

## 1 Hamiltonsche Kreise

a) Ein vollständiger bipartiter Graph  $K_{n,m}$  ist nur genau dann ein Baum, wenn  $m = 1$  oder  $n = 1$ . Wenn nämlich  $n = 1$  ist, dann beträgt die Anzahl der Knoten  $V$   $m + 1$ ; die Anzahl der Kanten  $E$  beläuft sich jedoch nur auf  $m$ . Da  $|E| = |V| - 1$  eine Charakterisierung von Bäumen ist, liegt hier folglich ein Baum vor.

Der vollständige bipartite Graph enthält einen Hamiltonschen Kreis genau dann wenn  $n = m$  ist. Dies gilt, da man ja nur von der Knotengruppe A zur Knotengruppe B kommt, und wenn z.B. weniger Knoten in B als in A wären, man nicht mehr zum Ursprungspunkt zurückkäme, ohne einen bereits besuchten Knoten in A zu benutzen.

b) In einem Hamiltonschen  $K_{n,m}$  ist wie in a) festgestellt  $n = m$ ; es gilt also  $|V| = 2n$  und  $|E| = n^2$ . Um die Anzahl der verschiedenen Kreise  $A$  zu bestimmen wählt man sich einen Startknoten für den Kreis, dafür gibt es  $2n$  Möglichkeiten, und nun hat man auf Grund der Vollständigkeit des bipartiten Graphen  $n$  Möglichkeiten, zum nächsten Knoten zu gehen. Von hieraus können noch  $n - 1$  neue Knoten erreicht werden usw. Es gilt also  $2n * n * (n - 1) * (n - 1) * \dots * 1 = 2n((n - 1)!)^2$ . Da es sich um einen Kreis handelt, erhalten wir für jeden Knoten gleiche Bilder, darum muss noch durch  $2n$  geteilt werden.

$$A = \frac{2n((n - 1)!)^2}{2n} = ((n - 1)!)^2$$

## 2 Bäume und kürzeste Wege

a) In einem  $q$ -ären Baum setzt sich die Anzahl der Knoten  $n$  aus der Anzahl der Blätter  $l$  und der der inneren Knoten  $i$  zusammen;  $n = l + i$

Des weiteren gilt  $n = q*i + 1$ , da  $q*i$  die Anzahl der Kanten beschreibt und die Anzahl der Knoten nach der Definition eines Baumes folglich um eins größer sein muss.

Aus diesen beiden Gleichungen können nun Formeln für  $i$  und  $l$  abgeleitet werden:

$$n = q * i + 1$$

$$n = l + i$$

$\Rightarrow i = n - l$  Einsetzen in die andere Gleichung:

$$n = q * (n - l) + 1 \text{ Durch Umstellen erhält man}$$

$$l = n - \frac{n - 1}{q}$$

$$i = n - \left(n - \frac{n - 1}{q}\right)$$

$$i = \frac{n - 1}{q}$$

Die Anzahl der Blätter kann also mit

$$l = n - \frac{n - 1}{q}$$

berechnet werden. Die Anzahl der inneren Knoten ergibt sich dann als

$$i = \frac{n - 1}{q} .$$

**b)** Ein Wald  $F$  mit  $n$  Knoten kann minimal 0 Kanten besitzen, in diesem Fall bildet jeder der  $n$  Knoten seine eigene Zusammenhangskomponente, seinen eigenen Baum, da  $|E| = |V| - 1$  gilt.

Die maximale Anzahl an Kanten erhält man in einem Wald, wenn dieser aus nur zwei Bäumen besteht und wenn einer davon aus nur einem einzelnen Knoten besteht. Der andere Baum hätte dann  $(n - 1) - 1 = n - 2$  Kanten, laut Definition  $|E| = |V| - 1$ .

**c)** Alleine wegen der Tatsache, dass ein ungerichteter und zugleich zusammenhängender Graph vorliegt, kann von einem Knoten  $x$  über  $a$  der Knoten  $y$  erreicht werden. Konkret bedeutet ungerichtet nichts anderes, als dass man in beide Richtungen die Kante benutzen kann und zusammenhängend, dass man von jedem Knoten jeden anderen erreichen kann. Und sobald es Wege zwischen zwei Punkten gibt, gibt es auch einen kürzesten.

Algorithmisch könnte der kürzeste Weg wie folgt berechnet werden:

Suche kürzesten Weg von  $x \rightarrow a$

Suche kürzesten Weg von  $a \rightarrow y$

Setze kürzeste Wege zusammen  $x \rightarrow y$

Gib  $x \rightarrow y$  aus

### 3 Totale Senke finden

Öhm ...

