Netwerkoptimalisatie

Kleinherenbrink, Y.C.

Yvette Kleinherenbrink

Netwerkoptimalisatie

Master thesis, defended on 11 September 2009

Thesis advisor: L.Kallenberg & P. Kop

Mathematisch Instituut, Universiteit Leiden

Voorwoord

Dagelijks reizen er zo'n 1,1 miljoen mensen met de trein en deze mensen zijn verdeeld over zo'n 4.700 treinritten. Het reizen met de trein wordt mogelijk gemaakt door de Nederlandse Spoorwegen. Elke reiziger van de NS wil zo snel mogelijk op het eindstation aankomen en het liefst met zo min mogelijk overstappen. De NS zoekt dit voor je uit. Op de site van de NS kun je invullen waar je naar toe wilt en waar je wilt vertrekken. Met één druk op de knop vertelt de computer je hoe laat je op je beginstation moet opstappen, waar en hoe laat je moet overstappen, hoe laat je aankomt op je eindstation en hoe lang je daar in totaal over doet. Ideaal dat de computer dat zo voor je kan berekenen. Maar daar zit uiteraard een hoop techniek en computerwerk achter. Wat de computer eigenlijk voor je doet is een optimale route voor je vinden. De computer maakt hierbij gebruik van een graaf. In dit boekje staan een aantal problemen beschreven waarvoor we op zoek gaan naar een optimale oplossing. Alle problemen die worden beschreven maken gebruik van een graaf.
Het doel van het boekje is om inzicht te geven in het maken van een graaf bij een probleem. Tevens geeft dit boekje aan hoe je optimale oplossingen kunt vinden van een probleem door gebruik te maken van de opgestelde graaf.
Het boekje is als volgt opgebouwd. In het eerste hoofdstuk maken we kennis met het begrip graaf en het begrip netwerk. In hoofdstuk 2 wordt het probleem 'Netwerkstromen' uitgelegd en opgelost. In hoofdstuk 3 wordt dit gedaan met het 'Koppelingsprobleem' en hoofdstuk 4 licht het 'Knapzakprobleem' verder toe. Wat de problemen precies inhouden vind je in de desbetreffende hoofdstukken. Hoofdstuk 5 bevat een aantal afsluitende opgaven. Deze kun je maken met behulp van de theorie die eerder in het boekje beschreven is. Het laatste hoofdstuk bevat een aantal korte uitwerkingen en antwoorden van een aantal opgaven die in de hoofdstukken staan. Er staat niet van alle opgaven een antwoord bij.
Het is aan te raden om de opgaven die je tegenkomt eerst te maken voordat je verder gaat. De opgaven zorgen voor een verduidelijking van de tekst en zijn een oefening voor het begrijpen en het zelf toepassen van de theorie. Alleen de afsluitende opgaven zijn los van elkaar te maken.

Inhoudsopgave

Hoofdstuk 1

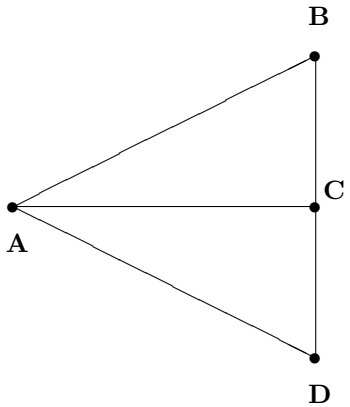
Grafen

1.1 Inleiding

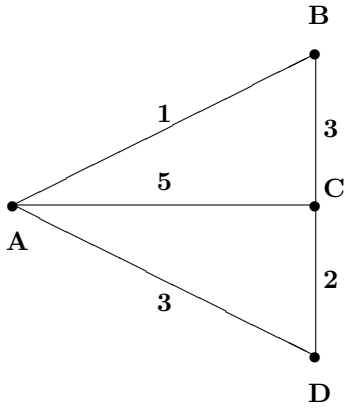
In dit hoofdstuk zullen we laten zien wat grafen precies zijn en wat we verstaan onder een netwerk. Verder zullen we wat vertellen over de geschiedenis van de grafentheorie en bij welke problemen de grafen onder andere gebruikt worden. Ook komen verschillende soorten grafen aan bod, waarvan een aantal in de verdere hoofdstukken terug zullen komen.

1.2 Wat zijn grafen?

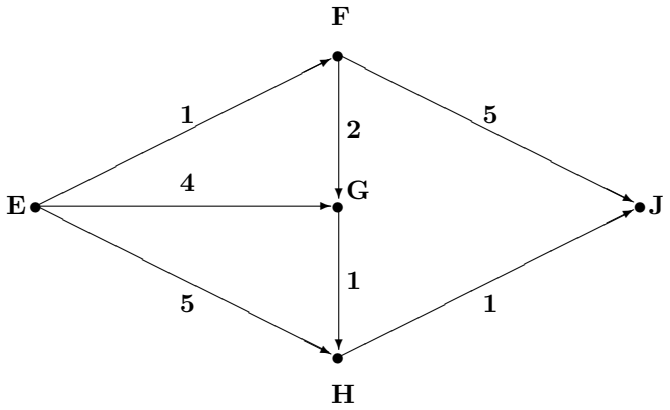
Om problemen op te lossen maken we vaak gebruik van een *graaf*. De NS-kaart is een voorbeeld waarin gebruik wordt gemaakt van grafen. Met behulp van de NS-kaart kun je bepalen wat de kortste route is of wat de snelste route is. Om een route te vinden staan alle treinstations van Nederland op het kaartje met daarbij alle verbindingen die er zijn tussen de verschillende steden. De NS-kaart is eigenlijk niets anders dan een schematisch kaartje van Nederland. Zo'n schematisch kaartje wordt ook wel een *graaf* genoemd. Een graaf bestaat uit punten die onderling verbonden zijn door lijnen. De punten in de graaf noemen we de *knooppunten* en de lijnen die de knooppunten verbinden noemen we de *takken* van de graaf. In het onderstaand kaartje staan vier plaatsen met daarbij de verbindingen tussen de plaatsen. Zoals je zult begrijpen is de graaf die de NS gebruikt een stuk groter dan het onderstaande kaartje. Maar door het onderstaande kaartje zul je een idee krijgen van hoe de grote gedetailleerde kaart in elkaar zit.



Willen we nu de snelste route weten om van plaats A naar C te komen dan zullen we eerst moeten weten hoe lang we over de verschillende verbindingen doen. Het kan namelijk zijn dat het sneller is om via B te reizen in plaats van direct naar C te gaan. Om ervoor te zorgen dat we de de snelste route kunnen bepalen, moeten we de tijd die we er over doen bij de verbindingen zetten. We zeggen ook wel dat we elke tak van de graaf een *waarde* geven. We krijgen dan bijvoorbeeld de volgende graaf, waarbij we er dus 1 minuut over doen om tussen A en B te reizen:



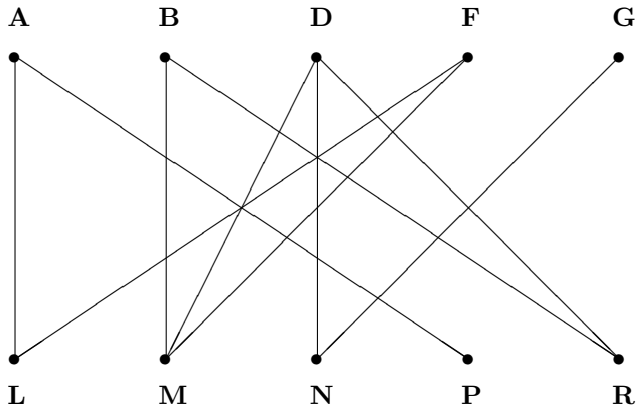
We zien in het bovenstaande kaartje dat we vanuit A beter naar C kunnen rijden via stad B in plaats van de directe verbinding van A naar C te gebruiken. In deze graaf zegt de lengte van de takken in het plaatje dus niets over de tijd die we er over doen om van het ene naar het andere knooppunt te komen. Zo zien we dat de tak van A naar B langer is dan de tak van A naar C, maar de eerste tak heeft een waarde 1 en de tweede een waarde 5. Als iemand anders ook de graaf zou moeten tekenen die hoort bij de vier plaatsen, dan kan het zijn dat deze persoon de graaf anders tekent dan de bovenstaande graaf. Toch geeft de graaf hetzelfde weer. We zeggen dan dus ook dat het dan gaat om twee dezelfde grafen, ook al zien ze er anders uit. Twee grafen met hetzelfde aantal knooppunten en verbindingen tussen dezelfde tweetal knooppunten noemen we dus gelijk.
Een ander voorbeeld waar we handig gebruik kunnen maken van een graaf is een stadsplattegrond. We hebben in een stad te maken met kruispunten en we willen van het ene naar het andere kruispunt. Ook hebben we in steden nog te maken met éénrichtingsverkeer. Ook hier moeten we rekening mee houden bij het plannen van een route. We moeten dus aangegeven in welke richting we van het ene kruispunt naar de andere mogen rijden. In de graaf die hoort bij een stadsplattegrond komen er daarom geen lijnen meer tussen de verbindingen te staan, maar gebruiken we pijlen. Een pijl geeft dan de richting aan waarin we mogen rijden. Omdat we hier gebruikmaken van pijlen in plaats van lijnen, hebben we te maken met een *gerichte graaf*. Staan er ook nog getallen bij de pijlen, die dus bijvoorbeeld aangeven hoe lang we over dat stukje doen, dan praten we over een *netwerk*. Een netwerk is een gerichte graaf waarbij de pijlen een bepaalde waarde hebben gekregen. Het onderstaande kaartje geeft een netwerk weer.



In het bovenstaande netwerk kunnen we op zoek gaan naar de snelste route om van punt E naar punt J te komen. Zoals je kunt zien is de route van E via F, G en H naar J de snelste met waarde 5. In de graaf noemen we dit ook wel het *pad* $EFGHJ$. Een pad is dus een opeenvolging van punten en pijlen tussen de opeenvolgende punten. Op een pad mag elk knooppunt maar één keer voorkomen.

We hebben gezien dat we een graaf kunnen maken als we te maken met een aantal steden met verbindingen daartussen en bij het maken van een stadsplattegrond. Deze twee worden gecombineerd bij het programmeren van een TomTom. We kunnen een TomTom laten zoeken naar de snelste route, maar ook naar de kortste route. Of een route die de snelwegen vermijdt. Om een route te vinden, zullen onder andere alle plaatsen van Nederland in de TomTom moeten staan, met daarbij de verbindingen tussen deze steden. Maar ook de plattegronden van de steden zullen erin gezet moeten worden. Eigenlijk zal er dus een zeer gedetailleerde kaart van Nederland in moeten staan. In een TomTom wordt gebruik gemaakt van een graaf om deze gedetailleerde kaart weer te geven.

We zullen in dit boekje ook grafen tegenkomen met speciale eigenschappen.
Een voorbeeld van zo'n bijzondere graaf is de volgende:

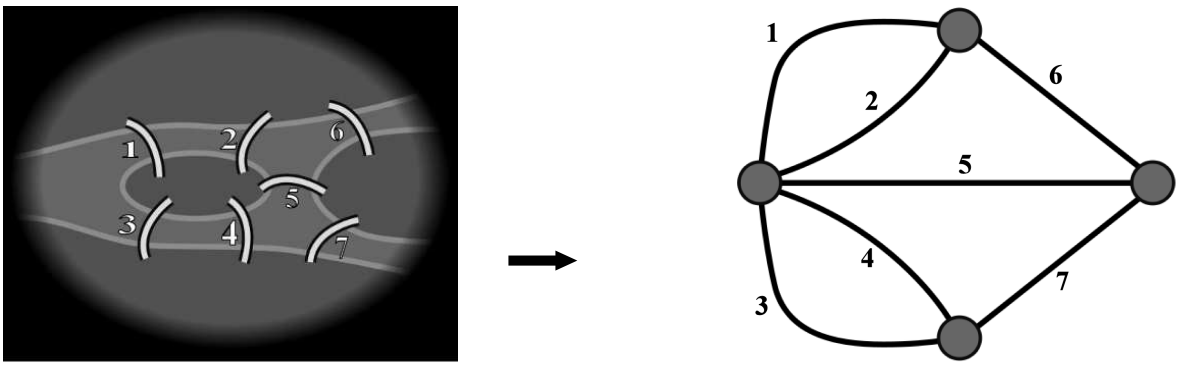


Deze graaf heeft de eigenschap dat de knooppunten verdeeld zijn in twee groepen. Eén groep knooppunten aan de bovenkant en een groep aan de onderkant. Er lopen alleen takken van bovenste verzameling knooppunten naar de onderste verzameling knooppunten en er lopen geen takken tussen de knooppunten uit de groepen onderling. Deze bijzondere graaf noemen we een *bipartiete graaf*.

