Topologische Sortierung von Graphen

[1] Bei einem Aufbausimulationsspiel kann man verschiedene Gebäude bauen. Jedes Gebäude produziert eine bestimmte Ware. Damit das Gebäude arbeiten kann, benötigt es seinerseits Waren anderer Gebäude als Rohstoff:

- Die Minen müssen mit Essen versorgt werden, das entweder Fleisch vom Metzger oder Brot vom Bäcker sein kann. Je nach Typ produzieren Minen Kohle oder Eisenerz.
- Die Eisenschmelze stellt daraus das vom Werkzeugmacher benötigte Eisen her.
- Der Werkzeugmacher heizt seine Schmiede mit Kohle und erzeugt aus dem Eisen Sensen für die Getreidefarm, Fleischermesser für den Metzger und Pickel für die Minen.
- Mit dem Getreide der Farm werden die Schweine gefüttert und in der Mühle das Mehl für den Bäcker gemahlen.
- Die Schweine werden vom Metzger geschlachtet.

Diese Abhängigkeiten machen es schwer zu entscheiden, in welcher Reihenfolge man die Gebäude bauen sollte, damit sie nicht ungenutzt herumstehen.

(A1)

Gib (wenn möglich) eine sinnvolle Baureihenfolge der Gebäude an.

Modellierung von Abhängigkeiten

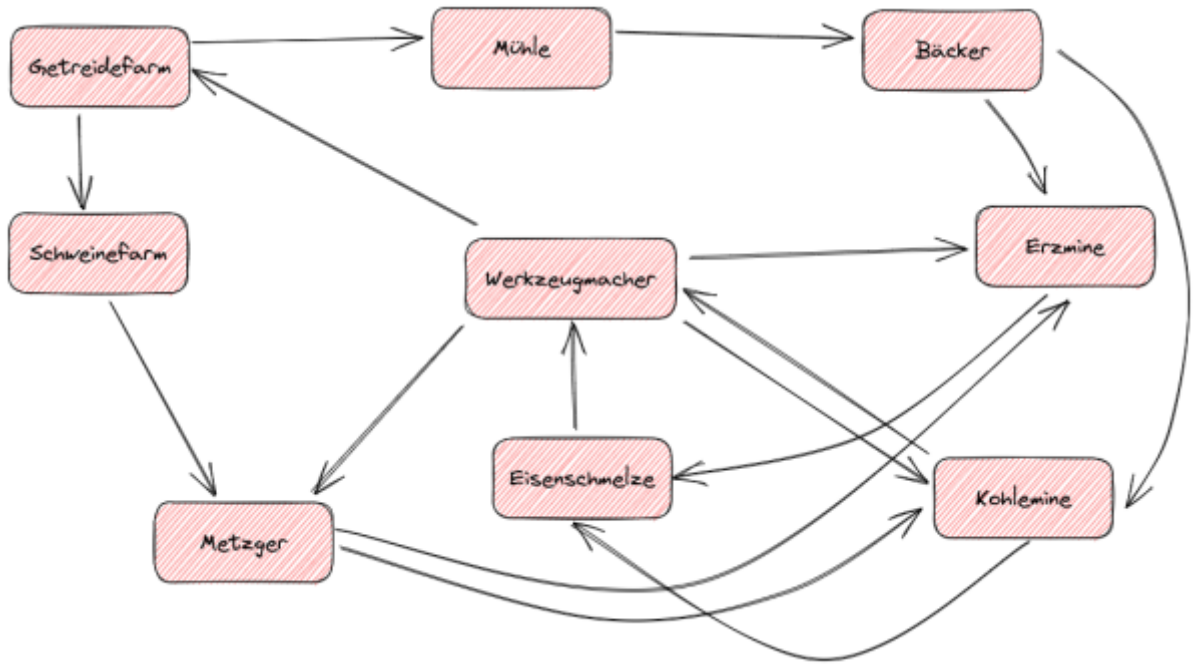
Die Problematik der gegenseitigen Abhängigkeiten tritt in vielen Situationen auf. Es wäre also von Vorteil, mit einem Algorithmus entscheiden zu können, ob eine sinnvolle Reihenfolge gefunden werden kann und wenn ja welche. Eine solche Reihenfolge nennt man **topologische Sortierung auf einem gerichteten Graphen**

(A2)

Modelliere die Situation im Aufbausimulationsspiel als gerichteter Graph.

