Datenstrukturen und Algorithmen, SS 2024
Prof. Dr. P. Rossmanith
D. Cloerkes, M. Gehnen, D. Mock, N. Speetzen

Abgabe: 07.06.2024

Übungsblatt mit Lösungen 07

Aufgabe T23

Zeigen oder widerlegen Sie: Jeder endliche ungerichtete Graph mit mehr als einem Knoten hat mindestens zwei Knoten mit gleichem Grad.

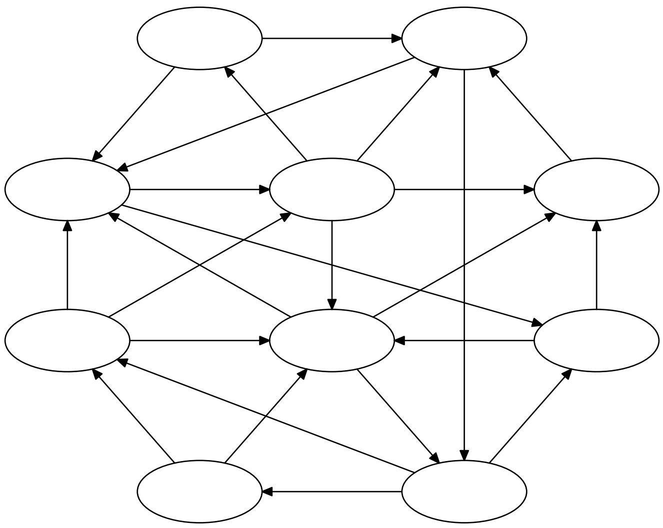
Lösungsvorschlag

Diese Aussage ist wahr.

Falls jeder Knoten unterschiedlichen Grad hat, und es n Knoten gibt, müssen alle Knotengrade von 0 bis $n-1$ vorkommen, da der maximale Grad höchstens $n-1$ sein kann. Der Knoten mit Grad $n-1$ kann nicht mit dem Knoten mit Grad 0 verbunden sein. Er hat also nur $n-2$ mögliche Nachbarn. Also kann er nicht $n-1$ viele Kanten haben. Damit haben wir einen Widerspruch, und gezeigt, dass die Aussage wahr ist.

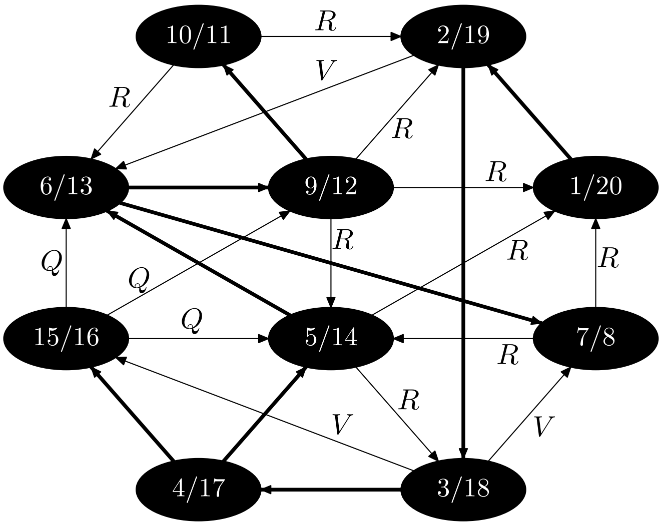
Aufgabe T24

Führen Sie eine Tiefensuche auf dem folgenden Graphen durch. Geben Sie zu jedem Knoten *discovery* und *finish* Zeiten an, und geben Sie dann zu jeder Kante an, welchen Typ sie bezüglich des entstandenen Tiefensuchwaldes hat (Baumkante, Vorwärtskante, Rückwärtskante, Querkante).



Welche starken Zusammenhangskomponenten hat der Graph?

