Kap. 0.2 Definitionen und Graphrepräsentationen

Professor Dr. Petra Mutzel

Abteilung für Computational Analytics

Institut für Informatik (Abt. 1)

Universität Bonn

SoSe 2024 Graphenalgorithmen VO 2 16. April 2024

Outline

Überblick

1. **Definitionen**
   - Graphen
   - Nachbarschaft und Knotengrade
   - Pfade, Wege und Kreise
   - Besondere Graphen und Graphstrukturen

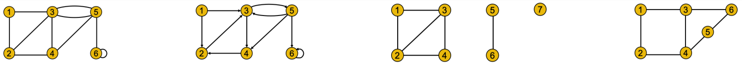
2. **Graphrepräsentationen**
   - Listenbasierte Strukturen
   - Array Strukturen
   - Adjazenzmatrix
   - Dynamische Graphen

Überblick

Definitionen: Graphen

Definition

- Ein Graph $G = (V, E)$ besteht
- aus einer Menge $V = V(G)$ von Knoten und
- einer (Multi-)menge $E = E(G)$ von Kanten, die Paaren von Knoten entsprechen.

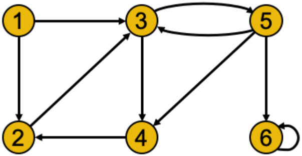


- Bei Multimenge kann ein Paar (v, w) mehrfach in E vorkommen $\rightarrow$ Mehrfachkanten
- Eine Kante (v, v) heißt Schleife (self-loop)

Annahmen:

- V und E sind endliche Mengen
- Mehrfachkanten erlaubt

Gerichtete Graphen

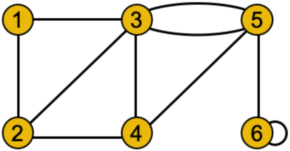


Definition

- Sind die Paare in E geordnet: $E \subseteq V \times V \rightarrow$ gerichteter Graph (Digraph)
- Kanten (u, v) heißen dann: (gerichtete) Kanten (Bögen, directed edges, arcs)

- Maximale Kantenanzahl eines Digraphen ohne Schleifen und Mehrfachkanten: $|E| \leq |V|(|V| - 1)$

Ungerichtete Graphen



Definition

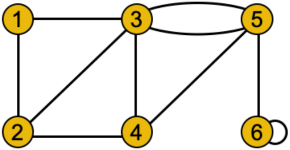
- Sind die Paare in E ungeordnet: $\rightarrow$ ungerichteter Graph
- Kanten $\{u, v\}$ heißen dann: Kanten (edges)

- Maximale Kantenanzahl eines ungerichteten Graphen ohne Schleifen und Mehrfachkanten: $|E| \leq \frac{1}{2}|V|(|V| - 1)$

In der Literatur (und in der VO) schreibt man oft ungerichtete Kanten als (u, v) oder uv statt $\{u, v\}$

Überblick

Nachbarschaft in Graphen

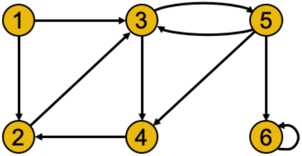


Definition (Nachbarn in Graphen)

Sei $e = (v, w)$ eine Kante in E, dann sagen wir:

- v und w sind adjazent (benachbart)
- v (bzw. w) und e sind inzident
- v und w sind Endpunkte von e
- v und w sind Nachbarn
- e ist eine ausgehende Kante von v und eine eingehende Kante von w (falls G Digraph)

Nachbarschaft in Digraphen



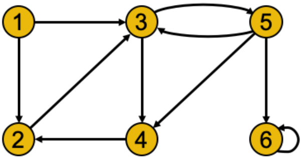
Definition (Nachbarn in Digraphen)

Sei $G = (V, A)$ ein gerichteter Graph. Dann definieren wir:

- Eingehende Nachbarmenge von $v \in V$:
  $N^{-}(v) := \{u \in V \mid (u, v) \in A\}$
- Ausgehende Nachbarmenge von $v \in V$:
  $N^{+}(v) := \{w \in V \mid (v, w) \in A\}$

Die Nachbarknoten eines Knoten sind normalerweise nur die Endknoten der ausgehenden Nachbarmenge.