Zoals je zult begrijpen wordt natuurlijk niet alleen bij de NS en de TomTom gebruik gemaakt van een graaf. Er zijn veel problemen die we kunnen oplossen door daarbij gebruik te maken van een graaf. Een aantal van die problemen gaan we zien in de volgende hoofdstukken. Het eerste wat we zullen doen als we een probleem hebben is een graaf maken die hoort bij het probleem. De oplossing van het probleem kunnen we dan vinden met behulp van de graaf. Vaak maken we daarbij gebruik van een algoritme. Een algoritme is eigenlijk niets meer dan een stappenplan om uiteindelijk bij de oplossing uit te komen. Het bewijs dat een algoritme echt klopt en er dus uitkomt wat we willen laten we hier achterwege. De bewijzen zijn wel terug te vinden in de literatuur.

1.3 Geschiedenis van de grafentheorie

Het eerste artikel over een onderwerp dat betrekking heeft op grafen werd in 1736 geschreven door de Zwitserse wiskundige *Leonhard Euler*. Hij schrijft hierin over de 'Zeven bruggen van Königsberg'. Het probleem gaat over de stad Königsberg (heden ten dage Kaliningrad). Deze stad ligt aan de rivier de Pregel, waarin twee eilanden liggen die door zeven bruggen met elkaar en met de vaste wal verbonden waren. Een schematische afbeelding van de stad met de rivier en de bruggen staat hieronder. De vraag was nu of het mogelijk is om zó te wandelen dat je precies één maal over elke brug liep. Euler heeft van dit probleem een graaf gemaakt. De knooppunten van de graaf stelden de oevers en de eilanden voor en de takken in de graaf representeerde de bruggen (zie ook het plaatje hieronder). Euler heeft bewezen dat het niet mogelijk om een pad in de graaf te vinden, waarbij elke tak precies één keer bewandeld werd.



Tot op de dag van vandaag wordt voor vele problemen in de wiskunde gebruik gemaakt van grafen. Eén van de meest beroemde problemen waarbij er gebruik wordt gemaakt van een graaf is het 'vierkleuren'-probleem. In dit probleem is het de bedoeling om de knooppunten van een zogenaamde vlakke graaf te

kleuren, waarbij bij het kleuren van de knooppunten twee knooppunten die met elkaar verbonden zijn door een tak niet dezelfde kleur mogen krijgen. De vraag is of zo'n kleuring van de knooppunten mogelijk is met hooguit vier kleuren. Het probleem van graafkleuringen wordt verder toegelicht in het Zebraboekje 'Grafen in de praktijk' van Hajo Broersma.

Hoofdstuk 2

Netwerkstromen

2.1 Introductie

In dit hoofdstuk gaan we kijken naar het zogenaamde maximale-stroom-probleem. Voor dit probleem willen we zoveel mogelijk van een bepaald product versturen van een begin- naar een eindpunt. Maar dit kunnen we niet zomaar doen, omdat we te maken hebben met een aantal beperkingen. Hieronder zie je drie voorbeelden van een maximaal-stroom-probleem waarvoor we graag een oplossing zouden willen vinden.

2.1.1 Pakketservice

Een pakketservicebedrijf wil elke dag zo veel mogelijk pakketjes transporteren van Nijmegen naar Amsterdam. Elk transport moet alleen wel eerst langs een aantal andere steden voordat dit bij Amsterdam aankomt. Die andere steden kunnen Utrecht, Den Haag, Haarlem of Rotterdam zijn. Maar tussen twee van deze steden geldt dat er per dag maar een beperkt aantal vrachtwagens beschikbaar zijn die tussen die twee steden kunnen rijden. Elk stukje transportroute (stukje route tussen twee steden) heeft dus een bepaalde *capaciteit*. Ook is er niet tussen elk tweetal steden een route. De capaciteiten van de mogelijke transportroutes staan in de onderstaande tabel:

		NAAR					
		Nijmegen	Rotterdam	Utrecht	Den Haag	Haarlem	Amsterdam
VAN	Nijmegen	-	9	-	11	-	-
	Rotterdam	-	-	6	-	3	-
	Utrecht	-	-	-	-	4	5
	Den Haag	-	1	-	-	3	-
	Haarlem	-	-	-	-	-	10
	Amsterdam	-	-	-	-	-	-

Alle pakketjes die binnenkomen in een stad, gaan daar ook weer weg, behalve in de eindbestemming Amsterdam. We willen zoveel mogelijk pakketjes versturen van Nijmegen naar Amsterdam, waarbij de capaciteiten van de transportroutes natuurlijk niet overschreden mogen worden.

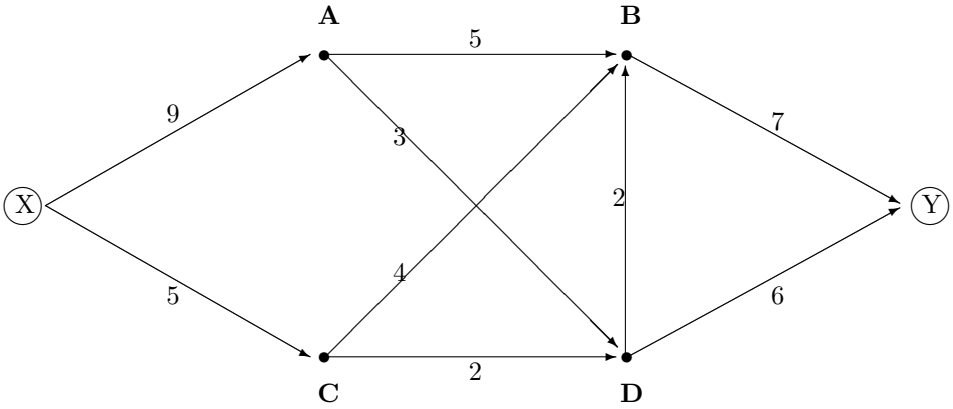
OPGAVEN

1 Hoeveel transporten met pakketjes kun je nu per dag van Nijmegen naar Amsterdam krijgen?

2.1.2 E-mail

Elke dag sturen de gebruikers van het internet elkaar miljarden boodschappen per e-mail. E-mail is één van de toepassingen van het internet waar vrijwel iedereen gebruik van maakt. Maar een e-mail die je

stuurt komt niet in één keer bij de ontvanger terecht. Daarvoor moet het bericht eerst een aantal tussenstations passeren en per e-mailbericht kunnen dit verschillende tussenstations zijn. Dit hangt allemaal af van je eigen mailserver en de mailserver van de ontvanger. Het bericht moet dus door een heel netwerk heen voordat het bij de ontvanger terecht kan komen. Maar elk stukje van het netwerk heeft zo zijn beperkingen, doordat er maar een beperkt aantal mailtjes per minuut 'door' dat stukje netwerk kunnen. Nemen we nu twee bedrijven, bedrijf X en bedrijf Y, die elkaar per minuut zoveel mogelijk mailtjes willen versturen. Het netwerk ziet er, sterk vereenvoudigd, als volgt uit. Hierbij geven de getallen bij de pijlen de capaciteiten weer, dat wil zeggen het maximum aantal mailtjes dat per minuut verstuurd kan worden.

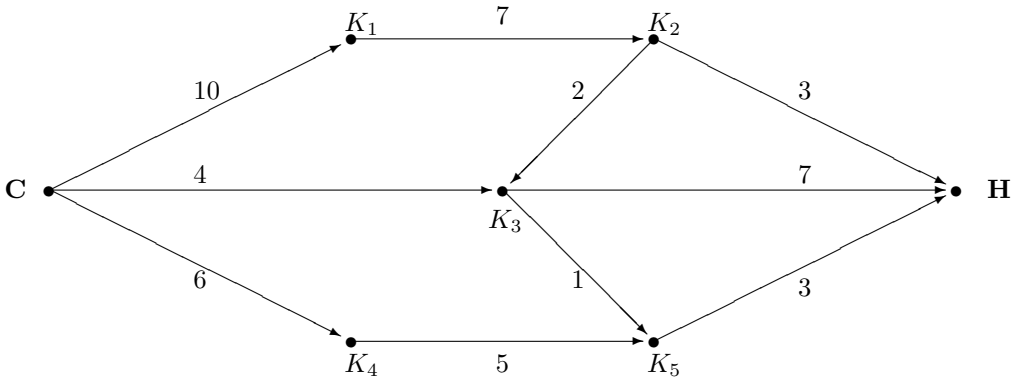


OPGAVEN

2 Heb je een idee hoeveel mailtjes er nu per minuut verstuurd kunnen worden van bedrijf X naar bedrijf Y?

2.1.3 Gasleidingen

Je hebt een nieuw huis gekocht en natuurlijk heb je ook gas nodig. Maar het gas dat je gebruikt moet bij de centrale vandaan komen. Daarbij stroomt het via buizen naar jouw huis toe. Het is uiteraard niet één rechte buis, maar het gas moet via een heel netwerk van buizen uiteindelijk bij het huis aankomen. Stel het gas moet vanuit de centrale C door het onderstaande, sterk vereenvoudigde, netwerk van buizen om uiteindelijk bij het huis H aan te komen. De pijlen geven aan in welke richting het gas door die buis kan stromen. In de tussenpunten, de punten K_1 tot en met K_5, splitsen de buizen zich. Maar niet elke buis is even dik en dus kan er per minuut niet door elke buis evenvaal gas stromen. Bij de pijlen staat dan ook hoeveel gas er per minuut doorheen kan.



OPGAVEN

3 Heb je een idee hoeveel gas er nu per minuut van de centrale naar het huis kan stromen?

In de bovenstaande drie problemen willen we zoveel mogelijk 'stroom' van het begin- naar het eindpunt sturen. In het eerste voorbeeld zijn de pakketjes de 'stroom'. De e-mails zijn dit bij het tweede probleem en bij het derde probleem gaat het om het gas. Daarbij hebben we te maken met capaciteiten

van de verbindingen. Alle drie problemen hebben dus hetzelfde karakter. We zullen één van de problemen verder uitwerken.

2.2 Maximale stroom

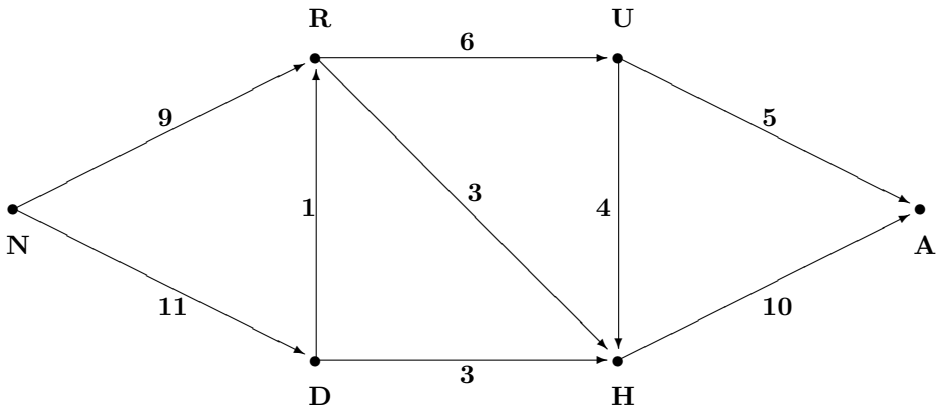
We pakken het eerste voorbeeld uit de vorige paragraaf er nog eens bij, het probleem 'Pakketservice'. Daarbij hadden we de volgende tabel met daarin de capaciteiten van de transportroutes:

		NAAR					
		Nijmegen	Rotterdam	Utrecht	Den Haag	Haarlem	Amsterdam
VAN	Nijmegen	-	9	-	11	-	-
	Rotterdam	-	-	6	-	3	-
	Utrecht	-	-	-	-	4	5
	Den Haag	-	1	-	-	3	-
	Haarlem	-	-	-	-	-	10
	Amsterdam	-	-	-	-	-	-

Nu willen we per dag zoveel mogelijk transporteren van Nijmegen naar Amsterdam. Maar we zullen rekening moeten houden met de capaciteiten van de transportroutes. In het probleem bekijken we hoeveel transporten er over elke route zullen 'stromen', zonder dat we de capaciteiten van de route overschrijden. We willen dit zó doen dat er een maximaal aantal pakketten in Amsterdam aankomt. We zeggen ook wel dat we voor het probleem zoeken naar een *maximale stroom* van Nijmegen naar Amsterdam.

