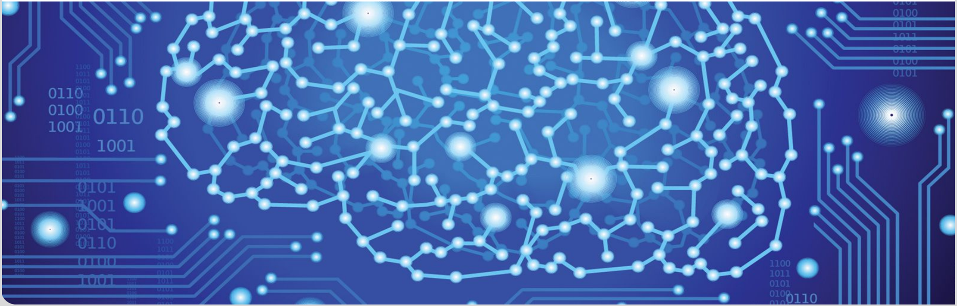
12. Tutorium
Algorithmen in Graphen, Quantitative Aspekte von Algorithmen

Grundbegriffe der Informatik, Tutorium #30

Alexander Klug | 22. Januar 2018

FAKULTÄT FÜR INFORMATIK



Roadmap

1. Algorithmen in Graphen
2. Quantitative Aspekte von Algorithmen

Roadmap

Wdh.: Adjazenzmatrix

Def.: Adjazenzmatrix

Sei $G = (V_G, E_G)$ gerichteter Graph.

Adjazenzmatrix

$A \in \{0, 1\}^{|V_G| \times |V_G|}$

$$A_{ij} = \begin{cases} 1, & \text{falls } (i, j) \in E_G \\ 0, & \text{falls } (i, j) \notin E_G \end{cases}$$

Sei $U = (V_U, E_U)$ ungerichteter Graph.

Adjazenzmatrix

$A \in \{0, 1\}^{|V_U| \times |V_U|}$

$$A_{ij} = \begin{cases} 1, & \text{falls } \{i, j\} \in E_U \\ 0, & \text{falls } \{i, j\} \notin E_U \end{cases}$$

Wdh.: Adjazenzmatrix

Def.: Adjazenzmatrix

Sei $G = (V_G, E_G)$ gerichteter Graph.
Adjazenzmatrix $A \in \{0,1\}^{|V_G| \times |V_G|}$

$$A_{ij} = \begin{cases} 1, & \text{falls } (i,j) \in E_G \\ 0, & \text{falls } (i,j) \notin E_G \end{cases}$$

Sei $U = (V_U, E_U)$ ungerichteter Graph.
Adjazenzmatrix $A \in \{0,1\}^{|V_U| \times |V_U|}$

$$A_{ij} = \begin{cases} 1, & \text{falls } \{i,j\} \in E_U \\ 0, & \text{falls } \{i,j\} \notin E_U \end{cases}$$

Besondere Eigenschaften der Adjazenzmatrix

- Schlingen lassen sich an einer 1 auf der Diagonalen erkennen (Wert von A_{ii})
- Bei ungerichteten Graphen ist A immer symmetrisch (also $A_{ij} = A_{ji}$).

Wdh.: Wegematrix

Def.: Wegematrix

Sei $G = (V_G, E_G)$ gerichteter Graph.

Wegematrix $W \in \{0, 1\}^{|V_G| \times |V_G|}$

$$W_{ij} = \begin{cases} 1, & \text{falls es in } E_G \text{ Pfad von } i \text{ nach } j \text{ gibt.} \\ 0, & \text{sonst} \end{cases}$$

Kantenrelation

Kantenmenge als Relation

Sei G ein gerichteter Graph mit der Knotenmenge V. Die Kantenmenge ist wie folgt definiert:
Sei $E \subseteq V \times V$ eine Relation auf V definiert durch
$x, y \in E \;\Leftrightarrow\;$ es gibt eine Kante vonx x nach y in G.

Beobachtungen

- Was gilt für E^2?