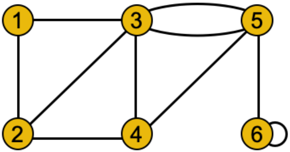
Nachbarschaft in Digraphen



Definition (Nachbarn in Digraphen)

- $A^{-}(v)$:=Menge der eingehenden Kanten von v
- $A^{+}(v)$:=Menge der ausgehenden Kanten von v
- $A(v) := A^{-}(v) \cup A^{+}(v)$
- Eingangsgrad $d^{-}(v) := |A^{-}(v)|$
- Ausgangsgrad $d^{+}(v) := |A^{+}(v)|$
- Knotengrad $d(v) := d^{-}(v) + d^{+}(v)$

Nachbarschaft in Graphen



Definition (Nachbarn in ungerichteten Graphen)

Sei $G = (V, E)$ ein ungerichteter Graph. Dann sagen wir:

- Nachbarmenge von $v \in V$: $N(v) := \{w \in V \mid (v, w) \in E\}$
- Menge der zu v inzidenten Kanten $E(v) := \{(u, v) \mid (u, v) \in E\}$
- Knotengrad $d(v)$ ist die Anzahl der zu v inzidenten Kanten, wobei eine Schleife 2 Mal gezählt wird

Ungerade Knotengrade

Lemma (Ungerade Knotengrade (Lemma 1))

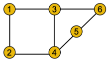
In einem ungerichteten Graphen $G = (V, E)$ ist die Anzahl der Knoten mit ungeradem Knotengrad gerade.

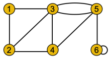
Beweis.

Summiert man über alle Knotengrade, so zählt man jede Kante genau zwei Mal:

$$\sum_{\text{ungerade } v \in V} d(v) + \sum_{\text{gerade } v \in V} d(v) = \sum_{v \in V} d(v) = 2|E|$$

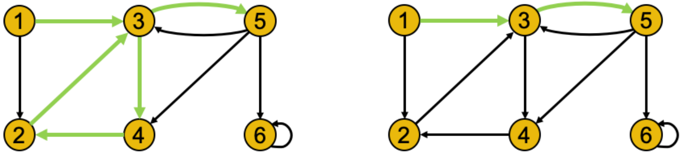
Damit die linke Seite der Gleichung gerade ist, muss die Anzahl der Knoten mit ungeradem Grad gerade sein. □





Überblick

Pfade, Wege und Kreise

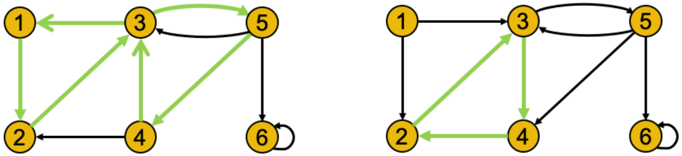


Definition

Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph.

- Ein Pfad (*path*) P der Länge k ist eine Folge $(v_0, e_1, v_1, \ldots, e_k, v_k)$ von abwechselnd Knoten und Kanten aus G mit $e_i = (v_{i-1}, v_i)$ für $i = 1, \ldots, k$. Man schreibt auch: $P = (v_0, v_1, \ldots, v_k)$.
- Ein Weg ist ein Pfad in dem alle Knoten verschieden sind.

Pfade, Wege und Kreise



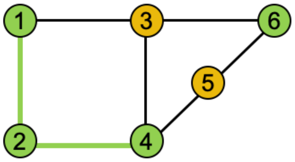
Definition

Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph.