Lösungsvorschlag



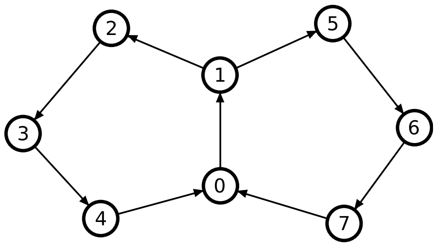
Der Graph hat eine einzige starke Zusammenhangskomponente.

Aufgabe T25

Beweisen oder widerlegen Sie: Ergibt eine Tiefensuche in einem gerichteten Graphen genau eine Rückwärtskante, so liefert jede Tiefensuche in diesem Graphen genau eine Rückwärtskante.

Lösungsvorschlag

Die Aussage ist falsch, wie man sich an folgendem Gegenbeispiel überlegen kann.

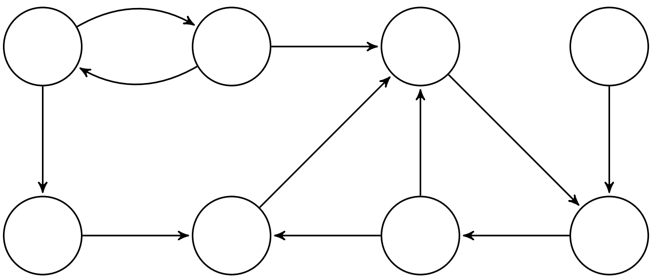


Beginnen wir die Tiefensuche am Knoten 1, so findet man genau eine Rückwärtskante, nämlich $(0, 1)$ (egal welchen Kreis man zuerst besucht). Eine Tiefensuche, die am Knoten 0 startet, wird jedoch die Rückwärtskanten $(4, 0)$ und $(7, 0)$ finden!

Aufgabe T26

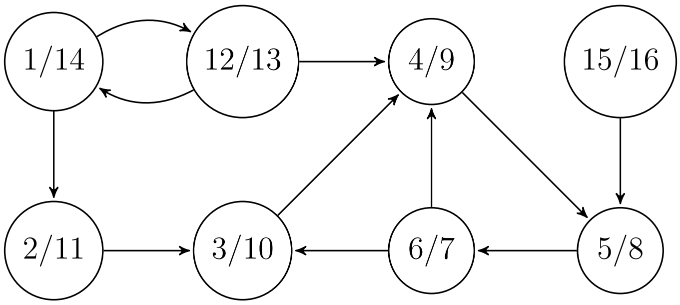
Verwenden Sie den Algorithmus von Kosaraju, um die starken Zusammenhangskomponenten dieses Graphen zu finden.

Markieren Sie zum Darstellen der Lösung im Graphen alle Knoten, welche zur selben starken Zusammenhangskomponente gehören, mit demselben Buchstaben.

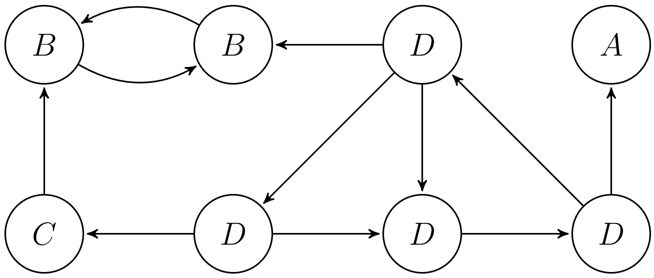


Lösungsvorschlag

Tiefensuche auf dem Graphen:



Inverse Tiefensuche in Reihenfolge der finish-Zeiten resultiert in den Zusammenhangskomponenten:



Aufgabe H22 (8 Punkte)

Zeigen Sie, wie viele Zusammenhangskomponenten ein ungerichteter Graph höchstens haben kann, wenn alle n Knoten einen Grad von mindestens $\lfloor \frac{n}{2} \rfloor - 1$ haben.

Achten Sie auch auf Ausnahmen für kleine n. Geben Sie Ihre Lösung ggfs. in Abhängigkeit von n an.