Kantenrelation

Karlsruher Institut für Technologie

Kantenmenge als Relation

Sei G ein gerichteter Graph mit der Knotenmenge V. Die Kantenmenge ist wie folgt definiert:
Sei $E \subseteq V \times V$ eine Relation auf V definiert durch
$x, y \in E \;\; \Leftrightarrow \;\;$ es gibt eine Kante von$x$ nach y in G.

Beobachtungen

- Was gilt für E^2? $\Rightarrow$ Knoten, die über einen Pfad der Länge 2 verbunden sind
- Analog $E^3, E^4, \ldots$
- Also $(x, y) \in E^*$ genau dann, wenn es einen Pfad (beliebiger Länge) von x nach y gibt.

Berechnung der Erreichbarkeitsrelation

Karlsruher Institut für Technologie

Erreichbarkeitsrelation

$$E^* = \bigcup_{i=0}^{\infty} E^i = \bigcup_{i=0}^{n-1} E^i$$

Matrizen für die Relation E^k

Sei G ein gerichteter Graph mit Adjazenzmatrix A. Für alle $k \in \mathbb{N}_0$ gilt:

$$\operatorname{sgn}((A^k)_{ij}) := \begin{cases} 1 & \text{falls in } G \text{ ein Pfad der Länge } k \text{ von } i \text{ nach } j \text{ existiert} \\ 0 & \text{falls in } G \text{ kein Pfad der Länge } k \text{ von } i \text{ nach } j \text{ existiert} \end{cases}$$

Berrechnung der Wegematrix

Berrechnung der Wegematrix

Es sei G ein gerichteter Graph mit Adjazenzmatrix A. Dann gilt für alle $k \geq n - 1$:

$$W = \operatorname{sgn}\left(\sum_{i=0}^{k} A^i\right)$$

ist die Wegematrix des Graphen G.

Roadmap

Laufzeit

```
public int algorithm (int[] a) {
    int sum = 0;
    for (int i=0; i<a.length; i++)
        if (a[i]>0)
            sum += a[i];
    return sum;
}
```

Aufgabe

Wie viele Instruktionen werden aufgerufen?

Laufzeit

```
public int algorithm (int[] a) {
    int sum = 0;
    for (int i=0; i<a.length; i++)
        if (a[i]>0)
            sum += a[i];
    return sum;
}
```

Aufgabe

Wie viele Instruktionen werden aufgerufen?
Im worst case? Im best case? Im average case?

O-Notation

Karlsruher Institut für Technologie

Def. Asymptotisches Wachstum $\asymp$

Seien $f, g\colon \mathbb{N}_0 \to \mathbb{R}_0^+$. Dann gilt

$$f \asymp g \Leftrightarrow \exists c, c' \in \mathbb{R}_+ : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : cf(n) \leq g(n) \leq c' f(n)$$

Man sagt auch g wächst genauso schnell wie f. $\asymp$ ist eine Äquivalenzrelation.

Beispiele

- $42n^6 - 33n^3 + 222n^2 - 15 \asymp 66n^6 + 55555n^5$
- $n^{3+1} + 5n^2 \asymp 3n^3 - n$

$\mathcal{O}$-Notation

Θ-Kalkül

$$\Theta(f) = \{g \mid f \asymp g\}$$
$$= \{g \mid \exists c, c' \in \mathbb{R}_+ : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : cf(n) \leq g(n) \leq c' f(n)\}$$

Bemerkungen

- Im Θ-Kalkül von $f(n)$ sind genau die Funktionen enthalten, die asymptotisch gleich schnell wachsen wie $f(n)$.
- Schreibe $g(n) \in \Theta(f(n))$, wenn $g(n)$ asymptotisch gleichschnell wächst wie $g(n)$.
- Ist f ein Polynom, so sind insbesondere in $\Theta(f(n))$ alle Polynome enthalten, die den gleichen Grad wie f haben.
- Es gilt $\log_b(n) \in \Theta(\log_a(n))$. Die Basis ist also egal und man kann auch $\Theta(\log n)$ schreiben.

