Graphentheorie 1

Diskrete Strukturen

Uta Priss
ZeLL, Ostfalia

Sommersemester 2016

Agenda

Hausaufgaben

Graph-Äquivalenz

SetlX

Ankündigung: die nächsten beiden Mittwoche findet die Vorlesung in Raum 026 statt (wegen Baulärm).

SetlX-Aufgaben

Bei fast allen als falsch bewerteten Aufgaben war der Code entweder unvollständig oder der Grund war ein Tippfehler oder Syntaxfehler (falsche Klammer/Variable/Operationssymbol etc).

Als Ursache für die Tippfehler/Syntaxfehler kommen eigentlich nur Fehler beim Abschreiben in Frage.

Fragen: Was bedeuten „inzident“ und „adjazent“?

- Die Kanten ab und bc in Abbildung sind inzident, nicht aber die Kanten ad und bc. Wieso ist bc inzident und nicht inzident?
- „inzident“ (zusammenfallen, ähnliches Wort: coincidence) und „adjazent“ (daneben-liegen)

Fragen

- Was ist nun die gewünschte Schreibweise bei Kanten von normalen Graphen? $\{a, b\}$, (a,b), ab oder ba?
- ... entspricht die Benutzung von Digraphen einer mathematischen Norm?

Es gibt keine Normen. (So ziemlich) jede Schreibweise ist ok, wenn klar ist, was gemeint ist.

zB Ein Digraph mit Kante *ab* ..., Ein Graph mit Kante *ab* ...
Dabei hat *ab* jeweils eine etwas andere Bedeutung.

Fragen

Bei der graphischen Darstellung von Relationen wurden bisher Digraphen verwendet. Könnte eine symmetrische Relation zwischen den Elementen a und b auch in einem Graphen (nicht Digraphen) dargestellt werden?

D.h. kann man die Pfeile weglassen, wenn man weiß, dass die Relation symmetrisch ist?

Fragen

Gibt es eine Möglichkeit anhand der „Formel“ des Graphens zu prüfen, ob er planar ist?

Welchen Nutzen kann man aus den Sätzen 15.3 (Summe der Grade) und 15.4 (Anzahl der Knoten mit ungeradem Grad) ziehen?

Fragen

Gibt es eine Möglichkeit anhand der „Formel“ des Graphens zu prüfen, ob er planar ist?

Für manche Eigenschaften gibt es Methoden, sie zu überprüfen (z.B. für Planarität gibt es eine Methode mit speziellen Teilgraphen), für andere Eigenschaften (z.B. Isomorphie) gibt es keinen effizienten Algorithmus.

Welchen Nutzen kann man aus den Sätzen 15.3 (Summe der Grade) und 15.4 (Anzahl der Knoten mit ungeradem Grad) ziehen?

Man sammelt erst mal Fakten, die man leicht beweisen kann, um besser zu verstehen, was die mathematische Struktur „Graph“ eigentlich ist. Einiges davon braucht man in Anwendungen bestimmt.

Fragen

- Adjazenzmatrix (nächste Woche)
- Multigraphen (nicht klausurrelevant)
- Ist es richtig, dass ein Multigraph eigentlich ein nicht möglicher planarer Graph ist, der nun doch irgendwie möglich ist? (nein)
- Kann es beim Digraph Schlingen geben?
- Ist einem isolierten Knoten eine Position zuzuordnen?
- Wie kann ein Graph eine Richtung haben?

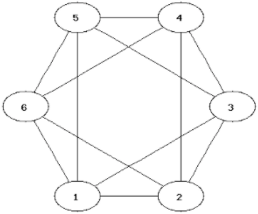
Fragen zur Äquivalenz

- Was muss man noch prüfen außer Anzahl Knoten und Kanten?
- Reicht es die Grade der Knoten/Nachbarn zu überprüfen?
- Warum gibt es unendlich viele zueinander isomorphe Graphen?
- Sind zwei Graphen äquivalent zueinander, wenn sie schon allein die gleiche Anzahl an Knoten und Kanten haben?

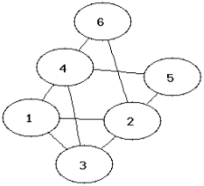
Welche dieser Graphen sind gleich?

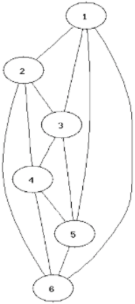
3:

1:

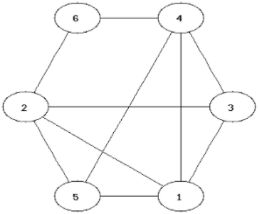


2:

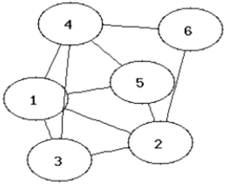




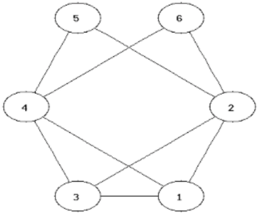
4:



5:



6:



Was ist der Unterschied zwischen den Graphen auf der vorhergehenden Folie und Graphen, wie sie im Buch definiert sind?

Was bedeuten „gleich“ und „äquivalent“ für Graphen?

Wiederholung: was ist eine Bijektion?

$f : D \to M$ bijektiv $\Longleftrightarrow$ ...

Vervollständigen Sie diese Definition. Schreiben Sie die Bedingungen nicht als Text, sondern als Formeln.

Zusammenhang Bijektion und Äquivalenz

Wann sind zwei Graphen äquivalent?

Schreiben Sie die Äquivalenzrelation $r_{äquiv}$ für eine Menge $\mathbb{G}$ von Graphen formal auf.

Warum sind die Bedingungen der Äquivalenzrelation („reflexiv“, „symmetrisch“ und „transitiv“) jeweils erfüllt?

Was sind also die Äquivalenzklassen für Graphen?

$$r_{\ddot{a}quiv} \subseteq \mathbb{G} \times \mathbb{G}$$
$$(g_1, g_2) \in r_{\ddot{a}quiv} :\Longleftrightarrow \exists_f : f \textit{ ist eine bijektive Abbildung}$$
$$\textit{wie in Def. 15.6 im Buch beschrieben.}$$

Reflexiv: f ist eine Identitätsfunktion, die jeden Knoten auf sich selbst abbildet.

Symmetrisch: da f bijektiv ist, gibt es eine Umkehrfunktion f^{-1}.

Transitiv: man kann die bijektiven Abbildungen verketten. Wenn f_1 Knoten von g_1 auf Knoten von g_2 abbildet, und f_2 Knoten von g_2 auf Knoten von g_3, dann bildet $f_2 \circ f_1$ Knoten von g_1 auf Knoten von g_3 ab.

Wie programmiert man mit Graphen?

Welche Datenstrukturen bieten sich an?

Beispiel: For-Schleife der Kanten

```
kantenliste := procedure(knoten, kanten) {
   for (k1 in knoten) {
      for (k2 in knoten) {
         if ({k1,k2}in kanten && k1 <= k2) {
            print({k1,k2});
         }
      }
   }
};
```

Wie würde man alle Kanten in einem Digraphen ausgeben?

Schreiben Sie eine Funktion, die den Komplementärgraph zu einem Graphen ausgibt. Der Komplementärgraph hat genau dann eine Kante, wenn der ursprüngliche Graph keine Kante hat.

Ankündigung: die nächsten beiden Mittwoche findet die Vorlesung in Raum 026 statt (wegen Baulärm).