Lösungsvorschlag

Die Knoten des Graphen können in zwei vollständig verbundene Komponenten aufgeteilt werden, sodass für alle Knoten die Bedingung erfüllt ist. Würden die Knoten in drei getrennte Komponenten aufgeteilt werden, können die Knoten in der kleinsten Komponente maximal den Grad $\lfloor \frac{n}{3} \rfloor - 1$ haben. Da $\lfloor \frac{n}{3} \rfloor - 1 < \lfloor \frac{n}{2} \rfloor - 1 \quad \forall n > 3$, sind im allgemeinen nur zwei Zusammenhangskomponenten möglich.

Die einzigen Ausnahmen sind $n = 1$ mit einer einzigen Komponente und $n = 3$, wo die untere Schranke für den Grad bei 0 liegt, womit hier 3 Komponenten möglich sind.

Aufgabe H23 (3 + 6 + 6 Punkte)

In einem ungerichteten Graph ist jede Rückwärtskante auch eine Vorwärtskante sowie jede Vorwärtskante auch eine Rückwärtskante. Beweisen Sie die folgenden Aussagen:

a) Ein ungerichteter Baum mit $n \geq 1$ Knoten hat genau $n - 1$ Kanten.

b) Eine Tiefensuche in einem ungerichteten Graphen liefert keine Querkanten.

c) Ergibt eine Tiefensuche in einem ungerichteten Graphen genau eine Rückwärtskante, so liefert jede Tiefensuche in diesem Graphen genau eine Rückwärtskante.

Lösungsvorschlag

a) Dies kann man sich einfach veranschaulichen, indem man in einem Baum willkürlich eine Wurzel festlegt und alle Kanten von der Wurzel weg orientiert—auf diese Weise zeigt nun genau eine Kante auf jeden Knoten, abgesehen von der Wurzel, auf welche keine Kante zeigt. Alternativ kann man auch Induktion benutzen: Ein Baum mit einem Knoten hat null Kanten. Ein neu eingefügtes Blatt erhöht die Kantenzahl um eins.

b) Angenommen eine Tiefensuche würde eine Querkante zischen zwei Knoten u und v liefern. Das heißt v ist kein Nachfahre von u und u ist ein Nachfahre von v. Sei u der Knoten mit der geringeren discovery Zeit. Während u grau markiert ist, ist v also noch weiß markiert. Eine Tiefensuche würde die Kante von u nach v als Baumkante aufnehmen. Dies ist ein Widerspruch. Somit kann es keine Querkanten geben.

c) Stellen wir zunächst fest, dass wir uns auf zusammenhängende Graphen beschränken können: Sollte der Graph mehrere Komponenten besitzen, so kann natürlich nur in einer eine Rückwärtskante auftauchen.

   Diese Komponente besteht also aus den Baumkanten, einer Rückwärtskante und laut b) keiner Querkante. Angenommen die Komponente hat n Knoten. Laut a) besteht der Tiefensuchbaum aus $n - 1$ Kanten. Insgesamt gibt es also n Kanten.

   Jeder Tiefensuchbaum muss, da wir ihn als Spannbaum unseres Graphen auffassen können, $n - 1$ Baumkanten besitzen. Da der Graph n Kanten besitzt, muss die verbleibende Kante eine Rückwärtskante sein.

Aufgabe H24 (7 Punkte)
Sie haben Zuweisungen der folgenden Art:

```
y := 3*a*a     d := y + 3     a := c - x     x := 7     c := 2*x - 3
```

Gehen Sie davon aus, dass arithmetische Operationen mit nicht initialisierten Variablen einen Fehler werfen. Wie können Sie für beliebige Zuweisungen eine gültige Folge der Zuweisungen bestimmen, die keinen Fehler wirft? Wann ist dies nicht möglich?

Falls relevant, dürfen Sie annehmen, dass jeder Variable nur einmal ein Wert zugewiesen wird.