O-Notation

Karlsruher Institut für Technologie

Def.: O-Kalkül

$$O(f) = \{g \mid \exists c \in \mathbb{R}_+ : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : g(n) \leq cf(n)\}$$

$g(n) \in O(f(n))$ (oder $g \preceq f$) genau dann, wenn g asymptotisch höchstens so schnell wächst wie f.

Def.: Ω-Kalkül

$$\Omega(f) = \{g \mid \exists c \in \mathbb{R}_+ : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : g(n) \geq cf(n)\}$$

$g(n) \in \Omega(f(n))$ (oder $g \succeq f$) genau dann, wenn g asymptotisch höchstens so schnell wächst wie f.

Beobachtung: $\Theta(f) = O(f) \cap \Omega(f)$

O-Notation

Aufgabe

1. Für welches $c \in \mathbb{R}_+$ gilt $5n^4 \in O(n^c)$ bzw $5n^4 \in \Omega(n^c)$?
2. Für welches $c \in \mathbb{R}_+$ gilt $5n^4 \in O(c^n)$ bzw $5n^4 \in \Omega(c^n)$?
3. Für welches $c \in \mathbb{R}_+$ gilt $2^n \in O(c^n)$ bzw $2^n \in \Omega(c^n)$?
4. Zeige oder widerlege: $n \in \Theta(\sqrt{n})$

O-Notation

Aufgabe

1. Für welches $c \in \mathbb{R}_+$ gilt $5n^4 \in O(n^c)$ bzw $5n^4 \in \Omega(n^c)$?
2. Für welches $c \in \mathbb{R}_+$ gilt $5n^4 \in O(c^n)$ bzw $5n^4 \in \Omega(c^n)$?
3. Für welches $c \in \mathbb{R}_+$ gilt $2^n \in O(c^n)$ bzw $2^n \in \Omega(c^n)$?
4. Zeige oder widerlege: $n \in \Theta(\sqrt{n})$

Lösung

1. Es gilt: $\forall c \geq 4$ bzw. $\forall c \leq 4$.
2. Es gilt: $\forall c > 1$ bzw. $\forall c \leq 1$.
3. Es gilt: $\forall c \geq 2$ bzw. $\forall c < 2$.
4. Annahme: Die Behauptung ist richtig.
   Dann gilt: $n \in O(\sqrt{n}) n \in \Omega(\sqrt{n})$, insbesondere $n \in \Omega(\sqrt{n})$.
   $\Rightarrow \exists c \in \mathbb{R}_+ : \exists n_0 \in \mathbb{N}_0 : \forall n \geq n_0 : n \leq c\sqrt{n} \Leftrightarrow \frac{n}{\sqrt{n}} \leq c \Leftrightarrow \sqrt{n} \leq c$.
   Widerspruch.

O-Notation

Aufgabe

Zeige oder widerlege:

$$f(n) + g(n) \in O(g(f(n)))$$

O-Notation

Aufgabe

Zeige oder widerlege:

$$f(n) + g(n) \in O(g(f(n)))$$

Lösung

Die Behauptung stimmt nicht. Wähle z.B. $f(n) = n^2$ und $g(n) = \sqrt{n}$ und führe dies zu einem Widerspruch.

O-Notation

- Θ entspricht nicht dem average case.

O-Notation

- Θ entspricht nicht dem average case.
- $O(1)$ bedeutet konstante Laufzeit

O-Notation

- Θ entspricht nicht dem average case.
- $O(1)$ bedeutet konstante Laufzeit
- Beachtet den Trick mit den Limites[1]

[1] Ich weiß nicht, ob ihr den als Beweis in der Klausur oder auf dem Blättern verwenden dürft. Zur Kontrolle sollte man ihn aber kennen.

Mastertheorem

Problemstellung:

Gegeben sei eine rekursiv definierte Funktion T.
Frage: Welche Laufzeit hat T?

