Technische Universität Chemnitz Wintersemester 2001/02
Professur Theoretische Informatik
Prof. Dr. A. Goerdt, Dipl.-Inf. F. Schädlich

6. Übung „Theoretische Informatik I“
Lösungsvorschläge

Aufgabe 6.1:

Ob eine Kante nur einmal ausgegeben oder einmal auf den Stack gelegt wird ist letztendlich egal. Wie in Aufgabe 4.2 findet man:

Frage nach der Farbe von v:

- wenn weiß, so neu entdeckt, Kante kann noch nicht im Keller sein,
- wenn schwarz, so schon fertig, Kante muß schon im Keller sein,
- wenn grau, so eher gefunden als v und entweder Vater von u, Kante ist schon im Keller, oder nicht, dann kann Kante noch nicht im Keller sein.

Aufgabe 6.2:

Beide Aussagen sind falsch, siehe z. B. Aufgabe 6.3.

Aufgabe 6.3:

Wir legen die alphabetische Reihenfolge der Knoten zugrunde.

1. $d[a] = 1$, $low[a] = 1$.
2. $d[b] = 2$, $low[b] = 2$, Keller=$(\{a, b\})$.
3. $d[c] = 3$, $low[c] = 3$, Keller=$(\{b, c\}, \{a, b\})$.
4. $d[d] = 4$, $low[d] = 4$, Keller=$(\{c, d\}, \{b, c\}, \{a, b\})$.
5. $low[d] = 1$, Keller= $(\{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$,
   $low[d] = 1$, Keller= $(\{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$,
   $f[d] = 5$, Keller= $(\{c, a\}, \{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$.
6. $d[g] = 6$, $low[g] = 6$, Keller=$(\{c, g\}, \{c, a\}, \{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$.
7. $d[e] = 7$, $low[e] = 7$, Keller=$(\{g, e\}, \{c, g\}, \{c, a\}, \{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$.
8. $d[f] = 8$, $low[f] = 8$, Keller=$(\{e, f\}, \{g, e\}, \{c, g\}, \{c, a\}, \{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$.
9. $low[f] = 6$, Keller=$(\{f, g\}, \{e, f\}, \{g, e\}, \{c, g\}, \{c, a\}, \{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$.
   $f[f] = 9$.
10. $low[e] = 6$, $f[e] = 10$.
11. Ausgabe der Komponente $(\{f, g\}, \{e, f\}, \{g, e\})$,
    Keller=$(\{c, g\}, \{c, a\}, \{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$,
    $d[h] = 11$, $low[h] = 11$, Keller=$(\{g, h\}, \{c, g\}, \{c, a\}, \{d, b\}, \{d, a\}, \{c, d\}, \{b, c\}, \{a, b\})$.

12. $d[i] = 12$, $low[i] = 12$, Keller=$(\{h,i\},\{g,h\},\{c,g\},\{c,a\},\{d,b\},\{d,a\},\{c,d\},\{b,c\},\{a,b\})$.
13. Keller=$(\{i,g\},\{h,i\},\{g,h\},\{c,g\},\{c,a\},\{d,b\},\{d,a\},\{c,d\},\{b,c\},\{a,b\})$, $low[i] = 6$, $f[i] = 13$.
14. $low[h] = 6$, $d[j] = 14$, $low[j] = 14$, Keller=$(\{h,j\},\{i,g\},\{h,i\},\{g,h\},\{c,g\},\{c,a\},\{d,b\},\{d,a\},\{c,d\},\{b,c\},\{a,b\})$.
15. $f[j] = 15$.
16. Ausgabe der Komponente $(\{h,j\})$, Keller= $(\{i,g\},\{h,i\},\{g,h\},\{c,g\},\{c,a\},\{d,b\},\{d,a\},\{c,d\},\{b,c\},\{a,b\})$, $low[h] = 6$, $f[h] = 16$.
17. Ausgabe der Komponente $(\{i,g\},\{h,i\},\{g,h\})$, Keller=$(\{c,g\},\{c,a\},\{d,b\},\{d,a\},\{c,d\},\{b,c\},\{a,b\})$, $f[g] = 17$.
18. Ausgabe der Komponente $(\{c,g\})$, Keller=$(\{c,a\},\{d,b\},\{d,a\},\{c,d\},\{b,c\},\{a,b\})$, $f[c] = 18$.
19. $f[b] = 19$.
20. Ausgabe der Komponente $(\{c,a\},\{d,b\},\{d,a\},\{c,d\},\{b,c\},\{a,b\})$, Keller leer, $f[a] = 20$.

