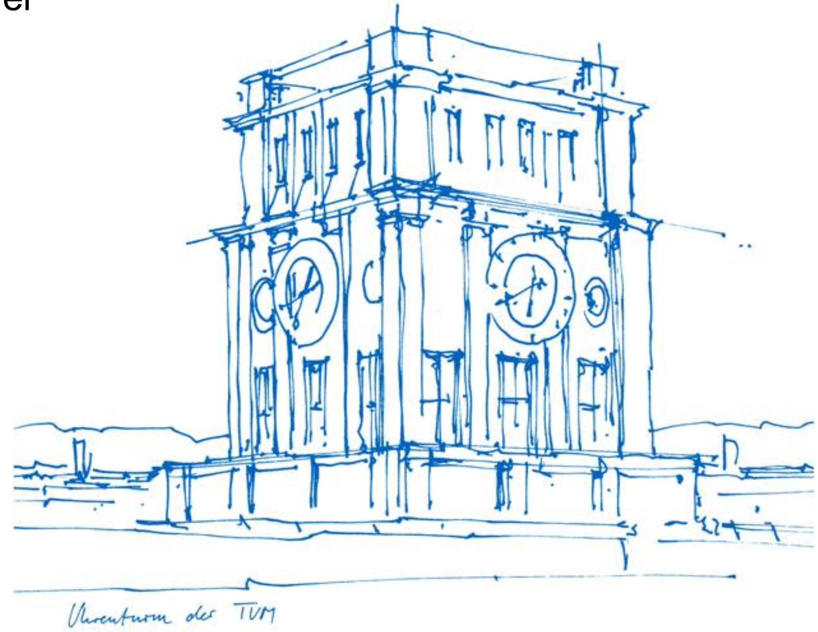
Diskrete Strukturen – Tutorium KW 50

Jeremias Bohn, Evghenii Beriozchin/Manuela Poschenrieder

Technische Universität München

Fakultät für Informatik

Garching, 10./11. Dezember 2020



Ungerichtete / Einfache Graphen

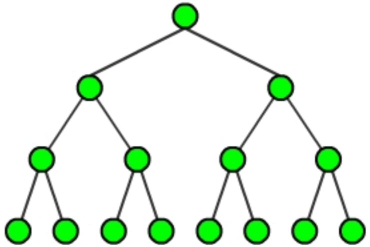
- Wir haben bisher nur gerichtete Graphen $G = (V, E)$ mit E einer Menge von 2-Tupeln betrachtet
- Ein ungerichteter bzw. einfacher Graph hat allerdings keine „Pfeile" sondern nur „Striche" zwischen einzelnen Elementen
  - Wir definieren daher für einen solchen Graphen E als eine Menge von Mengen, es gilt also $\{x, y\} = \{y, x\}$

Wichtige Graphentypen

- K_n: Vollständiger Graph, alle Knoten stehen miteinander in Verbindung
- C_n: Kreis der Länge n
- Q_n: Hyperwürfel der Dimension n
- Bäume

Bäume

- Ein einfacher Graph heißt **Baum**, wenn er kreisfrei und zusammenhängend ist
  - Ein Knoten u heißt **Blatt**, wenn $\deg(u) = 1$ gilt, also es eine Kante zu genau einem anderen Knoten gibt
  - Alle anderen Knoten heißen **innere Knoten**
- Binär- und Ternärbäume sind wichtige Datenstrukturen (siehe IN0007 Grundlagen: Algorithmen und Datenstrukturen)
  - In einem perfekten (bzw. vollständigen) Binärbaum hat jeder Knoten, bis auf die Blätter, 2 Kinderknoten und jeder Knoten, bis auf die Wurzel, einen Elternknoten (siehe Abbildung)
  - Analog dazu funktioniert der Ternärbaum mit 3 Kinderknoten
  - Der perfekte Binärbaum hat 2^h Blätter (h entspricht der Höhe, beginnend mit 0 bei der Wurzel) und $2^{h+1} - 1$ Knoten (lässt sich induktiv beweisen!)



Gradsequenz & Havel-Hakimi-Algorithmus

Eine Gradsequenz gibt die Anzahl der Knoten an, mit denen ein Knoten in Verbindung steht

- Ein Graph ist k-regulär, falls alle Knoten von Grad k sind

Havel-Hakimi-Algorithmus:

- Ein Graph mit $(d_1, \ldots, d_n)$ ist realisierbar, wenn ein Graph *sort*$(d_1, \ldots, d_{n-d_n-1}, d_{n-d_n} - 1, \ldots, d_{n-1} - 1)$ existiert
- $\Rightarrow$ Rekursiv überprüfen und dann generieren!

Selbstabbildungen und Notationen

Eine Funktion $f: A \to A$ heißt **Selbstabbildung**

- Ist f bijektiv, sprechen wir auch von einer Permutation

Wenn A endlich ist, können wir die Funktion in Zwei-Zeilen-Notation schreiben:

$$f := \begin{pmatrix} a_1 & a_2 & a_3 & a_4 & \dots & a_n \\ f(a_1) & f(a_2) & f(a_3) & f(a_4) & \dots & f(a_n) \end{pmatrix}$$

Ist f zudem eine Permutation, können wir die Zyklenschreibweise nutzen. Diese ergibt sich aus der Zwei-Zeilen-Notation:

$$\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} \Rightarrow (1,3)(2)$$

Automorphismen

Für einen Graphen $G = (V, E)$ heißt eine Funktion $\alpha\colon V \to V$ Automorphismus von G, wenn gilt:

$$(u, v) \in \mathrm{E} \Leftrightarrow (\alpha(u), \alpha(v)) \in E$$

Daraus folgt, dass nach Anwendung von α auf alle Knoten die Nachbarschaft $\Gamma(v)$ für alle Knoten v noch gleich sein muss, also muss gelten:

$$\Gamma\big(\alpha(v)\big) = \{\alpha(w) \mid w \in \Gamma(v)\}$$

Informell ist ein Automorphismus also eine Funktion über den Knoten eines Graphen, die diese umbenennt ohne die eigentliche Struktur des Graphen zu ändern.