### 4 BFS, DFS

Als Erstes wird BFS ausgeführt. Die hier verwendete Datenstruktur ist eine Warteschlange  $Q$ , in der alle neu entdeckten Knoten hinten angefügt werden und am Kopf der Warteschlange gearbeitet wird.

Im ersten Schritt befindet sich nur der Startknoten 1 in  $Q$ .

Man entdeckt von dort Knoten 2 und 6, 1 wird gestrichen  $\rightarrow Q: 2, 6$

Von 2 wird Knoten 4 und 5 entdeckt, 2 wird gestrichen  $\rightarrow Q: 6, 4, 5$

Von 6 wird nur Knoten 7 neu entdeckt, 6 wird gestrichen  $\rightarrow Q: 4, 5, 7$

Von 4 wird kein Knoten neu entdeckt, 4 wird gestrichen  $\rightarrow Q: 5, 7$

Von 5 wird kein Knoten neu entdeckt, 5 wird gestrichen  $\rightarrow Q: 7$

Von 7 wird nur Knoten 8 neu entdeckt, 7 wird gestrichen  $\rightarrow Q: 8$

Von 8 wird nur Knoten 3 neu entdeckt, 8 wird gestrichen  $\rightarrow Q: 3$

Von 3 wird kein Knoten neu entdeckt, 3 wird gestrichen  $\rightarrow Q$ : leer, d.h. der Algorithmus wird beendet.

Die Adjazenzliste sieht wie folgt aus:

1:	[2,6]
2:	[4,5]
3:	[]
4:	[]
5:	[]
6:	[7]
7:	[8]
8:	[3]

Nun wird die Tiefensuche angewendet. Hier ist die Datenstruktur ein Kellerspeicher  $K$ , d.h. alle neu entdeckten Knoten kommen oben auf und der oberste wird zuerst bearbeitet.

Im ersten Schritt befindet sich nur der Startknoten 1 in  $K$ .

Von 1 wird Knoten 2 neu entdeckt  $\rightarrow K: 1, 2$

Von 2 wird Knoten 4 neu entdeckt  $\rightarrow K: 1, 2, 4$

Von 4 wird Knoten 5 neu entdeckt  $\rightarrow K: 1, 2, 4, 5$

Von 5 wird kein Knoten neu entdeckt, 5 wird gestrichen  $\rightarrow K: 1, 2, 4$  Von 4 wird kein Knoten neu entdeckt, 4 wird gestrichen  $\rightarrow K: 1, 2$  Von 2 wird kein Knoten neu entdeckt, 2 wird gestrichen  $\rightarrow K: 1$  Von 1 wird Knoten 6 neu entdeckt  $\rightarrow K: 1, 6$

Von 6 wird Knoten 7 neu entdeckt  $\rightarrow K: 1, 6, 7$

Von 7 wird Knoten 8 neu entdeckt  $\rightarrow K: 1, 6, 7, 8$

Von 8 wird Knoten 3 neu entdeckt  $\rightarrow K: 1, 6, 7, 8, 3$

Von 3 wird kein Knoten neu entdeckt, 3 wird gestrichen  $\rightarrow K: 1, 6, 7, 8$

Von 8 wird kein Knoten neu entdeckt, 8 wird gestrichen  $\rightarrow K: 1, 6, 7$

Von 7 wird kein Knoten neu entdeckt, 7 wird gestrichen  $\rightarrow K: 1, 6$

Von 6 wird kein Knoten neu entdeckt, 6 wird gestrichen  $\rightarrow K: 1$

Von 1 wird kein Knoten neu entdeckt, 1 wird gestrichen  $\rightarrow K: \text{leer}$ , d.h. die Tiefensuche ist abgeschlossen. Die Adjazenzliste sieht wie folgt aus:

1:	[2,6]
2:	[4]
3:	[]
4:	[5]
5:	[]
6:	[7]
7:	[8]
8:	[3]

Es gibt auch eine topologische Sortierung, die sich aus der sortierten Knotenfolge des DFS ergibt: 1, 6, 7, 8, 3, 2, 4, 5

## 5 Würfel

Als erste Überlegung bildet man die Summe der Kantengewichte:  $\sum_{i=1}^{12} i = 78$ . Da jede Kante ja zwischen zwei Knoten liegt, und somit doppelt gezählt wird, kann  $(2 * 78) : 8 = 19,5$  als insgesamtes Gewicht der Kanten eines Knotens angesehen werden. Da dieses Ergebnis keine gerade Zahl ist, und somit nicht mit den zu Verfügung stehenden Gewichten realisiert werden kann, ist logischerweise die Aufgabe nicht lösbar.