Ook over de andere problemen uit de vorige paragraaf zeggen we dat we op zoek gaan naar een *maximale stroom* van het begin- naar het eindknooppunt. Voor elk knooppunt, behalve voor het begin- en eindknooppunt, moet gelden dat alle stroom die er binnenkomt er ook weer weggaat. Het uiteindelijke doel is om zoveel mogelijk 'stroom' van het begin- naar het eindknooppunt te sturen, zó dat de capaciteiten van de takken niet wordt overschreden.

Allereerst gaan we een graaf maken die bij het probleem hoort. We nemen voor elke stad die we hebben een knooppunt in de graaf. Bij het knooppunt zetten we de beginletter van de stad, zodat het duidelijk is welk knooppunt bij welke stad hoort. Het knooppunt behorende bij de stad waar de pakketjes vandaan komen, in ons geval dus Nijmegen, zetten we helemaal aan de linker kant. Het knooppunt behorende bij de stad waar de pakketjes uiteindelijk heen moeten, in ons geval dus Amsterdam, zetten we helemaal aan de rechter kant. De knooppunten van de tussenliggende steden komen daar tussen te liggen. Tussen de knooppunten komen pijlen te lopen die de transportroutes aangeven. Bij de pijlen zetten we de capaciteiten van de transportroutes. Uiteindelijk komt de graaf er als volgt uit te zien:



We hebben nu de graaf gemaakt die hoort bij het probleem. Ook hebben alle pijlen in de graaf een capaciteit gekregen. We kunnen nu een maximale stroom gaan bepalen van Nijmegen naar Amsterdam en daarbij maken we gebruik van het algoritme van Ford en Fulkerson.

2.3 Algoritme van Ford en Fulkerson

Om een maximale stroom te vinden gaan we het algoritme van Ford en Fulkerson gebruiken. Het algoritme is vernoemd naar de bedenkers van het algoritme, L.R. Ford en D.R. Fulkerson.
Het idee van het algoritme is als volgt. Allereerst gaan we in de graaf op zoek naar een pad van het beginknooppunt naar het eindknooppunt. In ons voorbeeld is Nijmegen het beginknooppunt en Amsterdam het eindknooppunt. Als we een pad hebben gevonden, gaan we zoveel mogelijk stroom over dat pad sturen, waarbij we rekening moeten houden met de capaciteiten van de routes. Is de waarde van een pijl bijvoorbeeld 3, dan kunnen we over die pijl hooguit 3 transporten vervoeren, oftewel de stroom is daar maximaal 3. Nadat we bepaald hebben hoeveel stroom we maximaal over dat pad kunnen sturen gaan we de graaf die we hadden een beetje aanpassen. Voor de pijlen die in het pad zitten gaan we ook een pijl in tegengestelde richting tekenen en die geven we dezelfde capaciteit als de stroom die we er net overheen gestuurd hebben. Van de capaciteit van de oorspronkelijke pijl halen we de hoeveelheid stroom die we er overheen gestuurd hebben vanaf. Zo hebben we weer een nieuwe graaf en beginnen we weer met het zoeken van een pad van het beginknooppunt naar het eindknooppunt. We gaan net zo lang door totdat we geen pad meer kunnen vinden. We geven met Δ de totale waarde van de stroom aan.

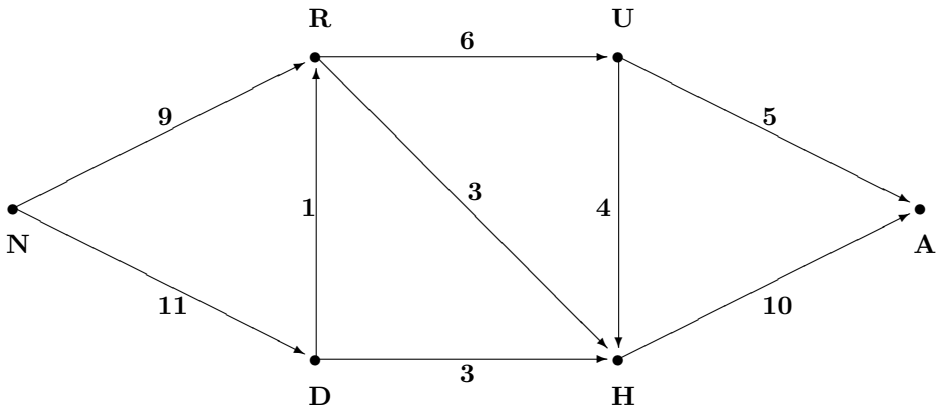
We zullen nu eerst het algoritme geven en daarna zullen we het algoritme toepassen op het voorbeeld.

0. Begin met $\Delta = 0$.
1. Zoek een pad van het beginknooppunt naar het eindknooppunt. Bestaat dit pad niet dan hebben we de maximale stroom gevonden.
2. Bepaal het minimum van de capaciteiten van de pijlen van het pad. Dit is de grootste hoeveelheid stroom die we over het pad kunnen sturen. Noem dit getal Δw.
3. We maken een nieuwe graaf:
   - Neem dezelfde knooppunten als de oorspronkelijke graaf.
   - De pijlen die niet in het pad zitten blijven onveranderd.
   - Voor de pijlen die wél in het pad zitten tekenen we pijlen in beide richtingen.
   - Geef de pijl in tegengestelde richting een capaciteit Δw. Als de pijl in tegengestelde richting al bestaat, tel dan Δw op bij de capaciteit van de pijl.
   - Van de pijl in de 'goede' richting (oftewel de richting waarin je over het pad loopt) trek je Δw van de capaciteit van de pijl af. Als deze capaciteit 0 wordt, laat dan de pijl weg.
4. Tel Δw op bij Δ.
5. Ga terug naar stap 1.

Het bewijs dat we met bovenstaand algoritme ook echt een maximale stroom vinden laten we hier achterwege[1].

We zullen het algoritme nu toepassen op het voorbeeld. Daarbij hebben we de volgende graaf gemaakt:

[1] zie Thijms: Operationele Analyse §3.2

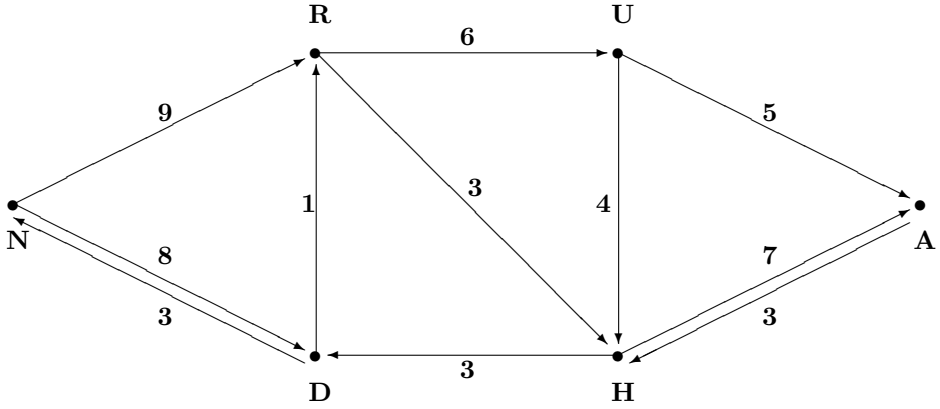


Eerste ronde
Stap 0. $\Delta = 0$

Stap 1. We nemen het volgende pad : N - D - H - A.

Stap 2. De pijlen van het pad hebben capaciteit 11, 3 en 10. Het minimum van deze capaciteiten is dus 3. We kiezen dus $\Delta w = 3$.

Stap 3. We maken een nieuwe graaf en die komt er als volgt uit te zien:



Zoals je in de bovenstaande graaf kunt zien is er geen pijl meer van Den Haag naar Haarlem. Volgens het algoritme hebben we namelijk Δw af moeten halen van de waarde van die pijl. Omdat de waarde 3 was en we er ook 3 vanaf moeten halen, houden we dus een waarde 0 over.

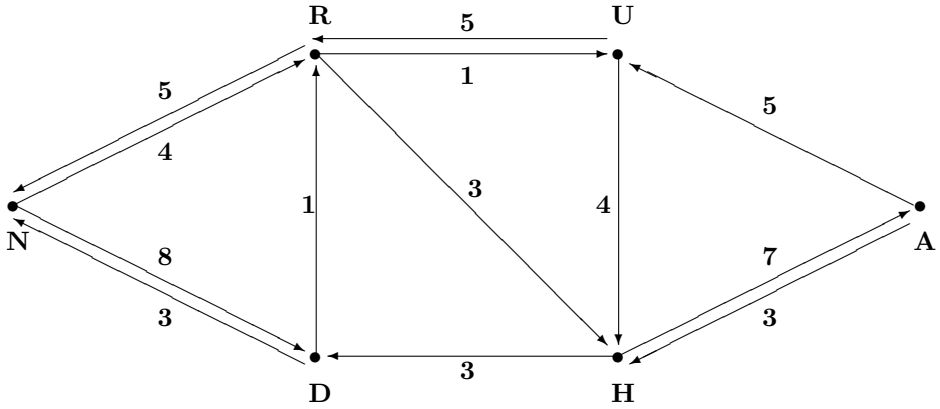
Stap 4. $\Delta = 0 + \Delta w = 0 + 3 = 3$.

Stap 5. We gaan terug naar stap 1.

Tweede ronde
Stap 1. We nemen het volgende pad: N - R - U - A.

Stap 2. De pijlen van het pad hebben capaciteit 9, 6 en 5. Het minimum van deze capaciteiten is dus 5. We kiezen dus $\Delta w = 5$.

Stap 3. We maken een nieuwe graaf en die komt er als volgt uit te zien:



Hetzelfde als in de vorige ronde zien we ook hier weer gebeuren. De pijl van Utrecht naar Amsterdam krijgt capaciteit 0 en daarom laten we hem weg.

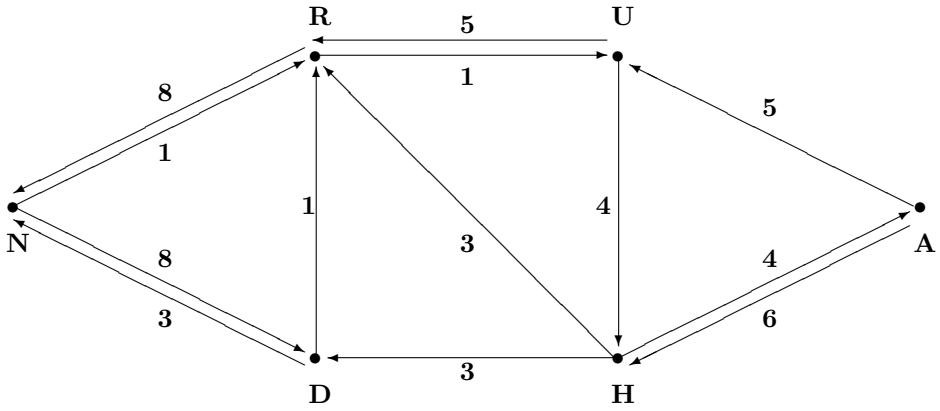
Stap 4. $\Delta = 3 + \Delta w = 3 + 5 = 8$.

Stap 5. We gaan terug naar stap 1.

Derde ronde
Stap 1. We nemen het volgende pad: N - R - H - A.

Stap 2. De pijlen van het pad hebben capaciteit 4, 3 en 7. Het minimum van deze capaciteiten is dus 3. We kiezen dus $\Delta w = 3$.

