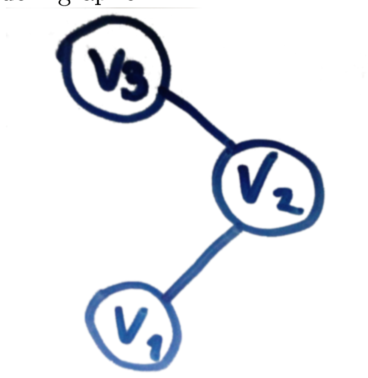
DAP2 UB12
Anja Rey, Gr.23 , Briefkasten 22

Max Springenberg, 177792

12.1 Graphen

12.1.1

Der Induktionsversuch ist falsch, da beispielsweise fuer n = 3 mit dem folgendem graphen:



Beim entfernen des Knotens v_2 keine Kanten mehr vorhanden sind.
Somit ist der Induktionsschritt nicht fuer alle Knoten geeignet.

12.1.2

Die Formel:

$$\frac{'Knotenzahl' * ('Knotenzahl' - 1)}{2} = \frac{|V| * (|V| - 1)}{2} = \frac{n * (n - 1)}{2}$$

gibt die maximale Kantenzahl an.
Ein Knoten kann mit den restlichen n-1 Knoten Kanten bilden. Jedoch gilt, dass die haelfte der Kanten doppelt gezaehlt werden, wenn man bedenkt, dass andere Knoten auch mit dem Knoten selbst wieder eine Kante bilden koennen. Deshalb muss durch 2 dividiert werden.

12.1.3

Die Maximale Anzahl an Kanten kann mit der Formel aus 12.1.2 berechnet werden. Wir waehlen nun die minimale Anzahl loser Knoten und die maximale Anzahl an Kanten fuer die verbleibenden Knoten.

minimale lose Knoten	1
Anzahl verbleibender Knoten	n - 1

wir setzen nun also fuer n, n - 1 in die Formel ein. Daraus resultiert die maximale Kantenzahl: $\frac{(n-1)*(n-2)}{2}$

12.2 Algorithmus von Dijkstra

12.2.1

Im folgendem werden die Resultate nach jedem Durchlauf dL_i der While-Schleife angegeben.

dL_0	a	b	c	d	e
d	0	∞	∞	∞	∞
color	w	w	w	w	w

$u = a$

$Q = \{(a, 1), (b, \infty), (c, \infty), (d, \infty), (e, \infty)\}$

---

dL_1	a	b	c	d	e
d	0	4	∞	1	12
color	s	w	w	s	w

$u = d$

$Q = \{(d, 1), (b, 4), (e, 12), (c, \infty)\}$

---

dL_2	a	b	c	d	e
d	0	4	14	1	2
color	s	w	w	s	s

$u = e$

$Q = \{(b, 4), (d, 14)\}$

---

dL_3	a	b	c	d	e
d	0	4	6	1	2
color	s	s	w	s	s

$u = b$

$Q = \{(c, 4)\}$

---

dL_4	a	b	c	d	e
d	0	4	6	1	2
color	s	s	s	s	s

$u = c$

$Q = \emptyset$

12.2.2

Die Heap-Eigenschaft eines AVL-Baumes wieder her zu stellen benoetigt O(log(n)), wir muessen unsere Liste immer wieder neu sortieren und landen damit in O(n*log(n)), unter Verwendung eines angepassten merge-sort Algorithmus.