- Was sind die Knoten, was die Kanten?
- Müssen die Kanten eine Richtung haben oder genügt es, Verbindungen herzustellen?
- Woran kannst du am Graphen erkennen, dass es für das Simulationsspiel keine topologische Sortierung gibt?

Lösung



Es gibt keine topologiosche Sortierung, weil es nicht möglich ist, einen Graph ohne Zyklus zu finden, der eine Reihenfolge für die Gebäude angibt. das Problem sind die Verbindungen vom Metzger und vom Bäcker zu den Minen, diese können nicht ersetzt werden, damit erhält man zwei Zyklen im Graphen, die man nicht vermeiden kann.

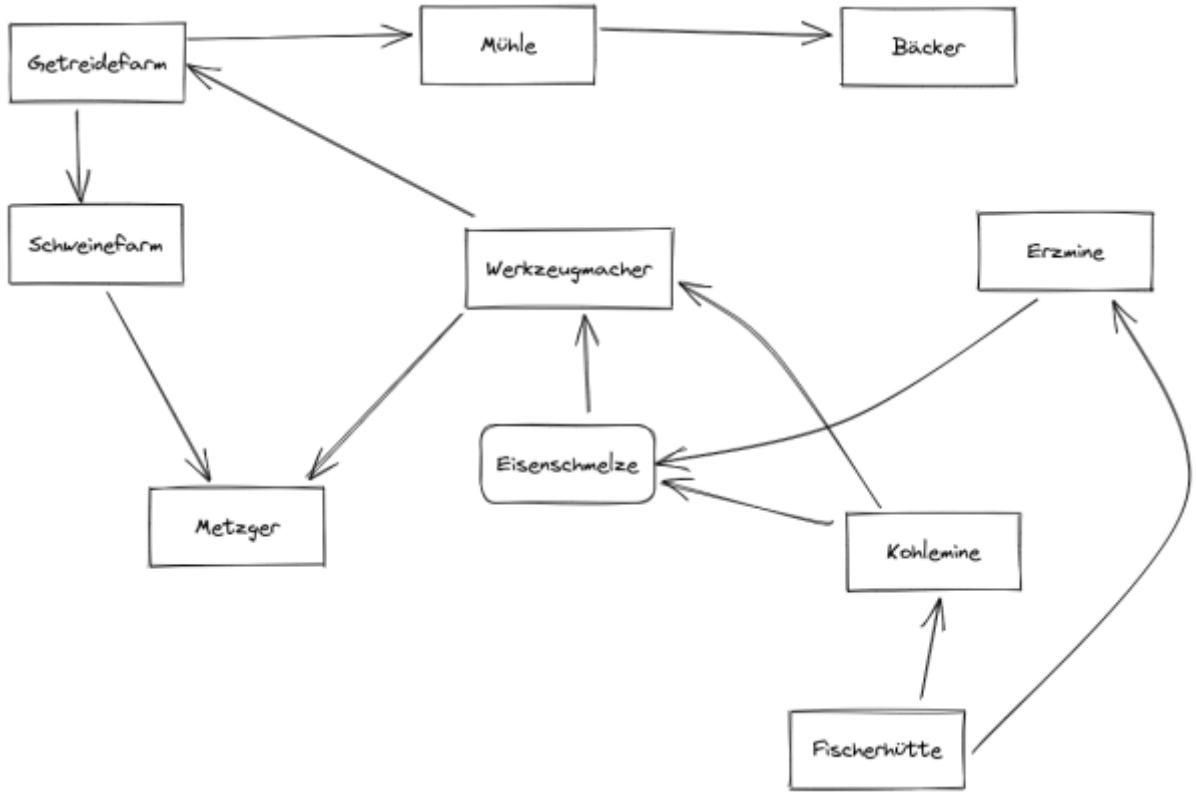
(A3)

Im Computerspiel gibt es zwei Möglichkeiten, *Verklemmungen* zu lösen.