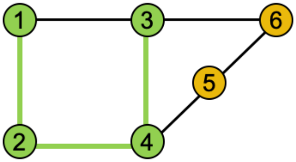
- Ist $(v_0, e_1, v_1, \ldots, e_k, v_k)$ ein Pfad mit $k \geq 2$ und $e = (v_k, v_0) \in E$, dann ist $(v_0, e_1, v_1, \ldots, e_k, v_k, e, v_0)$ ein Kreis der Länge $k + 1$ in G.
- Ein Kreis heißt einfach, wenn der ihn definierende Pfad $(v_0, e_1, v_1, \ldots, e_k, v_k)$ knotendisjunkt (also ein Weg) ist.

Überblick

Induzierter Graph



G[U]



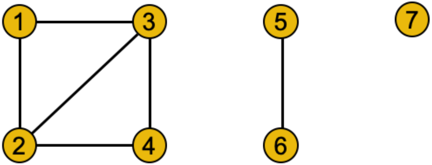
G[F]

Definition (Induzierter Graph)

Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph.

- $G[U]$ ist der von $U \subseteq V$ induzierte Teilgraph $H = (U, F)$ von G mit $F \subseteq E$ enthält alle Kanten mit beiden Endknoten in U.
- $G[F]$ ist der von $F \subseteq E$ induzierte Teilgraph $H = (U, F)$ von G mit $U \subseteq V$ enthält alle Knoten, die zu F inzident sind.

Zusammenhangskomponenten eines Graphen

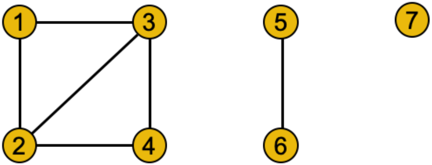


Sei $G = (V, E)$ ein ungerichteter Graph und $v, w \in V$.

Definition

- Gibt es einen Pfad von v nach w, dann gibt es auch einen von w nach v und wir schreiben $v \leftrightarrow w$.
- Zwei Knoten $v, w \in V$ heißen zusammenhängend, wenn $v \leftrightarrow w$ gilt.
- G heißt zusammenhängend, wenn zwischen je zwei Knoten $v, w \in V(G)$ gilt: $v \leftrightarrow w$; sonst heißt G unzusammenhängend.

Zusammenhangskomponenten eines Graphen

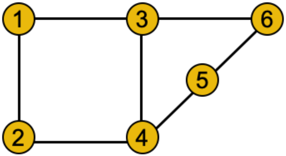


Definition (ff)

- Die Äquivalenzklassen von V bezüglich $\leftrightarrow$ heißen Zusammenhangskomponenten
- Die Zusammenhangskomponenten von G entsprechen also genau den größten zusammenhängenden Teilgraphen in G.

Zur Erinnerung: Äquivalenzrelation: reflexiv, symmetrisch, transitiv

Bipartiter Graph



Definition (Bipartiter Graph)

Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph.
G ist bipartit, wenn die Knotenmenge in zwei Mengen V_1 und V_2 geteilt werden kann, so dass alle Kanten zwischen V_1 und V_2 verlaufen,
d.h. für alle $e = (u, v) : (u \in V_1$ und $v \in V_2)$ oder $(u \in V_2$ und $v \in V_1)$.

Beobachtung:

Es gilt: G ist bipartit $\Leftrightarrow$ G enthält keine ungeraden Kreise (im ungerichteten Sinne)

Dünne und dichte Graphen

Definition (Dünne bzw. dichte Graphen)

Sei $G = (V, E)$ ein gerichteter oder ungerichteter Graph ohne Mehrfachkanten und $n = |V|$.

- Ein Graph G heißt dünn (engl. sparse), wenn die Anzahl der Kanten in $O(n \log n)$ ist.
- Ist die Kantenanzahl in $\Omega(n^2)$ dann bezeichnen wir den Graphen als dicht (engl. dense).

Überblick

Darstellung von Graphen im Rechner

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $n = |V|$ Knoten und $m = |E|$ Kanten ohne Schleifen und Mehrfachkanten.

