Lösungen der Hausaufgaben von Übungsblatt 5

Algorithmen und Datenstrukturen (WS 2013, Ulrike von Luxburg)

Lösungen zu Aufgabe 1

Zum Ende eines jeden Durchlaufs der äußeren Schleife muss geprüft werden, ob sich gegenüber dem Beginn desselben Durchlaufs irgendein Wert $v.dist$ geändert hat. Falls nicht, so kann sich auch in keinem weiteren Durchlauf irgendein Wert $v.dist$ ändern, d.h., der Algorithmus kann sofort abgebrochen werden (insbesondere braucht nicht noch auf negative Zyklen überprüft zu werden, da solche nach Voraussetzung nicht existieren). Da die Ausgabe nach diesem vorzeitigen Abbruch mit der des unmodifizierten Algorithmus übereinstimmt, wurde trotzdem zu jedem erreichbaren Zielknoten ein kürzester Pfad gefunden. Dieser Fall tritt nach genau $m+1$ Durchläufen der äußeren Schleife auf, ohne dass m a priori bekannt sein muss.

Lösungen zu Aufgabe 2

Da G ein DAG ist, können wir G in Zeit $\mathcal{O}(|V|+|E|)$ topologisch sortieren. Also gelte nun oBdA, dass $V = \{1, \dots, n\}$ und dass für jede Kante $(i, j) \in E$ gilt $i \leq j$. Die folgenden Zeilen liefern nun alle kürzesten Pfade ausgehend von einem beliebigen $s \in V$, wobei abschließend für jeden Knoten $v \in V$ in $v.dist$ die Distanz zu s steht, sowie in $v.\pi$ der Vorgänger entlang eines kürzesten Pfades von s nach v.

Setze $v.dist := \infty$ und $v.\pi :=$ NIL für alle $v \in V$
Setze $s.dist := 0$
for $i = s$ **to** $n-1$ **do**
  for each $j \in N(i)$ **do** ▷ $N(i) := \{j \in V \mid (i, j) \in E\}$
    if $i.dist + w(i, j) < j.dist$ **then**
      Setze $j.dist := i.dist + w(i, j)$ und $j.\pi := i$
    end if
  end for
end for

Dies entspricht im Wesentlichen dem Dijkstra-Algorithmus, allerdings ist die Reihenfolge der Knotenbesuche aufgrund der topologischen Sortierung klar durch "von s bis $n-1$" festgelegt. Laufzeit hierfür ist $\mathcal{O}(\max\{|V|, |E|\}) = \mathcal{O}(|V|+|E|)$, insgesamt also ebenfalls $\mathcal{O}(|V|+|E|)$.

Lösungen zu Aufgabe 3

Da nur aus s ausgehende Kanten negative Gewichte haben können, und G keine negativen Zyklen hat, traversiert jeder (vom Dijkstra-Algorithmus ermittelte) aus s ausgehende kürzeste Pfad ab seiner zweiten Kante stets nur nicht-negative Kanten.

Der Korrektheitsbeweis (Folie ≈ 249) gilt für solches G unverändert. Kritisch ist nur die Aussage "The path via x and y can only be shorter than the one via v if $w(x, y) < w(v, u)$", welche aber weiterhin gilt, da der Teilpfad von y zu u nicht-negatives Gewicht haben muss (weil nur aus s ausgehende Kanten negatives Gewicht haben können, aber s nicht auf einem kürzesten Pfad von y nach u liegen kann).

Lösungen zu Aufgabe 4

(a) Falls keine Wurzel gegeben ist, so wählen wir einen beliebigen Knoten als Wurzel. Ein Knoten ist ein Blatt, wenn er inzident zu nur einer einzigen Kante ist (auch die Wurzel selbst kann daher ein Blatt sein!). Ein längster kürzester Pfad muss von einem Blatt zu einem anderen Blatt führen, da ansonsten der Pfad an einem Ende (bis zu einem Blatt) entlang einer im Pfad nicht genutzten Kante verlängert werden könnte.

Außerdem gilt, dass ein längster kürzester Pfad von einem Blatt ausgehen muss, welches am weitesten von der Wurzel entfernt liegt. (Beweis: Sei $(a, \dots, t, \dots, b)$ ein kürzester Pfad von Blatt a zu Blatt b, worin t am nächsten an der Wurzel w liegt. Sei nun $d(w, c) > \max\{d(w, a), d(w, b)\}$. Zumindest einer der kürzesten Pfade $(c, \dots, a)$ oder $(c, \dots, b)$ muss über t führen. OBdA sei dies der Pfad von c nach b, also $(c, \ldots, t, \ldots, b)$. Dieser ist dann aber länger als $(a, \ldots, t, \ldots, b)$, weil $\ell(c, \ldots, t) > \ell(a, \ldots, t))$.

Dies liefert die Korrektheit des folgenden Algorithmus:

(i) Nutze einen Treewalk gestartet an Wurzel w, und speichere dabei die Distanzen aller Knoten zur Wurzel. Laufzeit $\mathcal{O}(|V|)$.