Beispiel:

$$T(n) = 8T\left(\frac{n}{2}\right) + 1000n^2$$

$\Rightarrow$ Mastertheorem

Mastertheorem

Karlsruher Institut für Technologie

Def.: Mastertheorem

Seien $a \geq 1$ und $b > 1$ Konstanten, $f : \mathbb{N} \rightarrow \mathbb{R}_0^+$ und $T(n)$ eine Laufzeitfunktion der Form

$$T = aT\left(\frac{n}{b}\right) + f$$

Dann gilt nach dem **Mastertheorem**:

- **Fall 1:**
  Wenn $f \in O(n^{\log_b a - \varepsilon})$ für ein $\varepsilon > 0$ ist, dann ist $T \in \Theta(n^{\log_b a})$.
- **Fall 2:**
  Wenn $f \in \Theta(n^{\log_b a})$ ist, dann ist $T \in \Theta(n^{\log_b a} \log n)$.
- **Fall 3:**
  Wenn $f \in \Omega(n^{\log_b a + \varepsilon})$ für ein $\varepsilon > 0$ ist, und wenn es eine Konstante d gibt mit $0 < d < 1$, so dass für alle hinreichend großen n gilt $af\left(\frac{n}{n}\right) \leq df$, dann ist $T \in \Theta(f)$.

Mastertheorem

Beispiel zum 1. Fall

Sei $T(n) = 8T\left(\frac{n}{2}\right) + 1000n^2$.

- Aus der Formel lässt sich ablesen:
  $a = 8$, $b = 2$, $f(n) = 1000n^2$
- $n^{\log_b a}$ bestimmen:
  $\log_b a = \log_2 8 = 3 \Rightarrow n^{\log_b a} = n^3$
- $n^{\log_b a}$ mit $f(n)$ vergleichen: $1000n^2 \in O(n^3 - \varepsilon)$?
  Ja, für $\varepsilon = 1$ gilt $1000n^2 \in O(n^2)$.
- Mit dem Mastertheorem folgt:
  $T(n) = \Theta(n^3)$

Mastertheorem

Beispiel zum 2. Fall

Sei $T(n) = 2T\left(\frac{n}{2}\right) + 10n$.

- Aus der Formel lässt sich ablesen:
  $a = 2$, $b = 2$, $f(n) = 10n$
- $n^{\log_b a}$ bestimmen:
  $\log_b a = \log_2 2 = 1 \Rightarrow n^{\log_b a} = n^1$
- $n^{\log_b a}$ mit $f(n)$ vergleichen: $10n \in \Theta(n)$?
  Ja!
- Mit dem Mastertheorem folgt:
  $T(n) = \Theta(n \log n)$

Mastertheorem

Beispiel zum 3. Fall

Sei $T(n) = 2T\left(\frac{n}{2}\right) + n^2$.

- Aus der Formel lässt sich ablesen:
  $a = 2$, $b = 2$, $f(n) = n^2$
- $n^{\log_b a}$ bestimmen:
  $\log_b a = \log_2 2 = 1 \Rightarrow n^{\log_b a} = n^1$
- $n^{\log_b a}$ mit $f(n)$ vergleichen: $n^2 \in \Omega(n^{1+\varepsilon})$?
  Ja, für $\varepsilon = 1$ gilt $n^2 \in \Omega(n^2)$.
- Zusatzbedingung überprüfen: Ist $af\left(\frac{n}{n}\right) \leq df$?
  Ja, für $d = \frac{1}{2}$ gilt $\forall n \geq 1 \ : \ \frac{1}{2}n^2 \leq \frac{1}{2}n^2$
- Mit dem Mastertheorem folgt:
  $T(n) = \Theta(n^2)$

Was ihr jetzt kennen und können solltet...

- Von der Adjazenzmatrix zur Wegematrix
- Laufzeiten von Algorithmen angeben und abschätzen
- Mit dem O-Kalkül arbeiten
- Die Laufzeit rekursiver Algorithmen mit dem Mastertheorem bestimmen

Ausblick

- Erste Nutzung von Graphen: endliche Automaten
- Neue Möglichkeiten mit formalen Sprachen: reguläre Ausdrücke und rechtslineare Grammatriken

Fragen?

Vielen Dank für Eure Aufmerksamkeit!
Bis nächste Woche :)