

Lösung Übungsblatt 5

Christoph van Heteren-Frese (Matr.-Nr.: 4465677)

Sven Wildermann (Matr.-Nr.: 4567553)

Tutor: Alexander Steen, eingereicht am 23. Mai 2013

Aufgabe 1

1. Mit Hilfe des Tiefensuche-Algorithmus kann ein Graph auf Zyklensfreiheit untersucht werden. Hierzu überprüft man mit diesem Algorithmus, ob der Graph G eine Rückwärtskante besitzt oder nicht. Methode: Erweitere DFS um Speicherung der Rückwärtskanten (Back-Kante / B-Kante), sobald man von "v" aus auf einem markierten Knoten "u" trifft, hat man eine B-Kante gefunden, falls "u" nicht der direkte Vorgänger von "v" ist.

2. Algorithmus:

```
1      function ISACYCLIC(GRAPH G=(V,E)): bool {
2          B:={}
3          for all v in V do { marked[v]:=false; p(v):=nil}
4          for all v in V do {
5              if not marked[v] then
6                  DFS-ACYCLIC(v)
7              }
8          if B:={} then return false else return true
9      }
10
11     procedure DFS-ACYCLIC(Node v){
12         marked[v]:=true
13         for all w in N(v) do
14             if not marked[w] then
15                 p(w):=v
16                 DFS-ACYCLIC(w)
17             else if p(v) != w then
18                 B:=B + (v,w)    // + ist hier die Mengenvereinigung
19     }
```

Komplexität: Da jede Kante und jeder Knoten genau einmal besucht wird, beträgt die Laufzeit von Tiefensuche $O(\#V + \#E)$, wobei $\#V$ = Anzahl der Knoten und $\#E$ = Anzahl der Kanten.

Aufgabe 2

TODO CHRIS !

Aufgabe 3

Diese Lösung des Leser-Schreiber-Problems funktioniert nicht. Ein Gegenbeispiel:

- R1 möchte auf die Datei F1 lesen und führt daher die Funktion ReaderIn (erfolgreich) aus.
- Nun möchte (während R1 liest) W1 auf die Datei F1 schreibend zugreifen. Da nun aber die Bedingung (nR größer 0) erfüllt ist, wartet W1 indem es `c.Wait()` ausführt.
- Hinweis: Diese Wait-Anweisung befindet sich innerhalb des Blocks, in dem `m.Lock` aktiv ist.
- `m.Unlock` wird von W1 also solange nicht ausgeführt wie R1 noch liest.
- Jetzt möchte R1 den Lesevorgang auf F1 wieder beenden, in dem es `ReaderOut()` ausführt.
- `ReaderOut()` kann allerdings nicht ausgeführt werden, da `m.Lock()` nicht ausgeführt werden kann - weil W1 diesen Lock nicht wieder freigegeben hat.
- **Damit ist ein Livelock entstanden. Der Lesezugriff auf die Datei kann ohne Neustart des Systems nicht mehr geschlossen werden. Eine Dateiveränderung ist mit dieser Lösung nicht mehr möglich, sobald auch nur ein Thread lesenden Zugriff auf die Datei hat.**

Aufgabe 4

TODO CHRIS !