- Weitere Gebäude werden eingeführt: z.B. eine Fischerhütte, die ohne Voraussetzung Essen (Fisch) produziert.
- Man erhält eine Anzahl von Anfangsgegenständen, mit denen man die Produktion in Gang bringen kann: z.B. 20 Pickel.

Gib an, welche Kanten des ursprünglichen Graphen (zumindest vorübergehend) entfernt werden können, wenn man die Fischerhütte hinzufügt und zu Beginn 20 Pickel zur Verfügung stehen. Zeige, dass so die Verklemmung gelöst wird und eine topologische Sortierung nun möglich ist. Zeichne den zu dieser topologischen Sortierung gehörigen Graphen.

Lösung



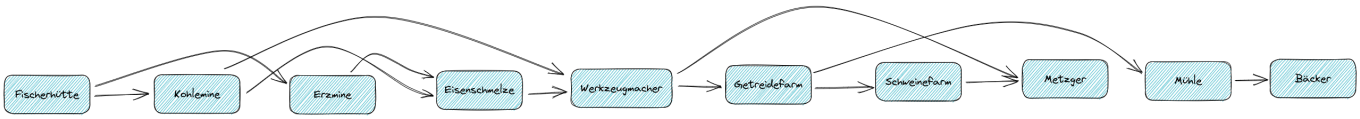
Definition: Topologische Sortierung, Reihenfolge

Wenn es bei einem gerichteten Graphen eine Abfolge der Knoten des Graphen gibt, so dass für jede Kante k gilt: Wenn k von A nach B geht, dann steht A vor B in der Abfolge, **dann existiert in diesem Graphen eine topologische Sortierung**.

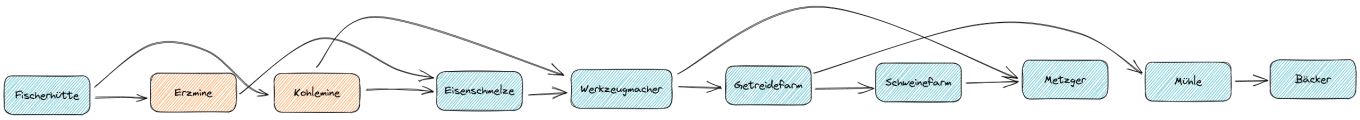
Um eine **Reihenfolge** anzugeben, muss man eine Abfolge der Knoten des Graphen angeben, so dass für jede Kante k gilt: Wenn k von A nach B geht, dann steht A vor B in der Abfolge.

Wichtig dabei ist, dass es innerhalb der Reihenfolge der Sortierung **nicht notwendig ist**, stets eine Kante von einem Knoten in der Sortierung zum nächsten zu haben. Es reicht, wenn man die Knoten von links nach rechts in einer Reihenfolge anordnen kann, so dass es keine Kanten gibt, die von

rechts nach links zeigen. So kann man den Lösungsgraphen aus der vorigen Aufgabe durch Verschieben der Knoten folgendermaßen anordnen:



Das stellt eine gültige topologische Sortierung dar, auch wenn es keine Kante von der Kohle- zur Erzmine gibt. Man erkennt auf diese Weise auch sofort, dass es mehr als eine topologische Sortierung eines gerichteten Graphen geben kann, die Reihenfolge ist also nicht eindeutig:



(Hands-On)

Du kannst das selbst ausprobieren, indem du die folgende Datei

aufbausimulation-graph-fischerhuette.zip

herunterlädst und auspackst. Du erhältst die Datei `Aufbausimulation-Graph-Fischerhuette.excalidraw`. Diese kannst du in Excalidraw öffnen, nun kannst du die Knoten des Graphen auf einer Achse anordnen, so dass es keine Kanten gibt, die von rechts nach links zeigen.