Lösungsvorschlag
Es herrschen Abhängigkeiten zwischen den Variablen. Wenn in einer Zuweisung für die Variable x der Wert der Variable y auftaucht, gibt es eine Kante zwischen der Zuweisung für y und der für x.

Falls der resultierende Graph ein DAG ist, können wir das Problem mittels topologischen Sortierens lösen.

Eine korrekte Sortierung für das Beispiel ist die folgende:

```
x := 7     c := 2*x - 3     a := c - x     y := 3*a*a     d := y + 3
```

Falls der resultierende Graph Kreise besitzt ist es nicht möglich die Operationen in eine fehlerfreie Reihenfolge zu bringen.

Aufgabe H25 (10 Punkte)
Ein *deterministischer Büchi-Automat* (DBA) ist ein Automat, welcher auf einem endlosen Eingabewort läuft und dieses genau dann akzeptiert, wenn während dieses Laufs ein Endzustand des Automaten unendlich oft besucht wird.

Formal ist dieser definiert durch $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$, mit der gleichen Bedeutung der Symbole wie bei einem DFA. Für jedes unendliche Wort $\alpha \in \Sigma^\omega$ gibt es einen eindeutigen Lauf ρ von $\mathfrak{A}$ auf α, definiert als $\rho(0) = q_0, \rho(i+1) = \delta(\rho(i), \alpha(i))$. Das Akzeptanzkriterium lautet dann: $\exists^\omega i : \rho(i) \in F$, wobei $\exists^\omega i$ zu lesen ist als *es existieren unendlich viele i*. Somit enthält die durch den Automaten erkannte Sprache genau alle unendlichen Worte, welche einen Lauf beschreiben, der mindestens einen Endzustand unendlich oft besucht.

Unter dieser Aufgabe finden Sie zwei Beispiele für solche Automaten.

Gegeben sei nun ein deterministischer Büchi-Automat $\mathfrak{A}$. Geben Sie einen Algorithmus an, mit dem Sie in Polynomialzeit feststellen können, ob die Sprache, die $\mathfrak{A}$ erkennt, mindestens ein Wort enthält.

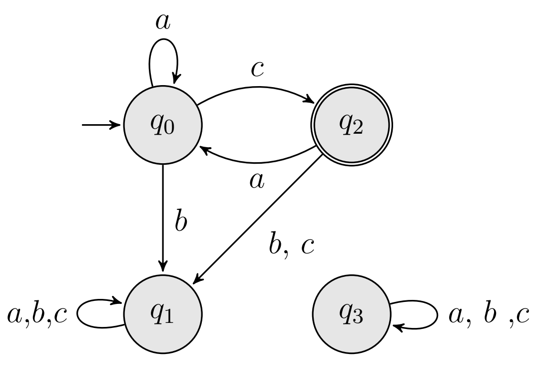


Abb. 1: Ein DBA für die Sprache $L = a^*(ca^+)^\omega$

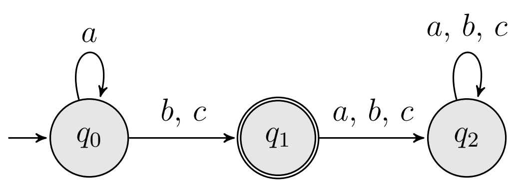


Abb. 2: Ein DBA für die Sprache $L = \emptyset$:

Lösungsvorschlag

Wir können alle starken Zusammenhangskomponenten des Automaten finden. Anschließend reicht es aus zu prüfen, ob mindestens eine der vom Startzustand erreichbaren Zusammenenhangskomponenten mindestens eine Kante (inklusive Selfloops auf Endzuständen) und einen Endzustand enthält. In diesem Fall existiert mindestens ein unendlicher Lauf, der in dieser Zusammenhangskomponente alle Knoten unendlich oft besuchen kann, von denen einer ein Endzustand ist. Somit ist das Wort, welches diesen beschreibt Teil der Sprache.