Stap 3. We maken een nieuwe graaf en die komt er als volgt uit te zien:



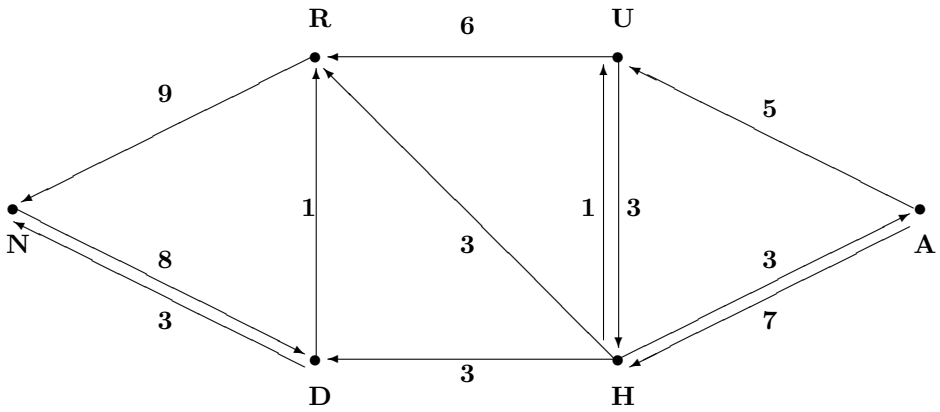
Stap 4. $\Delta = 8 + \Delta w = 8 + 3 = 11$.

Stap 5. We gaan terug naar stap 1.

Vierde ronde
Stap 1. We nemen het volgende pad: N - R - U - H - A.

Stap 2. De pijlen van het pad hebben capaciteit 1, 1, 4 en 4. Het minimum van deze capaciteiten is dus 1. We kiezen dus $\Delta w = 1$.

Stap 3. We maken een nieuwe graaf en die komt er als volgt uit te zien:



Stap 4. $\Delta = 11 + \Delta w = 11 + 1 = 12$.

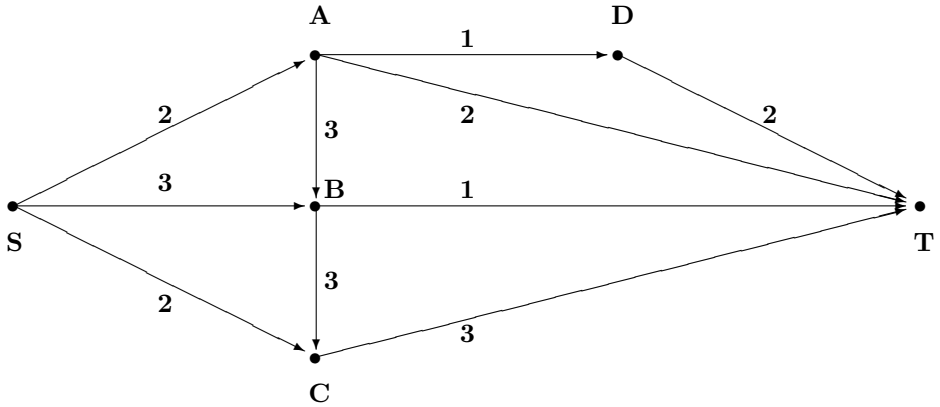
Stap 5. We gaan terug naar stap 1.

Vijfde ronde
Stap 1. We zien in de bovenstaande graaf dat we vanuit N alleen nog maar in D kunnen komen. Vanuit D kunnen we alleen naar R, maar vanuit R kunnen we nergens meer naar toe (we kunnen wel weer terug naar N, maar dat heeft niet zoveel zin). We kunnen vanuit N dus niet meer in A komen en daarom hebben we nu een maximale stroom gevonden die een waarde heeft van 12. Voor die stroom sturen we dus het volgende aantal transporten over de genoemde routes:

Nijmegen	-	Rotterdam:	9	Den Haag	-	Haarlem:	3
Nijmegen	-	Den Haag:	3	Haarlem	-	Amsterdam:	7
Den Haag	-	Rotterdam:	0	Utrecht	-	Haarlem:	1
Rotterdam	-	Utrecht:	6	Utrecht	-	Amsterdam:	5
Rotterdam	-	Haarlem:	3				

OPGAVEN

4 Bepaal met behulp van het algoritme van Ford en Fulkerson een maximale stroom van **S** naar **T** in het onderstaande netwerk.

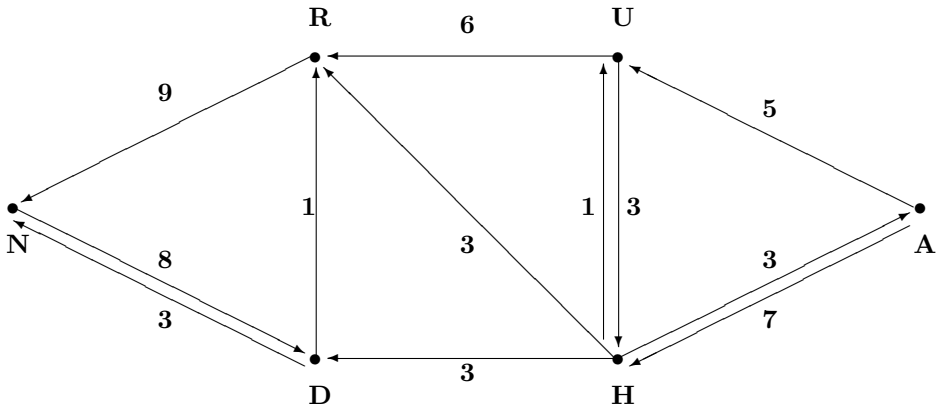


5 Los het probleem 'E-mails' uit paragraaf 2.1.2 op.

2.4 Minimale snede

In de vorige paragraaf hebben we gezien hoe we een maximale stroom in een netwerk kunnen bepalen. Stel nu dat we het probleem omdraaien. We willen voorkomen dat er iets van Nijmegen naar Amsterdam gestuurd wordt. Om dat te kunnen voorkomen zullen we een aantal pijlen in de graaf moeten verwijderen. Maar voor het weghalen van een pijl met capaciteit 1 moeten we € 100,- betalen. Voor het weghalen van een pijl met capaciteit 2 betalen we dus € 200,-. Uiteraard willen we de kosten zo laag mogelijk houden. Welke pijlen haal je weg zó dat je zo min mogelijk betaalt?

We pakken het voorbeeld er nog eens bij. We kunnen bijvoorbeeld de volgende pijlen weghalen: $N-R$, $D-R$ en $D-H$. We kunnen dan niet meer van N naar A komen. De genoemde pijlen hebben een totale capaciteit van 13 en dat zou betekenen dat we € 1300,- moeten betalen om deze pijlen weg te mogen halen. We zien dat we door de pijlen weg te halen twee verzamelingen van knooppunten krijgen. In de eerste verzameling X zitten alleen knooppunt N en D en in de andere verzameling Y zit de rest van de knooppunten. Verzameling X noemen we ook wel een *snede*. De *capaciteit* van de snede X is 13. Dit krijgen we door de capaciteiten van alle pijlen die van X naar Y lopen bij elkaar op te tellen. En dit doen we in de originele graaf. Uiteindelijk zijn we dus op zoek naar de snede waarvan de capaciteit minimaal is. Kort gezegd zijn we op zoek naar de *minimale snede*. We kunnen deze minimale snede bepalen met behulp van de laatste graaf die we gekregen hebben na het toepassen van het algoritme van Ford en Fulkerson. Hoe dit werkt bekijken we aan de hand van het voorbeeld. De laatste graaf die we daar kregen was de volgende:



We zien dat we in deze graaf vanuit knooppunt N alleen nog maar in D en R kunnen komen. Hier zien we dus de splitsing van de knooppunten en precies deze splising geeft de minimale snede. Verzameling X bestaat nu uit de knooppunten N, D en R en verzameling Y bestaat uit de knooppunten U, H en A. Nu zijn er drie pijlen die van X naar Y lopen, namelijk de pijl van R naar U, de pijl van R naar H en de pijl van D naar H. Als we de capaciteiten van deze pijlen bij elkaar op gaan tellen komen we uit op 12. De capaciteit van de verkregen snede is dus 12. We zien dat de capaciteit van de verkregen snede gelijk is aan de waarde van de maximale stroom. Ford en Fulkerson hebben laten zien dat dit altijd het geval is. De capaciteit van de minimale snede is dus gelijk aan de waarde van de maximale stroom. Het bewijs dat dit waar is laten we hier achterwege, maar is wel terug te vinden in de literatuur[2].
Nu we dit weten kunnen we dus altijd controleren of we ook echt de maximale stroom hebben gevonden.

OPGAVEN

6 Bepaal de minimale snede van de graaf uit opgave 4. Komt dit antwoord overeen met de waarde van de maximale stroom die je hebt gevonden?

[2]zie Thijms: Operationele Analyse

Hoofdstuk 3

Koppelingen

3.1 Introductie

Het probleem dat we in dit hoofdstuk gaan bekijken heeft alles te maken met het vormen van koppels. En zoals je zult zien kunnen dit verschillende soorten koppels zijn. Maar die koppels mag je niet zomaar maken. Daar zijn een aantal regels en eisen aan verbonden. Voor elk probleem kunnen die regels en eisen anders zijn, maar het algemene idee erachter is steeds hetzelfde. Hieronder staan drie van zulke problemen waar we graag een optimale oplossing voor zouden willen vinden.

3.1.1 Klassenplattegrond

Je bent docent wiskunde op een middelbare school. Aan het begin van het schooljaar wil je een plattegrond van de klas maken, om zo de namen sneller uit je hoofd te kennen. In de nieuwe klas zitten precies evenveel meisjes als jongens, zodat je elk meisje naast een jongen kunt zetten. Om er voor te zorgen dat de klas niet te veel commentaar gaat geven laat je de leerlingen zelf voor een deel een keuze maken. Aan de jongens uit de klas geef je een lijst met daarop de namen van de meisjes uit de klas. Op die lijst zetten ze hun eigen naam en kruisen ze aan welk meisje ze aardig vinden en waar ze dus wel naast zouden willen zitten. Ze moeten natuurlijk minstens één persoon aankruisen. Uiteraard gaat dit wel in overleg met de meisjes in de klas. Als een jongen dus de naam van een bepaald meisje aankruist, dan geldt dus ook dat dat meisje ook wel naast hem zou willen zitten.
Neem bijvoorbeeld klas 5 atheneum. Deze klas bestaat uit 10 leerlingen; 5 jongens en 5 meisjes. Aan deze klas geef je de lijsten met daarop de namen van alle meisjes uit de klas. De jongens zetten hun naam erboven en vullen de lijsten, in overleg met de meisjes, in. De gegevens die op de lijsten staan zet je in een tabel:

Naam jongen	Naast wie wil hij zitten?
Bart	Mariska, Romy
Arnold	Pauline, Laura
Dirk	Natascha, Romy, Mariska
Frank	Laura, Mariska
Gerard	Natascha

OPGAVEN

1 Kun jij er nu, met behulp van deze lijsten, voor zorgen dat iedereen naast iemand komt te zitten die hij/zij aardig vindt?

3.1.2 Rooster

Je moet een mondelinge toets afleggen voor Nederlands. Deze toets duurt 30 minuten. Je docent geeft je een aantal tijdstippen waarop je de toets kan doen. Je mag een aantal tijdstippen uitkiezen waarop je zou kunnen. Maar je bent niet de enige uit de klas die een mondeling moet doen. Ook de rest van

de leerlingen geeft aan de docent door wanneer hij/zij zou kunnen. De docent krijgt de volgende gegevens:

Persoon	Tijdstip waar hij/zij kan
1	13.00, 13.30, 14.00
2	14.00, 15.00
3	15.00, 15.30
4	15.30, 16.00
5	14.00, 14.30
6	13.00
7	15.00

OPGAVEN

2 Kun je de docent een voorstel doen voor het rooster van de mondelinge toetsen, zó dat iedereen ingepland wordt op een tijdstip dat hij/zij kan?

3.1.3 Bedrijf met opdrachten

Een bedrijf heeft 9 werknemers. Op een zekere dag wil het bedrijf zo veel mogelijk opdrachten uitvoeren. Voor elke opdracht geldt dat één werknemer een hele dag bezig is om de opdracht uit te voeren. Maar niet iedere opdracht kan door iedere werknemer worden verricht. In de onderstaande tabel staat welke opdracht door welke werknemer gedaan kunnen worden.

