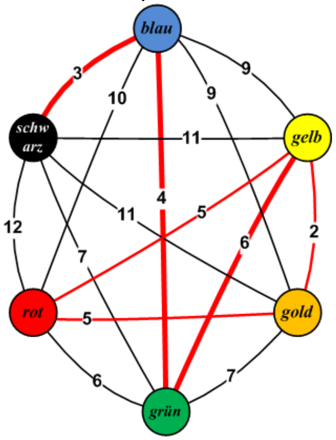
mit Eulertour der Länge 40:
blau-schwarz-blau-grün-gelb-gold-gelb-rot-gelb-grün-blau

Lösung von Aufgabe 6

Minimaler-Spannbaum-Heuristik



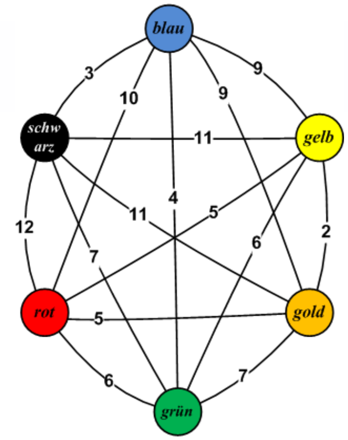
Minimaler Spannbaum mit Kruskal



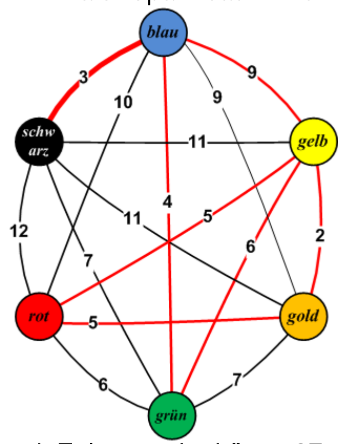
mit Eulertour der Länge 38:
blau-schwarz-blau-grün-gelb-gold-rot-gelb-grün-blau

Lösung von Aufgabe 6

Minimaler-Spannbaum-Heuristik



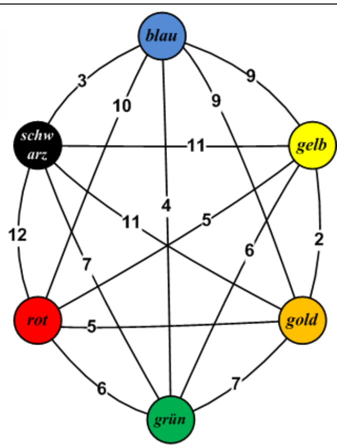
Minimaler Spannbaum mit Kruskal



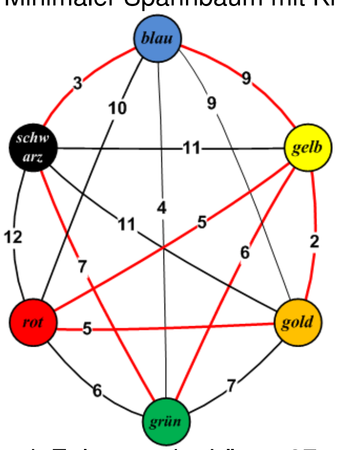
mit Eulertour der Länge 37:
blau-schwarz-blau-grün-gelb-gold-rot-gelb-blau

Lösung von Aufgabe 6

Minimaler-Spannbaum-Heuristik



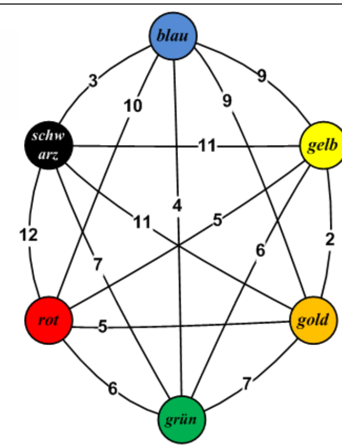
Minimaler Spannbaum mit Kruskal



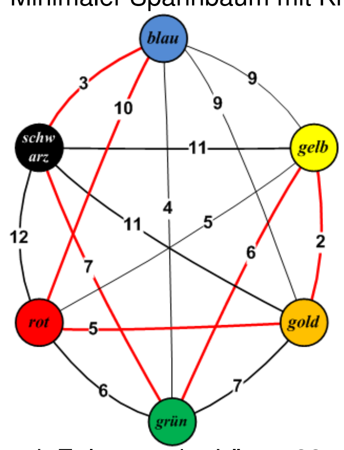
mit Eulertour der Länge 37:
blau-schwarz-grün-gelb-gold-rot-gelb-blau

Lösung von Aufgabe 6

Minimaler-Spannbaum-Heuristik



Minimaler Spannbaum mit Kruskal

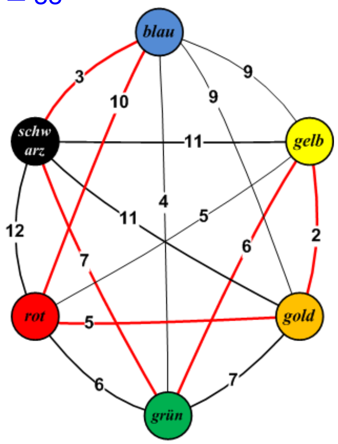


mit Eulertour der Länge 33:
blau-schwarz-grün-gelb-gold-rot-blau

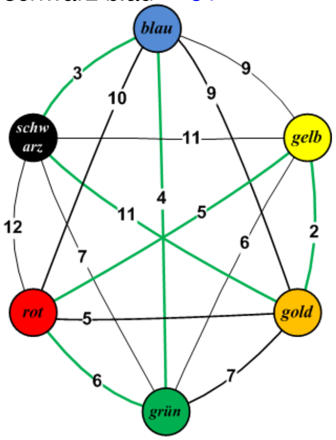
Lösung von Aufgabe 6

Ergebnisse

Minimaler Spannbaum mit Kruskal
blau-schwarz-grün-gelb-gold-rot-blau
$= 33$



Nächstgelegener Knoten Algorithmus
blau grün rot gelb gold schwarz blau $= 31$



Besserer ????