Basis-Operationen

- Zugriff auf Informationen des Knoten
- Iteration über alle Nachbarn eines Knoten v
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$?
- Einfügen oder Entfernen von Knoten oder Kanten

Wichtig zur Analyse ist auch der benötigte Speicherplatz

Darstellung von Graphen im Rechner

Gegeben sei ein gerichteter Graph $G = (V, E)$ mit $n = |V|$ Knoten und $m = |E|$ Kanten ohne Schleifen und Mehrfachkanten.

Annahme: Knoten gegeben als ID von $1 \dots n$ gespeichert in Array

Wir analysieren neben dem Speicherplatz die Operationen

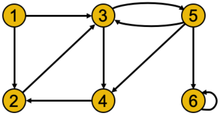
- Iteration über alle Nachbarn eines Knoten v
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$?
- Einfügen einer Kante
- Entfernung einer Kante

Annahme: Wir müssen auch bei Einfügen nicht auf Doppelkanten prüfen

Welche Datenstrukturen zur Graphrepräsentation kennen Sie?

Überblick

Einfache ungeordnete Kantenliste



Anfangsknoten	1	5	3	6	2	4	5	1	5	3
Endknoten.	3	3	4	6	3	2	6	2	4	5

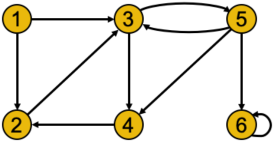
Einfach eine unsortierte Liste (als einfach verkettete Liste) aller Kanten

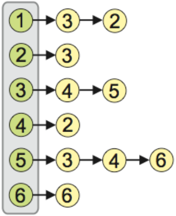
Analyse der Basis-Operationen

- Iteration über alle Nachbarn eines Knoten v: $O(m)$
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$? $O(m)$
- Addiere eine Kante: $O(1)$
- Entferne eine Kante: suchen $O(m)$, entfernen $O(1)$

Speicherplatz: $O(m)$ genauer: $2m$ Plätze für die Kantenliste (plus Pointer)

Ungeordnete Nachbarlisten per Knoten





Für jeden Knoten eine einfach verkettete Liste seiner Nachbarknoten

Analyse der Basis-Operationen

- Iteration über alle Nachbarn eines Knoten v: $O(d^+(v))$
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$? $O(d^+(v))$
- Addiere eine Kante: $O(1)$
- Entferne von Kante (v, w): suchen $O(d^+(v))$, entfernen $O(1)$

Speicherplatz: $O(m + n)$ genauer: m Plätze für die Kanten (plus Pointer)

Überblick

Adjazenz Arrays

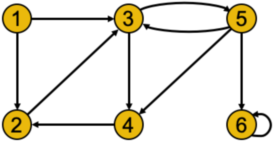
Für jeden Knoten ein unsortiertes Array seiner Nachbarknoten

Analyse der Basis-Operationen

- Iteration über alle Nachbarn eines Knoten v: $O(d^+(v))$
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$? $O(d^+(v))$
- Addiere eine Kante: $O(1)$ falls genug Speicherplatz vorhanden
- Entferne Kante (v, w): suchen $O(d^+(v))$, entfernen $O(d^+(v))$

Speicherplatz: $O(n^2)$ für dynamische Graphen, $O(m + n)$ für statische Graphen, genauer: m Plätze für die Kanten, n für die Knoten

Ein Adjazenz Array



Knoten	1	2	3	4	5	6	7
Indexarray	1	3	4	6	7	10	11

Knoten	1		2	3		4	5			6	7
Kantenarray	3	2	3	4	5	2	3	4	6	6	

- Alle Arrays (s. oben) werden zu einem Kantenarray zusammengefügt
- Ein zusätzliches Array (Indexarray) der Länge n enthält die Anfangspositionen dieser Unterfelder.
- Addiere den Eintrag $m + 1$ im Indexarray an Position $n + 1$