Zum Vergleich, ikannst du versuchen, dasselbe Ergebnis mit dem ursprünglichen Graphen (ohne Fischerhütte) zu erzielen:

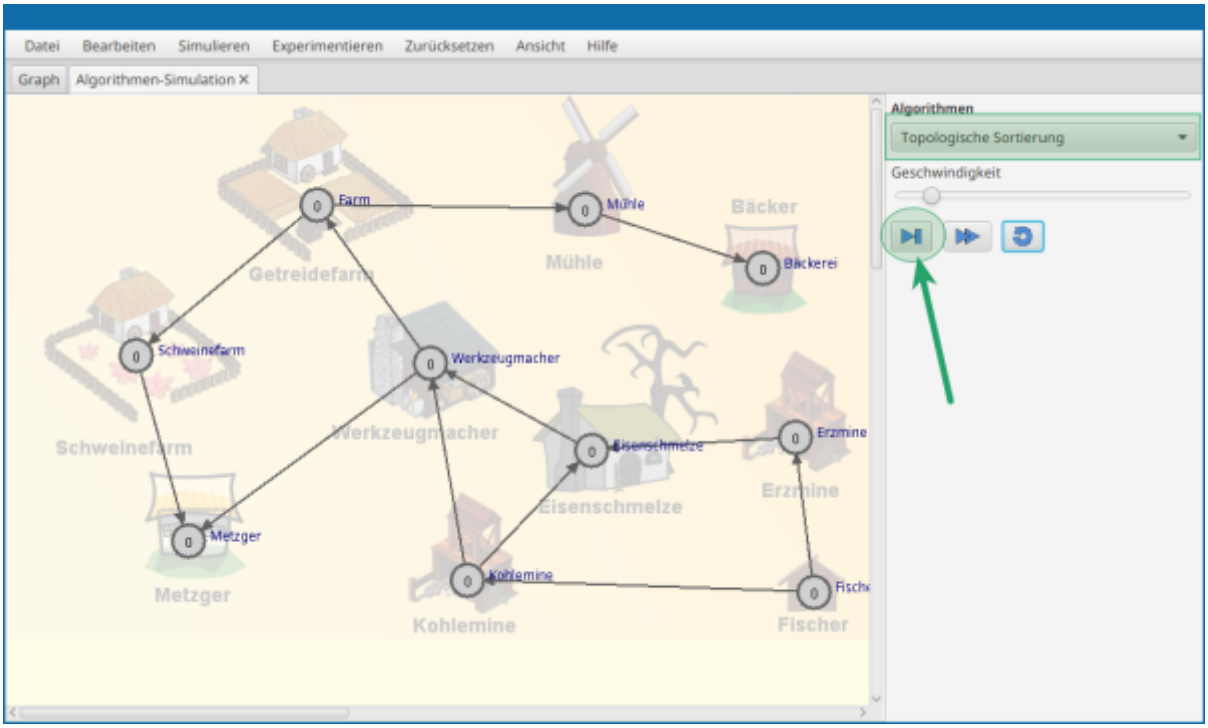
aufbausimulation-graph.zip

- man kann sehr gut erkennen, wie ein Zyklus im Graph eine topologische Sortierung unmöglich macht.

Algorithmus

(A4)

Öffne im Graphentester das Beispiel `02_topologischesortierung/02_aufbausimulation2.csv`. Untersuche die Situation mit Hilfe des bereits im Graphentester implementierten Algrithmus "Topologogische Sortierung".



Formuliere den Algorithmus als Pseudocode.

