6.11 Darstellung von Graphen im Computer

Wie sollen Graphen im Computer gespeichert werden?
Anforderungen:

- Adjazenz und Inzidenz sollen schnell getestet werden können
- Speicherbedarf soll begrenzt bleiben

Definition 6.1. Es sei $G = (V, E, \Psi)$ ein Graph mit n Knoten und m Kanten. Die Matrix $A = (a_{x,y})_{x,y \in V} \in \mathbb{Z}^{n \times n}$ heißt Adjazentmatrix von G, wobei

$$a_{x,y} = |\{e \in E \mid \Psi(e) = \{x, y\} \text{ bzw. } \Psi(e) = (x, y)\}|$$

Definition 6.2. Es sei $G = (V, E, \Psi)$ ein gerichteter Graph mit n Knoten und m Kanten. Die Matrix $A = (a_{x,e})_{x \in V, e \in E} \in \mathbb{Z}^{n \times m}$ heißt Inzidenzmatrix von G, wobei

$$a_{x,e} = \begin{cases} -1, & \text{falls } e \in \delta^+(x) \\ 1, & \text{falls } e \in \delta^-(x) \\ 0, & \text{sonst.} \end{cases}$$

Nachteil von Adjazenz- und Inzidenzmatrizen:

- Graph mit n Knoten und m Kanten braucht $\mathcal{O}(n^2)$ bzw. $\mathcal{O}(nm)$ Speicherplatz.

Wünschenswert: Darstellung mit linearem Speicherbedarf in $\mathcal{O}(n + m)$

Definition 6.3. Adjazenz-/Inzidenzlisten: Jeder Knoten verwaltet eine Liste der zu ihm inzidenten Kanten (oder alternativ zu ihm adjazenten Knoten). In gerichteten Graphen verwaltet jeder Knoten entweder eine Liste der ausgehenden Kanten oder zwei Listen, getrennt nach ausgehenden und eingehenden Kanten.

Vor- und Nachteile der Methoden:
Adjazenzmatrizen:

- Schneller Test, ob zwei Knoten mit einer Kante verbunden sind
- Bei $\Omega(n^2)$ Kanten sind sie effizienter als Listen

Adjazenzlisten:

- Brauchen weniger Speicher, wenn $m \ll n^2$.
- Können Algorithmen auf Graphen mit wenig ($z.B. \mathcal{O}(n)$) Kanten beschleunigen.

Bemerkung. In den meisten Kontexten eignen sich Adjazenz- oder Inzidenzlisten am besten.

7 Graphenalgorithmen

7.1 Elementare Datenstrukturen

Stacks

- Listen, die Elemente hinten einfügen und entnehmen
- Das letzte Element eines Stacks wird zuerst entnommen
- LIFO – Last in, first out

Queues

- Listen, die Elemente hinten einfügen und vorne entnehmen
- Das erste Element in einer Queue wird zuerst entnommen
- FIFO – First in, first out

7.2 Graphendurchmusterung

Input: Graph $G = (V, E, \Psi)$, Knoten $r \in V$
Output: Menge $R \subseteq V$ der von r erreichbaren Knoten $F \subseteq E$, so dass $(R, F, \Psi|_F)$ (bzw. Arboreszenz mit Wurzel r)

```
1  R ← {r}; F ← ∅; Q ← {r} while Q ≠ ∅ do
2      sei v das nächste Element in Q
3      if ∃ e ∈ δ^(+) mit w ∈ Ψ(e) \ R then
4          R ← R ∪ {w}
5          Q ← Q ∪ {w}
6          F ← F ∪ {e}
7      end
8      else
9          Q ← Q \ {v}
10     end
11 end
12 return R, F
```