Ein Adjazenz Array

Analyse der Basis-Operationen

- Iteration über alle Nachbarn eines Knoten v: $O(d^+(v))$
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$? $O(d^+(v))$
- Addiere eine Kante: $O(m + n)$
- Entferne Kante (v, w): suchen $O(d^+(v))$, entfernen $O(m + n)$

Speicherplatz: $O(m + n)$ genauer: m Plätze für die Kanten plus $n + 1$ für das Indexarray

$\Rightarrow$ empfohlen für dünne statische Graphen, denn die Zugriffe auf Arrays sind deutlich schneller als solche auf Listen

Nur einmal Allokation eines zusammenhängenden Arrays notwendig im Gegensatz zu vielen Allokationen vieler Arrays wie beim Adjazenz Array

Überblick

Adjazenzmatrix

$$
\begin{array}{c|cccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 0 & 1 & 1 & 0 & 0 & 0 \\
2 & 0 & 0 & 1 & 0 & 0 & 0 \\
3 & 0 & 0 & 0 & 1 & 1 & 0 \\
4 & 0 & 1 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 1 & 1 & 0 & 1 \\
6 & 0 & 0 & 0 & 0 & 0 & 1 \\
\hline
\end{array}
$$

- Sei M eine $n \times n$ Matrix mit $0/1$ Einträgen $m_{ij} = 1$ falls zwischen den Knoten mit IDs i und j eine Kante existiert.
- bei Mehrfachkanten bedeuten die Einträge die Anzahl der Kanten

Adjazenzmatrix

Analyse der Basis-Operationen

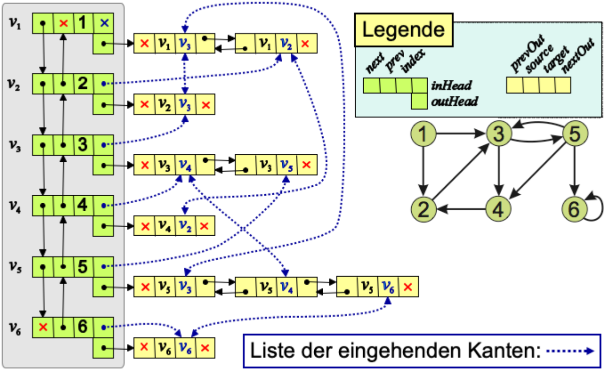
- Iteration über alle Nachbarn eines Knoten v: $O(n)$
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$? $O(1)$
- Addiere eine Kante: $O(1)$
- Entferne eine Kante: $O(1)$

Speicherplatz: $O(n^2)$ genauer: n^2 Plätze

$\Rightarrow$ empfohlen für dichte statische Graphen oder dynamische, wenn Knotenanzahl gleich bleibt

Überblick

Dynamische Graphen



Jeweils doppelt verkettete Listen für Knoten und Kanten

je eine Liste für die ausgehenden Kanten (schwarz) und eine Liste für die eingehenden Kanten (blau)

Analyse dynamischer Graphen

Analyse der Basis-Operationen

- Iteration über alle Nachbarn eines Knoten v: $O(d^+(v))$
- Abfrage von Kanten: Ist Kante (v, w) in $E(G)$? $O(d^+(v))$
- Addiere einen Knoten: $O(1)$
- Addiere eine Kante: $O(1)$
- Entferne Knoten v: $O(1)$ plus Aufwand für das Entfernen der Nachbarkanten $O(d^+(v) + d^-(v))$
- Entferne Kante (v, w): $O(d^+(v))$

Speicherplatz: $O(n + m)$

$\Rightarrow$ empfohlen für dynamische Graphen

Literatur zu Graphrepräsentationen

Sanders, Mehlhorn, Dietzfelbinger, Dementiev: Sequential and Parallel Algorithms and Data Structures: The Basic Toolbox, Springer, 2019, Kap. 8, Seiten 259–269