Opdracht	Werknemers die de opdracht kunnen uitvoeren
A	1, 2, 6, 8
B	5, 6
C	1, 2, 3, 4, 9
D	2, 5, 6
E	4, 7, 8, 9
F	2, 8
G	1, 6
H	2, 5, 6, 8
I	3, 7, 9

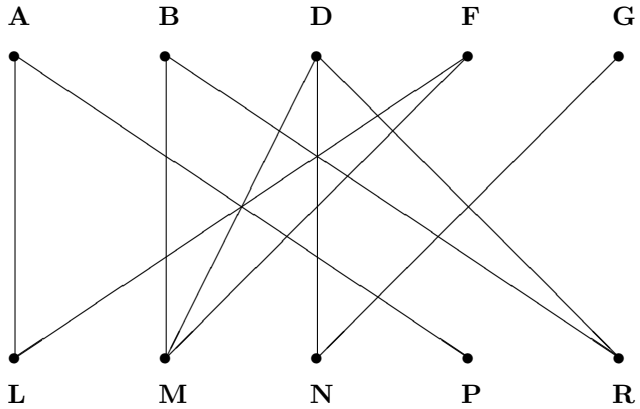
OPGAVEN

3 Het bedrijf wil weten of de opdrachten A, B, D, F, G en H die dag uitgevoerd kunnen worden en wie welke opdracht dan moet doen. Kun je het bedrijf vertellen of dit gaat lukken?

De bovenstaande problemen lijken allemaal erg veel op elkaar. In alle problemen willen we er voor zorgen dat we koppels krijgen. In het eerste voorbeeld is het de bedoeling om koppels te maken van een jongen en een meisje. Voor het tweede voorbeeld koppelen we een leerling aan een tijd. In het laatste voorbeeld willen we de werknemers koppelen aan een opdracht. Het eerste voorbeeld zullen we hier uitwerken. De rest van de voorbeelden kunnen op dezelfde manier opgelost worden.

3.2 Koppelingsprobleem

Het eerste probleem uit de vorige paragraaf wordt ook wel het huwelijksprobleem genoemd.
We gaan voor dit probleem eerst een graaf maken. Daarvoor maken we twee rijen; voor de jongens één en voor de meisjes één. In de bovenste rij zetten we de jongens en voor elke jongen uit de klas maken we een knooppunt. De onderste rij wordt een rij voor de meisjes met voor elk meisje uit de klas een knooppunt. Om het duidelijk te houden zetten we bij de knooppunten de eerste letter van de namen. Nu moeten er nog takken komen te lopen tussen de knooppunten om de graaf af te maken. De takken krijgen een betekenis en in ons geval tekenen we een tak tussen twee knooppunten als de twee personen wel naast elkaar willen zitten. We krijgen dus geen takken tussen twee jongens en ook geen tak tussen twee meisjes. Met deze betekenis van de takken, kunnen we nu de graaf gaan maken. Als we alle jongens één voor één af gaan, krijgen we uiteindelijk de volgende graaf:



In de graaf zien we dus een tak van bijvoorbeeld Arnold naar Pauline en Laura, omdat hij wel naast deze twee meisjes wil zitten, en deze meisjes wel naast hem.
Zoals je ziet bestaat de graaf eigenlijk uit twee verzamelingen van knooppunten; de knooppunten van de jongens en die van de meisjes. Laten we de verzameling knooppunten van de jongens X noemen en de verzameling knooppunten van de meisjes Y. Zoals je ziet lopen er alleen maar takken van X naar Y. We noemen dit een *bipartiete graaf*.

OPGAVEN

4 Stel er komt nog een jongen en meisje bij in de klas: Huub en Sanne. Ook Huub vult een lijst in en daar komt uit dat hij wel naast Pauline of Sanne zou willen zitten. Verder hebben Frank en Gerard aangegeven dat ze ook wel naast Sanne zouden willen zitten en ook de meisjes stemmen in met deze keuzes.
Teken de graaf die hoort bij dit probleem.

We gaan weer uit van het oorspronkelijke probleem, dus zonder dat Huub en Sanne erbij zitten. We willen nu dat elke jongen aan een meisje 'gekoppeld' wordt. Elk meisje kan dus ook maar aan één jongen gekoppeld worden, want ze kan maar naast één persoon zitten. In de graaf gaan we daarom een aantal takken kleuren en een tak kleuren we als de betreffende jongen en het betreffende meisje aan elkaar gekoppeld worden. Omdat we elke jongen en elk meisje willen koppelen zijn we in de graaf op zoek naar een kleuring van de takken. De gekleurde takken mogen niet aan elkaar grenzen. Grenzen twee gekleurde takken wel aan elkaar, dan zou dat betekenen dat een jongen of meisje aan twee personen wordt gekoppeld en dat kan niet. Een kleuring van de takken zó dat gekleurde takken niet aan elkaar grenzen noemen we een *koppeling*. Als we er voor kunnen zorgen dat elk knooppunt met een ander knooppunt verbonden is door precies één gekleurde tak, praten we over een *volmaakte koppeling*. In ons voorbeeld zijn we dus op zoek naar een volmaakte koppeling. Nu kan het ook zijn dat er een oneven aantal leerlingen en dus een oneven aantal knooppunten is. Óf er zitten wel een even aantal leerlingen in de klas, maar er zijn meer jongens dan meisjes (of andersom). In beide gevallen bestaat er dan dus geen volmaakte koppeling, want er blijven altijd leerlingen over die niet gekoppeld worden. In zo'n geval gaan we op zoek naar een kleuring van de takken zó dat zoveel mogelijk leerlingen gekoppeld zijn. We spreken dan niet van een volmaakte koppeling, maar van een *maximale koppeling*.

OPGAVEN

5 Is een volmaakte koppeling ook een maximale koppeling?
Is een maximale koppeling ook een volmaakte koppeling?

De vraag is nu alleen nog hoe we een volmaakte en/of maximale koppeling kunnen vinden.

3.3 Algoritme voor het bepalen een maximale koppeling

Voor een volmaakte koppeling moeten we in ieder geval hebben dat voor elk tweetal jongens het totale aantal meisjes waar de twee jongens naast willen zitten ook minstens twee is. Willen twee jongens namelijk allebei alleen maar naast één en hetzelfde meisje zitten, dan kunnen we nooit een volmaakte koppeling vinden. Het betreffende meisje kan namelijk nooit naast beide jongens komen te zitten. Hetzelfde hebben we voor elk drietal jongens. Het totale aantal meisjes waar de drie jongens naast willen zitten moet dan minstens drie zijn. En zo geldt dat voor elk k-tal jongens het totale aantal meisjes waar ze naast willen zitten minstens k moet zijn. Als dat het geval is, dan bestaat er een volmaakte koppeling. Het bewijs hiervan laten we achterwege. Is dit niet het geval, dan kunnen we alleen maar op zoek gaan naar een maximale koppeling en proberen zoveel mogelijk mensen te koppelen.

We gaan weer kijken naar ons voorbeeld (zonder Huub en Sanne). Het idee van het algoritme is om allereerst zo maar een aantal 'koppels' te maken. Hoe meer koppels je al kunt maken, hoe minder werk je daarna nog hoeft te doen. Geef de koppelingen aan met een pijl van Y naar X, oftewel met een pijl van het meisje naar de jongen. Van de andere takken in de graaf maken we een pijl van X naar Y. De volgende stap is om een knooppunt uit X te kiezen die nog niet gekoppeld is aan een knooppunt uit Y. Een knooppunt dat nog niet gekoppeld is noemen we een *vrij knooppunt*. Laten we het vrije knooppunt dat we kiezen uit X knooppunt x noemen. Loop nu via een pijl van de graaf naar een knooppunt uit Y. Loop nu via de pijlen steeds door totdat je in een vrij knooppunt van Y uit komt. Je hebt nu dus een pad gevonden, dat bestaat uit pijlen, van het vrije knooppunt x naar een vrij knooppunt in Y. Draai nu alle pijlen van dit pad om, zodat er nieuwe koppelingen ontstaan.
Zolang er nog vrije knooppunten zijn in X kunnen we steeds hetzelfde doen. Nu kunnen we twee dingen krijgen: óf er zijn geen vrije knooppunten meer óf er is geen pad te vinden van een vrij knooppunt uit X naar een vrij knooppunt uit Y. In beide gevallen bestaat er geen grotere koppeling meer. Wat we dus uiteindelijk zullen krijgen is een maximale koppeling. Als er in de graaf ook een volmaakte koppeling bestaat, zal het algoritme een volmaakte koppeling geven.
We zullen nu eerst het algoritme geven en vervolgens lichten we het algoritme toe aan de hand van een voorbeeld.

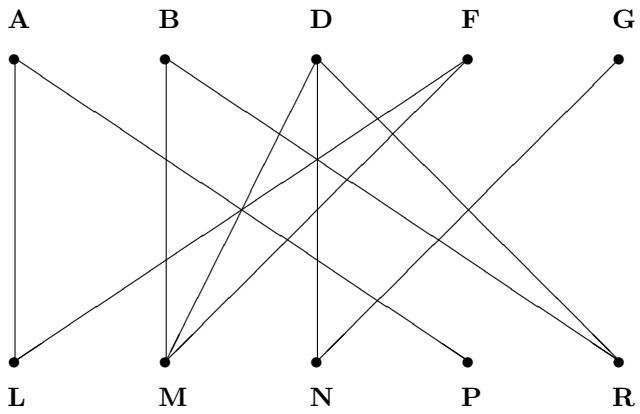
Ga uit van een graaf met jongens en meisjes waarvan nog geen takken zijn gekleurd. We noemen de verzameling knooppunten van de jongens weer X en die van de meisjes Y.

1. Koppel op de een of andere manier jongens aan een meisje en kleur de takken in de graaf die daar bij horen. Noem de koppeling M.
2. Maak de volgende gerichte graaf:
   - Neem dezelfde knooppunten als in de oorspronkelijke graaf.
   - Als een tak al gekleurd is in de koppeling M, maak dan een pijl van het knooppunt in Y naar het knooppunt in X.
   - Als een tak nog niet gekleurd is in de koppeling M, maak dan een pijl van het knooppunt in X naar het knooppunt in Y.
3. Kies een knooppunt x uit X dat nog niet gekoppeld is en zoek een pad in G van x naar een knooppunt uit Y dat nog niet gekoppeld is. Als dit niet kan, dan is M een maximale koppeling en stoppen we. Als dit wel kan, dan is dit het nieuwe pad.
4. Het pad heeft een even aantal knooppunten en dus een oneven aantal takken. Nummer de takken die in het pad zitten. De takken met een even nummer zitten dan in de koppeling M.

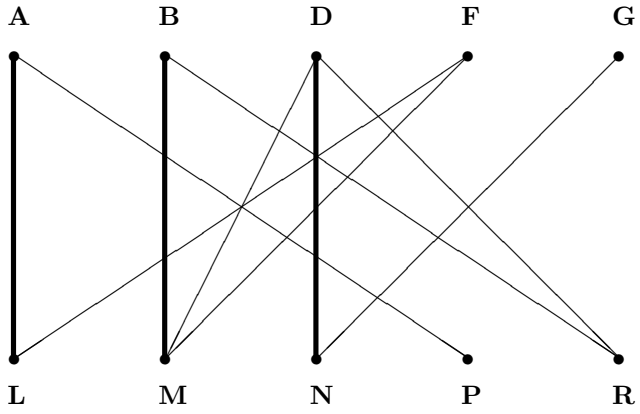
5. Verwijder de takken met een even nummer uit M en voeg de takken met een oneven nummer toe aan M.

6. Zijn alle knooppunten uit X gekoppeld, dan is M een volmaakte koppeling en stoppen we.
   Zijn nog niet alle knooppunten uit X gekoppeld, ga dan terug naar stap 2.

