ALGORITMIEK: opgaven werkcollege 10

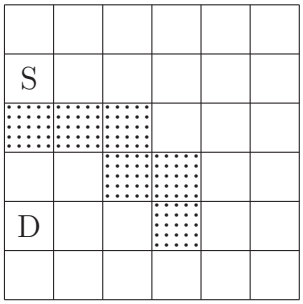
Gretige algoritmen

Opgave 1.
Ontwerp een gretig algoritme voor het toewijzingsprobleem (dat we kennen uit paragraaf 3.4). Levert je gretige algoritme altijd een optimale oplossing op?

Opgave 2. (Opgave 1.2.2 en opgave 9.1.5 uit het boek van Levitin.)
New World puzzle Er zijn vier mensen die een gammele brug willen oversteken; ze beginnen allemaal aan dezelfde kant. Je hebt 17 minuten om ze allemaal naar de overkant te krijgen. Het is nacht, en ze hebben één zaklamp. Er kunnen hoogstens twee mensen tegelijk de brug oversteken. Wanneer mensen de brug oversteken, of het er nu één is of twee zijn, moeten ze de zaklamp bij zich hebben. De zaklamp moet heen en weer meegenomen worden; hij kan bijvoorbeeld niet gegooid worden.
Persoon 1 heeft één minuut nodig om de brug over te steken, persoon 2 twee minuten, persoon 3 vijf minuten en persoon 4 tien minuten. Als twee mensen samen oversteken, lopen ze met de snelheid van de langzaamste. (N.B.: Volgens een gerucht op internet werd dit probleem bij een bekend softwarebedrijf in de buurt van Seattle aan sollicitanten gegeven.)
a. Bepaal voor het voorbeeldgeval hierboven een oplossing.
b. Beschouw nu het algemene geval, waarin we $n > 1$ mensen hebben, met oversteektijden $t_1, t_2, \ldots, t_n$. Alle andere voorwaarden in het probleem zijn hetzelfde. Je mag aannemen dat $t_1 \leq t_2 \leq \cdots \leq t_n$.
Bedenk een gretig algoritme voor dit probleem en bereken de totale tijd die het kost om de brug over te steken als je dat algoritme gebruikt. Ga na of het gretige algoritme altijd een optimale oplossing oplevert. Zo ja, toon dit aan. Zo nee, geef een instantie met zo weinig mogelijk mensen waaruit dit blijkt.

Opgave 3. (Opgave 9.1.8 uit het boek van Levitin.)
Bachet's gewichtenprobleem Bepaal een optimale rij van n gewichten $\{w_1, w_2, \ldots, w_n\}$ zodat het mogelijk is om met een balans elk voorwerp met een integer gewicht tussen 1 en W te wegen, voor een zo groot mogelijk getal W, ervanuitgaande dat
a. de gewichten alleen op de vrije schaal van de balans gezet kunnen worden (dus waar het te wegen voorwerp niet staat). Dit 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.
b. (voor de liefhebbers) de gewichten op beide schalen van de balans gezet kunnen worden. Nu kun je dus 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. Beschouw het volgende 6x6 bord:



Op dit bord kun je in één stap van een vakje naar een aangrenzend vakje (horizontaal, verticaal, diagonaal) lopen. De gearceerde vakjes zijn verboden terrein.
a. Voer een Breadth First Search uit op dit bord, uitgaande van vakje S. Schrijf in elk (niet-gearceerd) vakje de resulterende afstand vanaf S.
b. Bepaal, uitgaande van je antwoord bij **a.**, alle kortste paden van S naar D. Hoeveel verschillende kortste paden van S naar D zijn er?

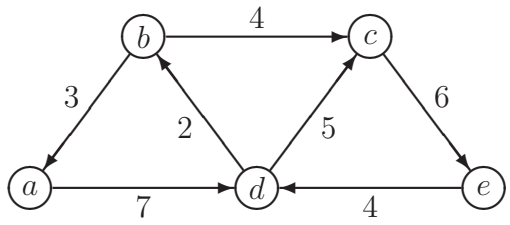
Opgave 5. (Opgave 9.3.1.a,b,c uit het boek van Levitin.)
Leg uit op welke manier (*if any*) het algoritme van Dijkstra of de onderliggende graaf moet worden aangepast om de volgende problemen op te lossen.
a. Bepaal kortste paden vanuit een enkele beginknoop in een *gerichte* gewogen graaf.
b. Bepaal een kortste pad *tussen twee gegeven knopen* in een gerichte of ongerichte gewogen graaf. (Deze variant wordt het *single-pair shortest-path problem* genoemd.)
c. Bepaal kortste paden *naar* een gegeven knoop vanuit elke andere knoop in een gerichte of ongerichte gewogen graaf. (Deze variant wordt het *single-destination shortest-paths problem* genoemd.)

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.

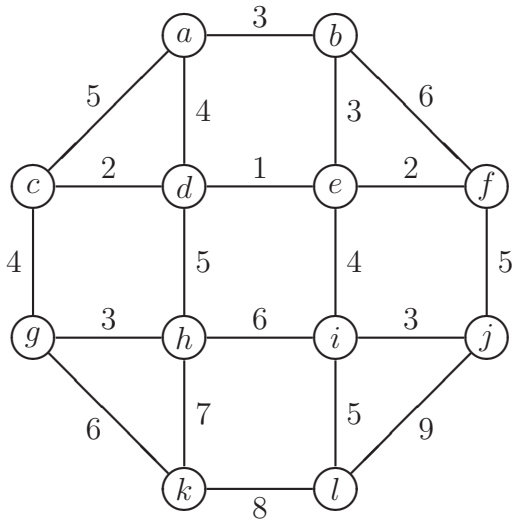
Opgave 6. (Opgave 9.3.2 uit het boek van Levitin.)
Los de volgende twee instanties van het kortste paden probleem vanuit knoop a op.
Voer het algoritme van Dijkstra uit zoals bij het voorbeeld op de sheets, door in de graaf steeds de kandidaatknopen te labelen met hun kandidaatafstand, en labels aan 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.
Geef de voortgang van het algoritme weer met een tabel van de volgende vorm (voor **a.**):

a	b	c	d	e	Actie
0	∞	∞	∞	∞	...
		...			...

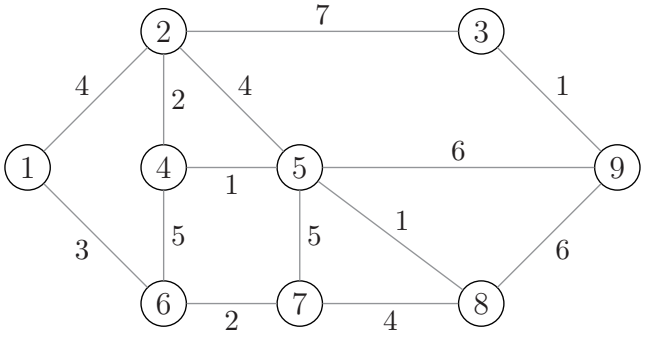
Zie ook de sheets voor een voorbeeld van het gebruik van deze tabel.
a.



b.



Opgave 7. Voer het algoritme van Dijkstra uit op de voorbeeldgraaf hieronder. Bepaal de (lengtes van de) kortste paden vanuit knoop 1.



Opgave 8. 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] := ∞;
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