18.11.09

Datenstrukturen:

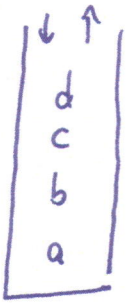
(B) Warteschlange oder „Queue", bzw.

„FIFO"-Regel: First In, First Out

Eingang → _ e d c b a, → Ausgang

(D) Stapel oder „Stack" (auch „Keller"), bzw.

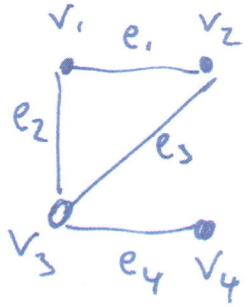
„LIFO"-Regel: Last In, First Out



3.4 Datenstrukturen für Graphen

Wie beschreibt man einen Graphen?

(1) Adjazenzmatrix



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

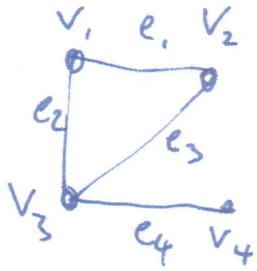
(„adjazent": verbunden)

Also: $A \in \{0,1\}^{n \times n}$

mit $a_{v,w} := \begin{cases} 1 & \text{für } \{v,w\} \in E \\ 0 & \text{sonst} \end{cases}$

Größe: n^2 für einen Graphen mit n Knoten.

(2) Inzidenzmatrix



$$\begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \begin{array}{c} \begin{array}{cccc} e_1 & e_2 & e_3 & e_4 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

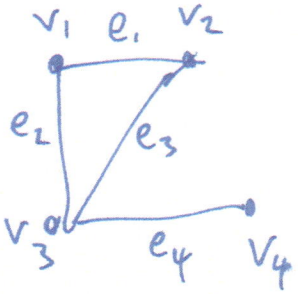
(„inzident": zusammentreffend)

Also: $A \in \{0,1\}^{n \times m}$

mit $a_{v,e} := \begin{cases} 1 & \text{für } v \in e \\ 0 & \text{sonst} \end{cases}$

Größe: nm für einen Graphen mit n Knoten, m Kanten. (i.d.R. viele Nullen!)

(3) Kantenliste:

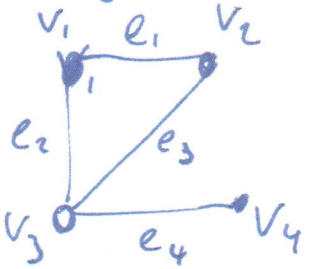


$\{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\}, \{v_3, v_4\}$

Benötigt wird Kantennummerierung!

Also $\log n$ für jeden Index,
insgesamt Platz in der Größenordnung von $m \log n$ → später mehr!

(4) Adjazenzliste:



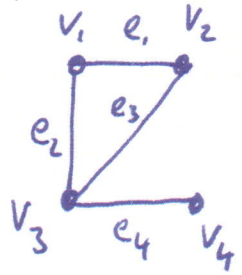
v_1 : v_2, v_3
v_2 : v_1, v_3
v_3 : v_1, v_2, v_3
v_4 : v_4

Benötigt etwas mehr Platz als die Kantenliste, ist aber u.U. praktischer im Kontext von Graphenalgorithmen (wo man z.B. Nachbarn für bestimmte Knoten sucht, die man nicht erst mühsam aus einer Liste herraussuchen will!)

Zugriff auf die einzelnen Teillisten mit n Pointern, benötigt $O(n \log m)$ zusätzliche Bits.

→ Insgesamt $O(n \log m + m \log n) = O(m \log n)$

(3) Kantenliste:



$$\{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\}, \{v_3, v_4\}$$

Benötigt wird Kantennummerierung!

↳ Jeder Index ist eine Binärzahl mit $\log_2 n$ Bits, oder eine Dezimalzahl mit $\log_{10} n$ Stellen.

Unterschied: Ein Faktor 3,3219...,

denn $\log_2 n = \log_2 10 \cdot \log_{10} n$

und $\log_2 10 = 3{,}3219\ldots$!

Für einen Graphen mit m Kanten und n Knoten

ergibt sich in obiger Schreibweise dezimal ein Platzbedarf von

$$(6m-1) + 2m \log_{10} n.$$

Dabei kann man an ein paar Stellen sparen (z.B. "{v" weglassen, "{" und "}" weglassen, "," durch Leerzeichen ersetzen),

andererseits kann man es auch teurer machen (z.B. "v" in ASCII codieren) oder anders codieren, etwa binär. So kann man auch

$$2m-1 + 2m \log_2 n$$ erhalten.