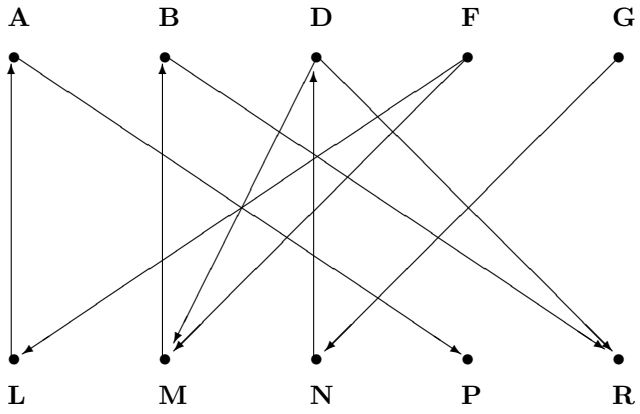
Nu we het algoritme gegeven hebben kunnen we het algoritme toelichten aan de hand van ons voorbeeld. Daarin hadden we de volgende graaf:



Eerste ronde
Stap 1. Allereerst gaan we op zoek naar een koppeling.
Laten we **A** koppelen aan **L**, **B** aan **M** en **D** aan **N**. We zien dat **F** en **G** nu niet meer te koppelen zijn. We krijgen de volgende kleuring van de takken:



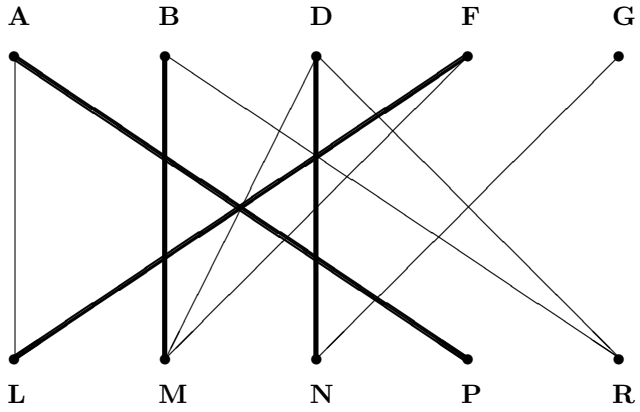
Stap 2. Voor de koppeling die we in stap 1 gevonden hebben krijgen we de volgende gerichte graaf:



Stap 3. We hebben knooppunt F nog niet gekoppeld, dus we nemen $x = \text{F}$. We nemen het volgende pad: $F(rank) \to L(aura) \to A(nton) \to P(auline)$. We komen dus uit bij Pauline en ook die was nog niet gekoppeld.

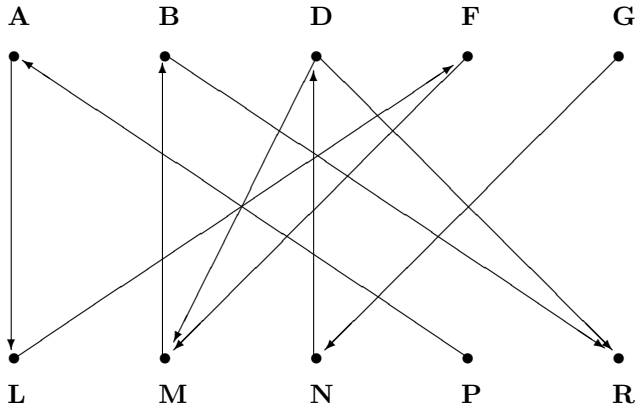
Stap 4. We nummeren de takken die in het pad zitten: $F \to L : 1$, $L \to A : 2$ en $A \to P : 3$.
We zien inderdaad dat tak 2 ook in de koppeling zit.

Stap 5. Tak 2 halen we weg uit de koppeling en tak 1 en 3 voegen we aan de koppeling toe. We krijgen dan de volgende kleuring van de takken:



Stap 6. Knooppunt G is nog niet gekoppeld, dus we gaan terug naar stap 2.

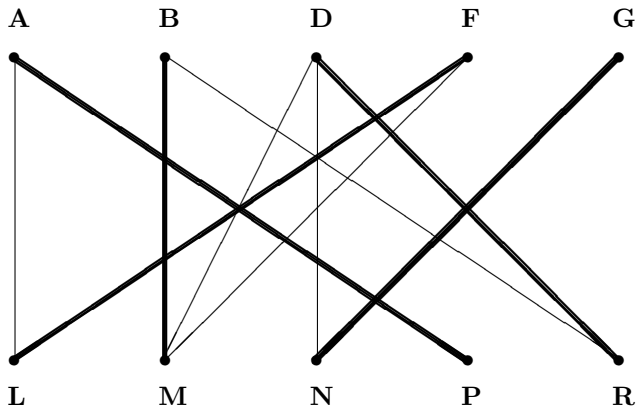
Tweede ronde
Stap 2. We hebben in stap 5 in de vorige ronde een nieuwe koppeling gekregen. Dat is de koppeling waar we nu mee verder gaan. Daar hoort dan de volgende gerichte graaf bij:



Stap 3. We hebben knooppunt G nog niet gekoppeld, dus we nemen $x =$ G. We nemen het volgende pad: $G(erard) \to N(atascha) \to D(irk) \to R(omy)$. We komen dus uit bij Romy en ook die was nog niet gekoppeld.

Stap 4. We nummeren de takken die in het pad zitten: $G \to N : 1$, $N \to D : 2$ en $D \to R : 3$.
We zien inderdaad dat tak 2 ook in de koppeling zit.

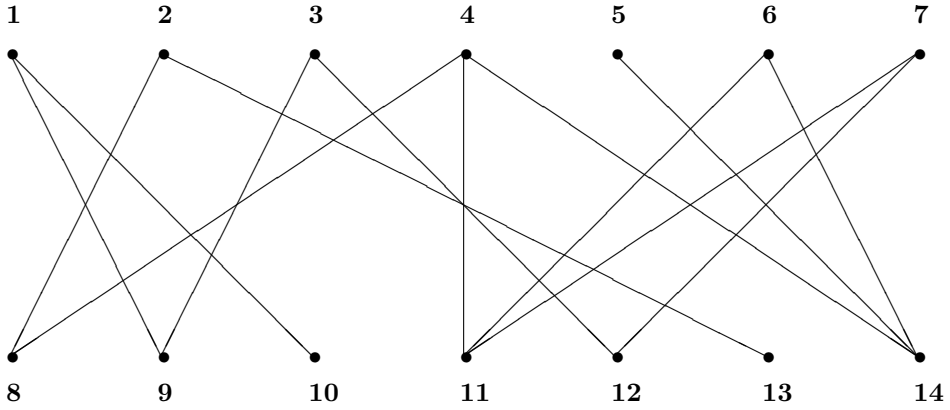
Stap 5. Tak 2 halen we weg uit de koppeling en tak 1 en 3 voegen we aan de koppeling toe. We krijgen dan de volgende kleuring van de takken:



Stap 6. Nu is elk knooppunt gekoppeld en hebben we de volmaakte koppeling gevonden. De kleuring die daar bij hoort is de kleuring van de bovenstaande graaf.

OPGAVEN

6 Bepaal een maximale koppeling van de onderstaande graaf.



7 Los het probleem 'Rooster' uit paragraaf 3.1.2 op, beginnend met de volgende 'koppels':
Persoon 1 - 13.00
Persoon 3 - 15.00
Persoon 4 - 15.30
Persoon 5 - 14.00.

8 Los het probleem 'Bedrijf met opdrachten' uit paragraaf 3.1.3 op.

3.4 Koppelingen met voorkeurslijsten

We gaan nog eens kijken naar het probleem van de docent om de kinderen in de klas naast elkaar te zetten zodanig dat een jongen en een meisje naast elkaar zitten. Hierboven hebben we de jongens uit de klas een lijst in laten vullen met daarop de namen van de meisjes waar ze wel naast willen zitten. Met behulp van die lijst hebben we een oplossing gevonden van het probleem. Maar we kunnen het probleem ook op een andere manier aanpakken. In plaats van één lijst, krijgen we nu twee lijsten. Je geeft alle jongens een lijst met alle namen van de meisjes en alle meisjes een lijst met alle namen van de jongens. Op de lijst geeft iedereen zijn voorkeur aan. Iedereen geeft dus aan naast wie ze het liefst willen zitten, naast wie ze als tweede keuze willen zitten, enzovoorts. Je krijgt dus van iedereen een voorkeurslijst. Neem bijvoorbeeld een klas met daarin 4 jongens (Huib, Igor, Johan en Kees) en 4 meisjes (Sanne, Tessa, Vera en Wendy). Je laat iedereen zijn voorkeurslijst opschrijven. De gegevens staan in de onderstaande tabel:

Naam jongen	Nr.1	Nr.2	Nr. 3	Nr.4
Huib	Sanne	Tessa	Vera	Wendy
Igor	Sanne	Wendy	Vera	Tessa
Johan	Tessa	Sanne	Vera	Wendy
Kees	Wendy	Tessa	Sanne	Vera

Naam meisje	Nr.1	Nr.2	Nr. 3	Nr.4
Sanne	Kees	Johan	Igor	Huib
Tessa	Huib	Johan	Kees	Igor
Vera	Igor	Johan	Kees	Huib
Wendy	Johan	Igor	Huib	Kees

Ook nu gaan we weer koppels maken en ook hier zijn we dus weer op zoek naar een koppeling. Alleen het vinden van 'de beste' koppeling gaat iets anders als hiervoor. Als we willekeurig koppels maken dan kan het gebeuren dat er een jongen en een meisje zijn die beiden liever aan elkaar gekoppeld worden dan aan hun huidige toegewezen persoon. Als dat het geval is spreken we van een *instabiele koppeling*. Is een koppeling niet instabiel, dan een noemen we de koppeling een *stabiele koppeling*. We willen in ieder geval een stabiele koppeling hebben.

We gaan eens kijken naar de volgende (willekeurige) koppeling;
- Huib ↔ Tessa
- Igor ↔ Vera
- Johan ↔ Sanne
- Kees ↔ Wendy.

We pakken er twee personen uit: Igor en Wendy. In de eerste tabel kunnen we zien dat Igor liever aan Wendy is gekoppeld dan aan Vera. In de tweede tabel kunnen we zien dat Wendy liever aan Igor is gekoppeld dan aan Kees. We hebben nu dus een jongen en een meisje gevonden, namelijk Igor en Wendy, die liever aan elkaar gekoppeld zijn dan aan hun huidige partner. We hebben hier dus te maken met een instabiele koppeling. Dit is dan ook geen goede oplossing voor het probleem.

De volgende koppelinggen zijn wel stabiele koppelingen (Ga dit na!):

-	Huib	↔	Tessa	-	Huib	↔	Tessa
-	Igor	↔	Vera	-	Igor	↔	Wendy
-	Johan	↔	Wendy	-	Johan	↔	Vera
-	Kees	↔	Sanne	-	Kees	↔	Sanne.

We pakken er weer twee personen uit: Igor en Johan. In de eerste tabel kunnen we zien dat Igor liever aan Wendy is gekoppeld dan aan Vera en dat voor Johan precies het tegenovergestelde geldt. Beiden geven dus de voorkeur aan de tweede stabiele koppeling. We zeggen dat een koppeling *'mannenoptimaal'* is als iedere jongen in iedere andere stabiele koppeling niet beter af is, oftewel niet gekoppeld is aan een meisje die de voorkeur heeft boven zijn huidige partner.

OPGAVEN

9 Ga na dat de tweede koppeling mannenoptimaal is.

Ons uiteindelijke doel is om een mannenoptimale stabiele koppeling te vinden. Om een oplossing van het probleem te vinden zouden we een graaf kunnen maken. Maar dat is niet erg zinvol. Wel bestaat er een algoritme om het probleem op te lossen. Het idee achter dit algoritme is om de jongens één voor één aan een meisje te koppelen. Krijg je te maken met het feit dat een meisje al gekoppeld is, dan gaan we

kijken aan wie de meisje liever gekoppeld is. En zo gaan we net zo lang door totdat elke jongen aan een meisje is gekoppeld. Het algoritme staat in de volgende paragraaf netjes uitgeschreven. Het algoritme is 'mannenoptimaal'. Uiteraard kunnen we het algoritme ook 'vrouwenoptimaal' maken door de meisjes één voor één te koppelen aan een jongen.

3.4.1 Algoritme voor het bepalen van een stabiele koppeling

Ga er van uit dat er voor iedereen een voorkeurslijst bekend is.

1. Begin: Koppel jongen 1 aan de meisje van zijn hoogste voorkeur.
2. Uitbreiding: Koppel de volgende jongen, zeg jongen i, aan de meisje van zijn volgende voorkeur (begin bij de hoogste), zeg dit is meisje j.
   a. Als meisje j jongen i accepteert, dat wil zeggen dat ze ófwel nog niet gekoppeld was ófwel ze wordt liever gekoppeld aan jongen i dan aan haar huidige partner; koppel jongen i aan meisje j.
   Als meisje j haar huidige partner, zeg jongen k, verlaat; probeer jongen k te koppelen volgens dezelfde procedure
   b. Als meisje j jongen i niet accepteert, ga terug naar stap 2.
3. Controle: Zijn alle jongens gekoppeld? Zo nee, ga naar stap 2. Zo ja, je hebt een stabiele mannenoptimale koppeling.

Na elke ronde zetten we alle koppels netjes in een tabel. Zo kunnen we goed bijhouden wie er wél en wie er nog niet gekoppeld is.
We passen het algoritme toe met de voorkeurslijsten van ons voorbeeld.