Hilfestellung: Verbale Beschreibung des Algorithmus

Bei jedem Knoten wird als Wert die Anzahl der eingehenden Kanten (Eingangsgrad) eingetragen. Das ist die Anzahl der noch unerfüllten Voraussetzungen. Danach wiederholt man folgendes:

- Suche einen noch unmarkierten Knoten, der den Wert 0 hat (= alle Voraussetzungen sind erfüllt). Füge diesen der topologischen Sortierung hinzu und markiere ihn als bearbeitet. Reduziere den Wert aller Knoten um 1, zu denen eine Kante vom aktuellen Knoten führt.
- Gibt es keinen Knoten mit dem Wert 0 und es sind noch nicht alle Knoten abgearbeitet, dann ist es nicht möglich, eine topologische Sortierung zu finden.

Lösung: Pseudocode

```
Topologische Sortierung:
Für jeden Knoten k des Graphen:
   setze den Wert des Knoten auf seinen Eingangsgrad

Setze topologische_Sortierung auf leer
Wiederhole solange es einen unmarkierten Knoten mit Wert 0 gibt
   Nimm einen unmarkierten Knoten k mit Wert 0
   Markiere k

   Füge k der topologische_Sortierung hinzu

   Für jeden Nachbarknoten n von k:
      reduziere den Wert des Knoten n um 1
Ende Wiederhole
```

```
Falls noch unmarkierte Knoten übrig sind:
  Ausgabe: es gibt keine topologische Sortierung
sonst:
  Ausgabe: topologische_Sortierung ist gefunden
```

(A5)

Implementiere den Algorithmus selbst in einer eigenen Klasse unter "eigene Algorithmen". Teste deinen Algorithmus an den Beispielen `03_beispiel_obst.csv` und `04_beispiel_obst2.csv`.

Lösungsvorschlag Algorithmus

```
// Reihenfolge
String Reihenfolge = "";
// Hole alle Knoten vom Graph g
List<Knoten> alleKnoten = g.getAlleKnoten();

// Schleife über alle Knoten
for(Knoten k: alleKnoten) {
    int Eingangsgrad = g.getEingehendeKanten(k).size();
    k.setWert(Eingangsgrad);
    k.setFarbe(7);
}

step();
boolean sort=true;


while(alleKnoten.size() > 0 && sort) {
    Collections.sort(alleKnoten);
    Knoten k = alleKnoten.remove(0);
    if (k.getIntWert() == 0) {
        k.setFarbe(4);
        Reihenfolge += k.getInfotext() + " ";
        for(Knoten n: g.getNachbarknoten(k)) {
            n.setWert(n.getIntWert()-1);
        }
     } else {
        sort=false;
     }
     step();
}

if (! sort) {
```

```
    System.out.println("Keine Sortierung möglich!");
} else {
    System.out.println(Reihenfolge);
}
```

Dateien

2024-03-19_15-44.png	85.5 KiB	19.03.2024 14:46
aufbausim.png	389.4 KiB	14.11.2022 18:40
aufbausimulation-graph-fischerhuette-geordnet.png	292.1 KiB	15.11.2022 10:44
aufbausimulation-graph-fischerhuette-geordnet2.png	300.3 KiB	15.11.2022 10:50
aufbausimulation-graph-fischerhuette.png	102.0 KiB	14.11.2022 20:09
aufbausimulation-graph-fischerhuette.zip	4.3 KiB	15.11.2022 10:54
aufbausimulation-graph.png	302.7 KiB	11.06.2024 13:20
aufbausimulation-graph.zip	4.5 KiB	15.11.2022 11:02
biberkochen.png	72.4 KiB	17.11.2022 06:53
einstieg_toposort.odp	83.7 KiB	17.11.2022 06:26
einstieg_toposort.pdf	81.5 KiB	17.11.2022 06:26
gruppenarbeit.png	127.1 KiB	17.11.2022 07:05
tabelle.png	42.9 KiB	17.11.2022 06:45
ts_gt.png	227.8 KiB	15.11.2022 08:57

1)

Grafik(en): Warenverkehr bei einer Aufbausimulation (alle Bilder Pixabay-Lizenz - kein Nachweis notwendig)