(ii) Bestimme einen (Blatt-)Knoten a, der am weitesten von der Wurzel entfernt liegt. Laufzeit $\mathcal{O}(|V|)$.

(iii) Nutze nun einen Treewalk gestartet an a, um die Distanzen aller Knoten zu a zu bestimmen. Wie bei (i) Laufzeit $\mathcal{O}(|V|)$.

(iv) Die maximale Distanz $\max_{b \in V} d(a, b)$ ist der Durchmesser des Graphen. Laufzeit $\mathcal{O}(|V|)$.

(b) Bestimme zunächst für alle Paare von Knoten einen kürzesten Pfad, zum Beispiel durch $|V|$ Aufrufe des naiven Dijkstra-Algorithmus mit jeweils Laufzeit $\mathcal{O}(|V||E|)$, insgesamt also Laufzeit $\mathcal{O}(|V|^2|E|)$. Ermittle dabei ein Knotenpaar in maximaler Distanz, welches somit den Durchmesser liefert.

Lösungen zu Aufgabe 5

(a) Wir betrachten den vollständigen gerichteten Graphen $G = (V, E)$ mit V der Menge der Währungen, in welchem jede Kante (i, j) mit $-\log(r_{i,j})$ gewichtet ist (stets definiert, da $r_{ij} \neq 0$). Es gilt:

$$-\log(r_{ij}) \begin{cases} < 0 & r_{ij} > 1 \\ = 0 & r_{ij} = 1 \\ > 0 & r_{ij} \in (0, 1) \end{cases} .$$

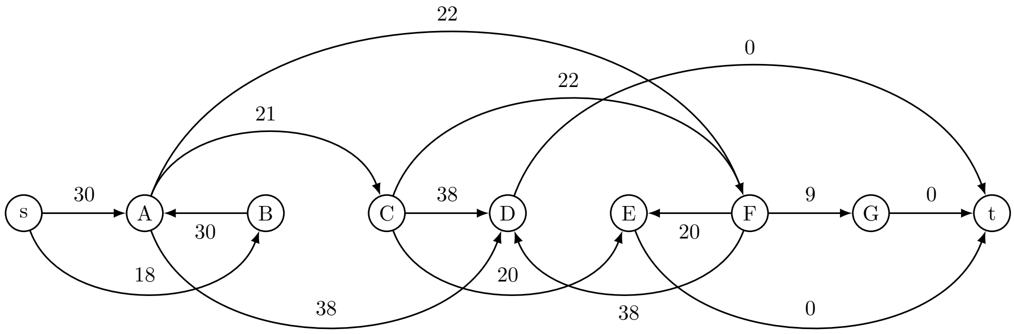
Den resultierenden Gesamt-Wechselkurs jeder Sequenz mehrfacher Währungswechsel (z.B. Währung i nach j, dann j nach k), erhält man durch das Produkt der einzelnen Wechselkurse (z.B. $r_{ij} \cdot r_{jk}$). Dessen negierter Logarithmus ist die Aufsummierung der einzelnen negierten Logarithmen (z.B. $-\log(r_{ij}r_{jk}) = -\log(r_{ij}) - \log(r_{jk})$), und damit die Länge des entsprechenden Pfades in G. Eine Währungsarbitrage tritt also genau dann ein, wenn ein negativer Zyklus existiert (d.h., ein zyklischer Gesamt-Wechselkurs größer als 1).

Mögliches Vorgehen also: Bestimme mittels Bellman-Ford in G für irgendeinen Startknoten v die kürzesten Wege zu allen anderen Knoten. Die Ausgabe ist `false` genau dann, wenn G einen negativen Zyklus enthält (also eine Währungsarbitrage).

(b) In obigem Verfahren entsprechen die negativen Zyklen genau den Sequenzen elementarer Währungswechsel, welche eine Währungsarbitrage ermöglichen.

Lösungen zu Aufgabe 6

Wir konstruieren einen Graphen $G = (V \cup \{s, t\}, E)$ mit V der Menge der Mitarbeiter, sowie den Dummy-Knoten s zum Zeitraum $s.start = s.ende = 9$ und t zum Zeitraum $t.start = t.ende = 17$. Eine gerichtete Kante (i, j) liegt vor, wenn $i.ende \in [j.start, j.ende]$, d.h., wenn die Mitarbeiter i und j den Zeitraum von $i.start$ bis $j.ende$ lückenlos abdecken können. Zudem ist (i, j) mit den Kosten von j gewichtet (wobei die Kosten von s und t als 0 angenommen werden). In diesem Graphen entspricht somit jeder Pfad von s nach t einem Schedule von Mitarbeitern, der den Zeitraum 9 Uhr bis 17 Uhr lückenlos abdeckt, wobei die Kosten des Pfades die Gesamtkosten dieses Schedules wiedergeben. Damit ist die gesuchte Lösung ein kürzester Pfad von s nach t in folgendem Graphen G:



Da (s, A, F, G, t) kürzester Pfad von s nach t, ist die Wahl $\{A, F, G\}$ kostenoptimal.