Eerste ronde
Stap 1. We koppelen Huib allereerst aan Sanne.

Stap 2. Koppel Igor aan Sanne.
Stap 2a. Sanne was al gekoppels aan Huib. Maar Sanne accepteert Igor, want ze is liever aan Igor gekoppels dan aan Huib. We gaan Huib dus weer volgens dezelfde procedure proberen te koppelen.

Stap 2. Koppel Huib aan zijn tweede keuze, Tessa.
Stap 2a. Tessa accepteert Huib, want zij was nog niet gekoppeld.

Naam jongen	Gekoppeld aan
Huib	Tessa
Igor	Sanne
Johan	-
Kees	-

Stap 3. Nog niet alle jongens zijn gekoppeld, dus we gaan terug naar stap 2.

Tweede ronde
Stap 2. Koppel Johan aan Tessa.
Stap 2a. Tessa was al gekoppeld aan Huib. Ze accepteert Johan niet, want ze is liever gekoppeld aan Huib dan aan Johan. We gaan dus terug naar stap 2.

Stap 2. Koppel Johan aan zijn tweede keuze, Sanne.
Stap 2a. Sanne was al gekoppels aan Igor. Maar Sanne accepteert Johan, want ze is liever aan Johan gekoppels dan aan Igor. We gaan Igor dus weer volgens dezelfde procedure proberen te koppelen.

Stap 2. Koppel Igor aan zijn tweede keuze, Wendy.
Stap 2a. Wendy accepteert Igor, want zij was nog niet gekoppeld

Naam jongen	Gekoppeld aan
Huib	Tessa
Igor	Wendy
Johan	Sanne
Kees	-

Stap 3. Nog niet alle jongens zijn gekoppeld, dus we gaan terug naar stap 2.

Derde Ronde
Stap 2. Koppel Kees aan Wendy.
Stap 2b. Wendy was al gekoppeld aan Igor. Ze accepteert Kees niet, want ze is liever gekoppeld aan Igor dan aan Kees. We gaan dus terug naar stap 2.

Stap 2. Koppel Kees aan zijn tweede keuze, Tessa.
Stap 2a. Tessa was al gekoppeld aan Huib. Ze accepteert Kees niet, want ze is liever gekoppeld aan Huib dan aan Kees. We gaan dus terug naar stap 2.

Stap 2. Koppel Kees aan zijn derde keuze, Sanne.
Stap 2a. Sanne was al gekoppeld aan Johan. Maar Sanne accepteert Kees, want ze is liever aan Kees gekoppeld dan aan Johan. We gaan Johan dus weer volgens dezelfde procedure proberen te koppelen.

Naam jongen	Gekoppeld aan
Huib	Tessa
Igor	Wendy
Johan	?
Kees	Sanne

Stap 2. Koppel Johan aan zijn derde keuze, Vera.
Stap 2a. Vera accepteert Johan, want zij was nog niet gekoppeld.

Naam jongen	Gekoppeld aan
Huib	Tessa
Igor	Wendy
Johan	Vera
Kees	Sanne

Stap 3. Alle jongens zijn gekoppeld, dus we hebben een stabiele mannenoptimale koppeling gevonden.

Het bewijs dat het algoritme inderdaad een stabiele mannenoptimale koppeling geeft laten we hier achterwege.

OPGAVEN

10 Bepaal een stabiele vrouwenoptimale (!!) koppeling met behulp van de onderstaande voorkeurslijsten.

Naam meisje	Nr.1	Nr.2	Nr. 3	Nr.4	Nr.5
Laura	Bart	Frank	Gerard	Dirk	Arnold
Mariska	Frank	Dirk	Gerard	Arnold	Bart
Natascha	Frank	Bart	Arnold	Dirk	Gerard
Pauline	Bart	Dirk	Arnold	Frank	Gerard
Romy	Bart	Dirk	Arnold	Gerard	Frank

Naam jongen	Nr.1	Nr.2	Nr. 3	Nr.4	Nr.5
Arnold	Mariska	Romy	Pauline	Laura	Natascha
Bart	Laura	Mariska	Natascha	Pauline	Romy
Dirk	Mariska	Natascha	Romy	Pauline	Laura
Frank	Laura	Natascha	Mariska	Pauline	Romy
Gerard	Romy	Natascha	Mariska	Laura	Pauline

Hoofdstuk 4

Stapsgewijze optimalisatie

4.1 Introductie

In dit hoofdstuk gaan we kijken naar problemen waarbij we bepaalde keuzes moeten maken. Alleen kunnen we niet alle keuzes maken, omdat we te maken hebben met bepaalde beperkingen. Hieronder staan drie voorbeelden, waar we graag een optimale oplossing voor zouden willen vinden.

4.1.1 Ik ga op reis en ik neem mee...

Je gaat op reis, maar je zou bij voorkeur je hele kledingkast mee willen nemen. Maar daar is je koffer helaas niet groot genoeg voor. Je wilt het liefst 10 spullen, ofwel 10 items, meenemen. Totaal hebben die een grootte van 48 cm^3. Maar in je koffer past maar 30 cm^3. Je bepaalt voor jezelf welke items je liever mee wilt nemen dan andere. Je geeft elk van de 10 items dus een bepaalde waarde. De waarde en de grootte van de verschillende items staan in de tabel hieronder.

item	1	2	3	4	5	6	7	8	9	10
waarde	16	22	12	8	20	6	10	25	14	13
grootte	5	7	4	3	6	3	3	9	4	4

OPGAVEN

1 Welke items neem jij mee om de totale waarde van de items die je meeneemt zo groot mogelijk te maken, maar wel op zo een manier dat je koffer dicht kan?

4.1.2 Projecten

Je hebt een eigen klusbedrijfje en je hebt een aantal verschillende klusjes liggen. Elke klus levert natuurlijk geld op, maar ze kosten ook wat. De kosten van een klus, ook wel project genoemd, en het bedrag dat het oplevert zijn voor elk project verschillend. Je hebt de volgende vier projecten liggen:

1. Keuken verbouwen
2. Badkamer verbouwen
3. Kantoorpand verbouwen
4. Zolder opknappen

In de tabel hieronder staan de kosten (in euro) van deze vier projecten en de opbrengst van de projecten. opleveren.

project	kosten	opbrengst
1	5.000	16.000
2	4.000	12.000
3	7.000	22.000
4	3.000	8.000

Maar je hebt zelf maar een kasvoorraad van 14.000 euro waar je alle kosten die je voor de projecten maakt mee moet betalen.

OPGAVEN

2 Welke projecten zou je wel doen en welke niet?

4.1.3 Hulp aan Derde Wereld landen

De Flying Doctors werken al jaren aan een betere gezondheid in Afrika. Om ervoor te zorgen dat de gezondheid in Afrika beter wordt hebben ze vier medische teams die uitgezonden kunnen worden naar vijf Derde Wereld landen. De organisatie wil dat de totale leeftijd, dat wil zeggen de gemiddelde leeftijd maal het aantal inwoners, zo hoog mogelijk wordt. Ze noemen dit ook wel de effectiviteit van de hulp. Hoe meer medische teams er naar een land gestuurd worden, hoe groter de effectiviteit is, oftewel hoe hoger de totale leeftijd is van het land. In de onderstaande tabel staat voor alle vijf landen de effectiviteit gegeven, afhankelijk van het aantal teams dat er heen gestuurd wordt.

	effectiviteit: totale leeftijd				
aantal teams	land 1	land 2	land 3	land 4	land 5
0	0	0	0	0	0
1	45	20	50	30	35
2	70	45	70	50	60
3	90	75	80	70	85
4	105	110	100	95	110

OPGAVEN

3 Probeer het probleem 'Hulp aan Derde Wereld landen' eens op te lossen met het gezonde verstand.

Alle bovenstaande problemen hebben hetzelfde karakter. We hebben een aantal gegevens met een aantal beperkingen. De beperking in het eerste voorbeeld is de grootte van de koffer, in het tweede voorbeeld is dit de kasvoorraad en bij het derde voorbeeld is het het aantal medische teams. Ook wil je in alle voorbeelden een bepaalde som zo hoog mogelijk hebben. In voorbeeld 1 wil je de som van de waarden van de items die je meeneemt zo hoog mogelijk hebben. Bij voorbeeld 2 wil je je winst (opbrengst - kosten) zo hoog mogelijk hebben. De som van de totale leeftijd is wat je bij voorbeeld 3 zo hoog mogelijk wilt hebben. Alle voorbeelden zien er dus hetzelfde uit, alleen zijn ze allemaal anders omschreven. Eén van de voorbeelden zullen we hier uitwerken. Voor de rest van de voorbeelden geldt dat het idee van het oplossen hetzelfde is.

4.2 Knapzakprobleem

Het eerste probleem van de vorige paragraaf wordt ook wel het *knapzakprobleem* genoemd. Dit probleem werken we uit om je een idee te geven hoe je de andere problemen ook op zou kunnen lossen.
We hebben dus de volgende waarden en de grootten van de verschillende items:

item	1	2	3	4	5	6	7	8	9	10
waarde	16	22	12	8	20	6	10	25	14	13
grootte	5	7	4	3	6	3	3	9	4	4

De vraag is nu welke items jij meeneemt om de totale waarde van al de items die je meeneemt zo groot mogelijk te maken, maar wel op zo een manier dat je koffer dicht kan.

We kunnen dit probleem op meerdere manieren aanpakken.
Als eerste kunnen we er voor kiezen om de kleinste items mee te nemen zodat we lekker veel mee kunnen nemen op reis. In ons voorbeeld kiezen we er dan voor om de items 4, 6 en 7 mee te nemen. De totale waarde van de koffer is dan gelijk geworden aan 24. De drie gekozen items hebben een totale grootte van 9 cm^3, dus kunnen we nog veel meer in de koffer kwijt. We gaan verder met inpakken en nemen de items 3, 9 en 10 mee. We hebben al 6 items in de koffer zitten! Totaal hebben deze een grootte van 21 cm^3 en een waarde van 63. Dat schiet al aardig op. Nog 9 cm^3 over in de koffer, dus we kunnen nog wel een item meenemen. Het volgende item met de kleinste grootte is item 1. Nu kan er nog maar 4 cm^3 mee, maar de overige items zijn helaas allemaal groter dan 4 cm^3. Hier houdt het dus op en de inhoud van de koffer zal dus blijven zoals die nu is. Oftewel, we hebben in totaal nu 7 items mee, namelijk de items 1, 3, 4, 6, 7, 9, 10. De koffer heeft daardoor een inhoud van 26 cm^3 en een waarde van 79 gekregen.
Maar we zien dat als we steeds de kleinste items kiezen (zoals hierboven gedaan is) dat we, nadat er al 6 items in de koffer zitten, nog 9 cm^3 aan ruimte over hebben. Het blijkt dat we nu geen twee items meer kunnen toevoegen. We moeten dus één item kiezen uit de overgebleven vier items; items 1, 2, 5 en 8. Hierboven gingen we verder met het kiezen van items net zoals we daarvoor al deden. Oftewel we kozen van de overgebleven items het kleinste item. Maar het is dan natuurlijk slimmer om het item met de hoogste waarde mee te nemen, in plaats van het kleinste overgebleven item. Dat zou dus betekenen dat we niet item 1, maar item 8 mee zouden nemen. Dit item zorgt ervoor dat de koffer echt helemaal vol komt te zitten en dat we een totale waarde van 88 hebben gekregen. En dit met een koffer waar maar liefst 7 items in zitten!

In plaats van de kleinste items mee te nemen kunnen we de items ook op volgorde van de hoogste waarde meenemen. We kunnen dan misschien wel minder items meenemen, maar wellicht wordt de waarde van de koffer dan wel hoger. In ons knapzakprobleem komt dat er op neer dat we als eerste item 8 in de koffer stoppen. We hebben dan nog $30 - 9 = 21$ cm^3 over in de koffer. Als tweede gaat item 2 mee en dan kan er nog $21 - 7 = 14$ cm^3 aan items mee. Item 5 gaat dan als derde mee op reis en daarmee houden we nog $14 - 6 = 8$ cm^3 ruimte over in de koffer. De totale waarde van de koffer is inmiddels 67 geworden. Als vierde stoppen we item 1 in de koffer en dan kan er nog maar $8 - 5 = 3$ cm^3 in. We kunnen nu alleen nog kiezen uit item 4, 6 of 7, want de rest van de items zijn te groot om nog mee te nemen. Uit deze 3 items kiezen we het item met de hoogste waarde, ofwel we kiezen item 7 uit om als laatste in de koffer te stoppen. Nu zit te koffer echt vol en zullen we uiteindelijk item 1, 2, 5, 7 en 8 meenemen op reis en samen hebben ze een waarde van 93. We zien dat we in dit geval maar 5 items mee kunnen nemen, maar de waarde van de koffer is wel hoger dan het vorige geval.

