ALGORITMIEK: opgaven werkcollege 12

Gretige algoritmen

Opgave 1. (zie tweede editie, opgave 9.1.4)
Ontwerp een gretig algoritme voor het toewijzingsprobleem (dat we kennen uit paragraaf 3.4 **(ook in tweede editie)**). Levert je gretige algoritme altijd een optimale oplossing op?

Opgave 2. Exercise 9.1.5. **(ook in tweede editie)** uit het boek van Levitin.
Bepaal eerst voor het voorbeeldgeval (Levitin, exercise 1.2.2 **(ook in tweede editie)**) een oplossing. Voor het algemene geval mag je aannemen dat $t_1 \leq t_2 \leq \cdots \leq t_n$. Bedenk een gretige strategie en bereken de totale tijd die het kost om de brug over te steken als je die strategie gebruikt. Ga na of de gretige methode altijd een optimale oplossing oplevert.

Opgave 3. Exercise 9.1.8.a. **(zie tweede editie, opgave 9.1.6.a.)** uit het boek van Levitin. Voor de liefhebbers ook 9.1.8.b. **(zie tweede editie, opgave 9.1.6.b.)**

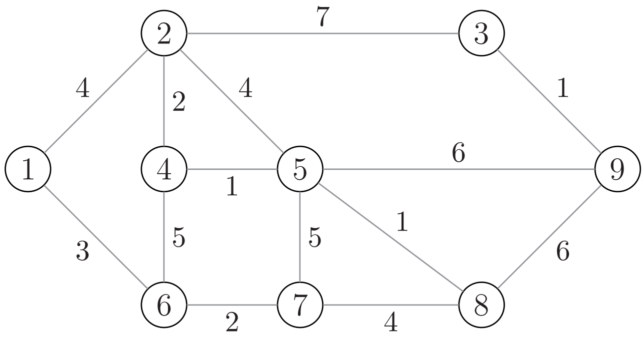
Opgave a. komt erop neer dat je met n positieve gehele getallen een zo groot mogelijke aaneengesloten reeks getallen moet kunnen maken door ze op alle mogelijke manieren op te tellen. De bedoeling is om de optimale rij "gewichten" op een gretige manier op te bouwen.
Bij b. kun je ook aftrekken, door sommige gewichten op de linkerschaal van de balans te zetten, en andere op de rechterschaal. Zo kun je bijvoorbeeld een voorwerp met gewicht 2 wegen met behulp van de gewichten $w_1 = 1$ en $w_2 = 3$ door w_1 bij het voorwerp op de ene schaal te plaatsen en gewicht w_2 op de andere.

Opgave 4. Exercise 9.3.1.a,b,c **(ook in tweede editie)** uit het boek van Levitin.

Opmerking: zie voor het algoritme van Dijkstra ook de sheets van het hoorcollege en hieronder. In het algoritme op de sheets worden alleen de kortste afstanden bepaald, maar de kortste paden zelf zijn eenvoudig te verkrijgen via een simpele aanpassing. Sla op de plek waar de labels worden aangepast de tak (v^*, v) op (danwel overschrijf de oude tak 'naar' v) indien pad[v] wordt aangepast tot pad[v^*]+ gewicht(v^*, v). (Zie onderstaande versie van het algoritme van Dijkstra.) Het pad via v^*, met als laatste tak (v^*, v), is dan blijkbaar korter dan het tot dusver gevonden pad van s naar v via knopen van de oorspronkelijke U. Wanneer de knoop v erbij wordt gekozen in U (v heeft op dat moment de minimale pad-waarde en wordt dan in het algoritme met v^* aangeduid), wordt de kandidaattak definitief. Deze tak maakt dan deel uit van het kortste pad van s naar v. Na afloop vormen al deze takken samen precies de boom bestaande uit alle kortste paden. In het boek worden niet de takken, maar de knoop waar je vandaan komt onthouden. Dit komt natuurlijk op hetzelfde neer.

Opgave 5. Exercise 9.3.2. **(ook in tweede editie)** uit het boek van Levitin.
Voer het algoritme uit zoals bij het voorbeeld op de sheets, door in de graaf steeds de kandidaatknopen met hun labels te markeren, en labels aan te passen indien nodig. Bouw ook de boom van kortste paden op, door samen met de keuze van v^* ook de tak die op het kortste pad ligt en v^* met de oorspronkelijke U verbindt, te markeren.

Opgave 6. Voer het algoritme van Dijkstra uit op de voorbeeldgraaf hieronder. Bepaal de (lengtes van de) kortste paden vanuit knoop 1. Ter controle van je antwoord kun je de bijbehorende sheet (uit hoorcollege 11) bekijken, waarop de resulterende boom van kortste paden staat.



Opgave 7. **(zie tweede editie, opgave 9.3.3.)** Geef een voorbeeld waaruit blijkt dat het algoritme van Dijkstra niet altijd werkt voor een samenhangende gewogen graaf met negatieve gewichten. Dat wil zeggen: geef een samenhangende gewogen graaf met minstens één negatief gewicht, waarbij het algoritme van Dijkstra vanuit een bepaalde startknoop niet voor alle andere knopen het kortste pad vindt.
Een graaf met 3 knopen is al genoeg.

Het algoritme van Dijkstra

// invoer: samenhangende gewogen graaf $G = (V, E)$ en startknoop s
// uitvoer: array dat de lengtes van de kortste paden vanuit s bevat;
// na afloop is pad$[v]$ = de lengte van een kortste pad van s naar v
// genereert ook de takken van de kortste paden boom

for $v \in V$ **do**
  pad$[v] := \infty$;
od
pad$[s] := 0$;
$U := \emptyset$;
while ($U \neq V$) **do**
  vind knoop $v^* \in V \setminus U$ met pad$[v^*]$ minimaal;
  $U := U \cup \{v^*\}$;
  for alle knopen v aangrenzend aan v^* **do**
    if pad$[v^*]+$ gewicht$(v^*, v) <$ pad$[v]$ **then**
      pad$[v] :=$ pad$[v^*]+$ gewicht(v^*, v);
      nieuwe kandidaattak voor v: (v^*, v)
    fi
  od
od