Aufgabe 6.4:

Hier können wir wieder eine topologischen Sortierung nutzen. Links fangen wir mit 0 an, jede folgende Generation bekommt 1 mehr, die entferntesten Blätter geben die Höhe des Graphen an.

```
L := topsort(G)
for i := 1 to |V| do
    hoehe(i) := 0
    weg(i) := i
l := head(L); L := tail(L)
while L ≠ nil do
    for each v ∈ adj[l] do
        if hoehe(l) + 1 > hoehe(v) then
            hoehe(v) := hoehe(l) + 1
            weg(v) := weg(l) ∘ v
    l := head(l); L := tail(L)
max := 0; hoehe(0) := 0
for i := 1 to |V| do
    if hoehe(v) > hoehe(max) then max := v
return weg(max) ist längster Pfad (mit der Länge hoehe(max))
```

Komplexität: TopSort und Tiefensuche (für die Hauptschleife) addiert, also $O(|V| + |E|)$.

Aufgabe 6.5:

$$\begin{pmatrix} 0 & 3 & \infty & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}_0 \Longrightarrow \begin{pmatrix} 0 & 3 & \infty & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}_1 \Longrightarrow \begin{pmatrix} 0 & 3 & \infty & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & 5 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}_2 \Longrightarrow$$

$$\begin{pmatrix} 0 & 3 & \infty & 4 & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & 5 & 11 \\ 2 & -1 & -5 & 0 & -2 \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}_3 \Longrightarrow \begin{pmatrix} 0 & 3 & -1 & 4 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}_4 \Longrightarrow \begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}_5$$

Aufgabe 6.6:

Zwischen zwei Knoten a und b des Graphen wurden folgende Wege untersucht:

- $a - b$

- $a - 1 - b$

- $a - 2 - b$
- $a - 1 - 2 - b$
- $a - 2 - 1 - b$
- $a - 1 - 2 - 1 - b$ *

- $a - 3 - b$
- $a - 1 - 3 - b$
- $a - 2 - 3 - b$
- $a - 1 - 2 - 3 - b$
- $a - 2 - 1 - 3 - b$
- $a - 1 - 2 - 1 - 3 - b$ *

- $a - 3 - 1 - b$
- $a - 1 - 3 - 1 - b$
- $a - 2 - 3 - 1 - b$
- $a - 1 - 2 - 3 - 1 - b$ *
- $a - 2 - 1 - 3 - 1 - b$ *
- $a - 1 - 2 - 1 - 3 - 1 - b$ *

- $a - 3 - 2 - b$
- $a - 1 - 3 - 2 - b$
- $a - 2 - 3 - 2 - b$ *
- $a - 1 - 2 - 3 - 2 - b$ *
- $a - 2 - 1 - 3 - 2 - b$ *
- $a - 1 - 2 - 1 - 3 - 2 - b$ *

- $a - 3 - 1 - 2 - b$
- $a - 1 - 3 - 1 - 2 - b$ *
- $a - 2 - 3 - 1 - 2 - b$ *
- $a - 1 - 2 - 3 - 1 - 2 - b$ *
- $a - 2 - 1 - 3 - 1 - 2 - b$ *
- $a - 1 - 2 - 1 - 3 - 1 - 2 - b$ *

- $a - 3 - 2 - 1 - b$
- $a - 1 - 3 - 2 - 1 - b$ *
- $a - 2 - 3 - 2 - 1 - b$ *
- $a - 1 - 2 - 3 - 2 - 1 - b$ *
- $a - 2 - 1 - 3 - 2 - 1 - b$ *
- $a - 1 - 2 - 1 - 3 - 2 - 1 - b$ *

- $a - 3 - 1 - 2 - 1 - b$ *
- $a - 1 - 3 - 1 - 2 - 1 - b$ *
- $a - 2 - 3 - 1 - 2 - 1 - b$ *
- $a - 1 - 2 - 3 - 1 - 2 - 1 - b$ *
- $a - 2 - 1 - 3 - 1 - 2 - 1 - b$ *
- $a - 1 - 2 - 1 - 3 - 1 - 2 - 1 - b$ *

Die mit * gekennzeichneten Wege sind nicht von Interesse, da hier Knoten doppelt auftreten. Voraussetzung für das korrekte Arbeiten des Floyd–Warshall–Algorithmus ist das Nichtvorhandensein von Kreisen negativer Länge. Damit können diese Wege, die einen Knoten doppelt und damit einen Kreis enthalten, nicht minimal sein.