OPGAVEN

4 We kunnen het bovenstaande knapzakprobleem ook als volgt aanpakken: we nemen de items mee op volgorde van de hoogste waarde per volume-eenheid, ofwel de hoogste waarde per cm^3. Laat zien dat de oplossing van het knapzakprobleem dan zó is dat we de items 1, 2, 5, 7, 9 en 10 zullen meenemen op reis. Bepaal tevens de waarde van de koffer.

De bovenstaande methoden om het knapzakprobleem aan te pakken zijn goede manieren om de oplossing van het probleem te benaderen. We hebben de volgende vier oplossingen gekregen:

- Kleinste items kiezen: we nemen item 1, 3, 4, 6, 7, 9 en 10 mee. Totale waarde = 79.
- Kleinste items kiezen en als laatste het item met de hoogste waarde: we nemen item 3, 4, 6, 7, 8, 9 en 10 mee. Totale waarde = 88.
- Items met hoogste waarde kiezen: we nemen item 1, 2, 5, 7 en 8 mee. Totale waarde = 93.
- Items met hoogste waarde per cm^3 kiezen: we nemen item 1, 2, 5, 7, 9 en 10 mee. Totale waarde = 95.

En bij enkele voorbeelden zal één van deze vier manieren ook echt de optimale oplossing geven. In ons voorbeeld is dit echter niet het geval. Als we er namelijk voor kiezen om item 1, 2, 3, 5, 9 en 10 mee te nemen op reis dan krijgt de koffer een totale waarde van 97. Deze waarde is dus hoger dan de waarden die we hebben gevonden bij de vorige vier methoden. Later zal blijken dat deze laatste oplossing ook de maximale waarde van de koffer geeft, maar om op deze oplossing van het probleem te komen hebben we een stukje wiskunde nodig. Om deze wiskunde duidelijk te maken gebruiken we het volgende voorbeeld.

Voorbeeld 1
We hebben de 5 items met de volgende waardes en groottes:

item	1	2	3	4	5
waarde	16	11	7	8	4
grootte	5	3	2	2	1

Deze items moeten in een koffer met grootte 7.

OPGAVEN

5 Los het bovenstaande probleem op door steeds de kleinste items mee te nemen. Wat is nu de waarde van de koffer?

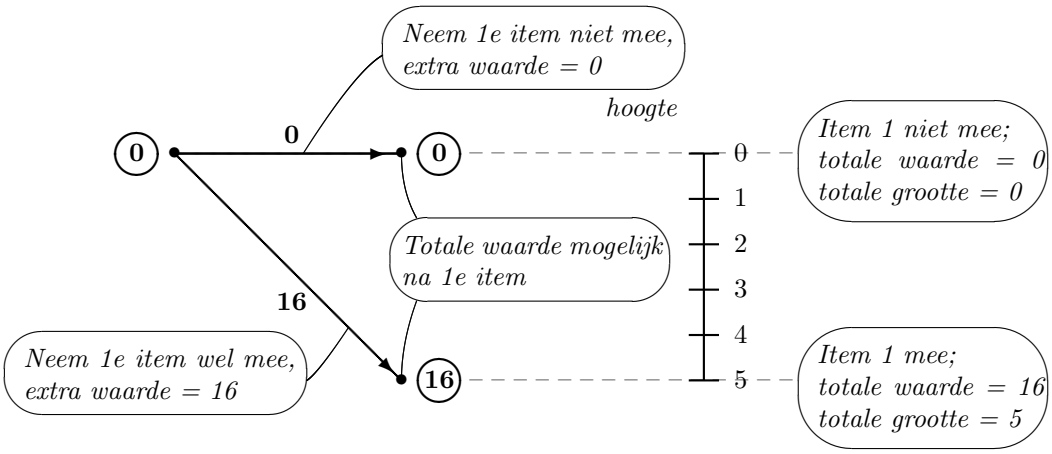
6 Los het bovenstaande probleem op door de items op volgorde van de hoogste waarde mee te nemen. Wat is nu de waarde van de koffer?

7 Los het bovenstaande probleem op door de items op volgorde van de hoogste waarde per cm^3 mee te nemen. Wat is nu de waarde van de koffer?

8 Zie je een combinatie van items die je mee kunt nemen, zó dat je een hogere waarde hebt dan de waardes uit de bovenstaande drie opgaven?

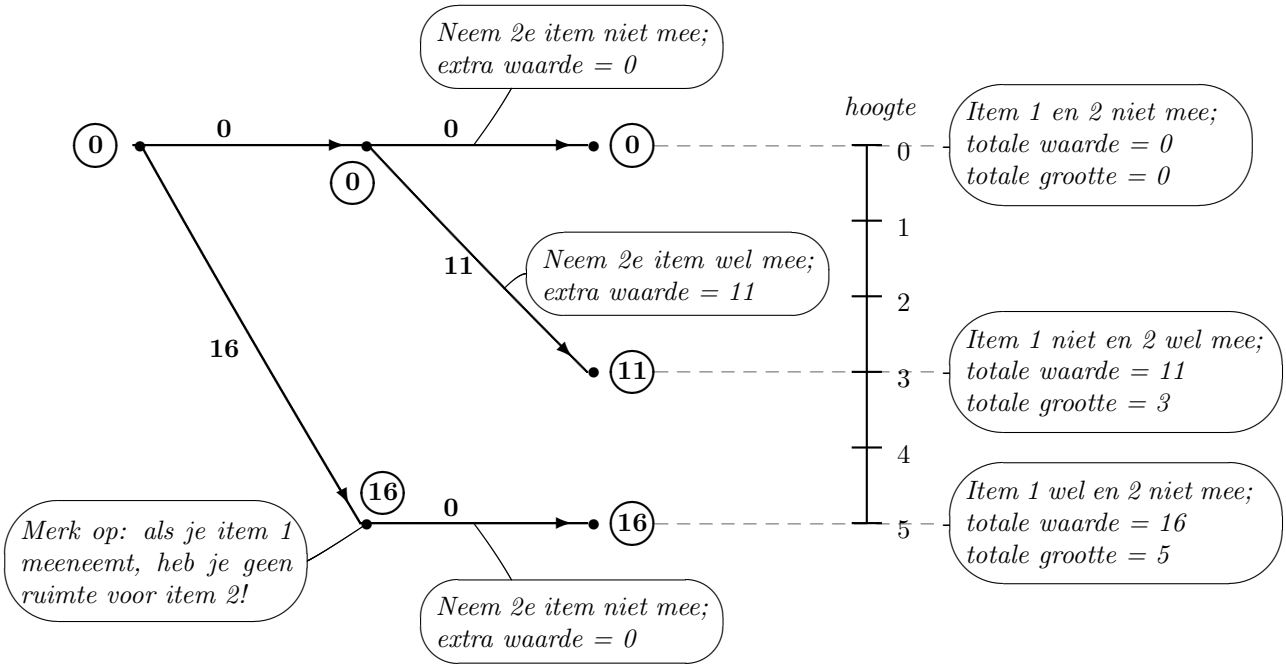
Door dit probleem met een wiskundige methode aan te pakken, zullen we zien dat we altijd op de optimale oplossing uit zullen komen.

Om het probleem wat duidelijker te maken gaan we van dit probleem eerst een graaf maken. Als we alleen naar het eerste item kijken ziet de graaf er als volgt uit:



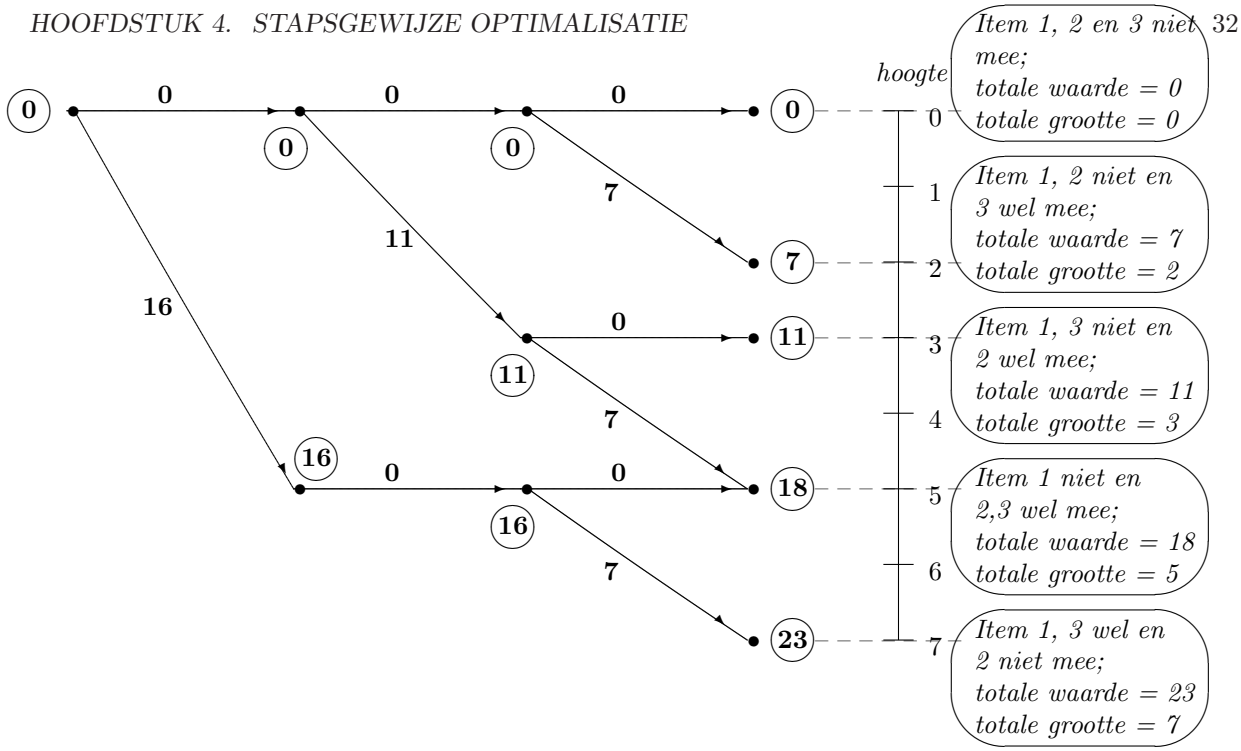
We beginnen met een beginknooppunt. Vanuit dat punt tekenen we twee pijlen; de eerste pijl geeft aan dat we item 1 niet meenemen in de koffer en de tweede pijl geeft aan dat we item 1 wél meenemen. Nemen we item 1 niet mee, dan verandert de inhoud van de koffer niet. Het eerste knooppunt zal dus op een hoogte 0 blijven zitten. Nemen we item 1 wel mee, dan krijgt de koffer een inhoud van 5 cm^3. Het tweede knooppunt komt dus op een hoogte 5 te zitten.
Ook geven we deze twee pijlen een waarde. Door item 1 niet in de koffer te doen, verandert er dus ook niets aan de totale waarde van de koffer. De eerste pijl geven we dus een waarde 0. Door item 1 wél in de koffer te doen, wordt de totale waarde van de koffer nu verhoogd met een waarde 16, namelijk de waarde van het eerste item. De tweede pijl krijgt dus waarde 16.
Ook de knooppunten geven we een waarde, waarmee we aangeven wat de totale waarde van de koffer op dat moment is. In het beginknooppunt hebben we nog niets aan de koffer toegevoegd en dus geven we dat knooppunt de waarde 0. Om verwarring met de waarde van de pijl te voorkomen zetten we om de waarde van de knooppunten een cirkel. Bij het knooppunt rechts bovenin zitten er nog geen items in de koffer en dus krijgt dat knooppunt ook de waarde 0. Als we in het knooppunt rechts onderin zitten hebben we item 1 in de koffer zitten en dus krijgt dat knooppunt een waarde 16.

Nu kijken we naar item 2. Op dezelfde manier als bij item 1 gaan we weer pijlen zetten naar nieuwe knooppunten. Als we zeggen dat het beginknooppunt in de 'nulde' kolom staat, dan staan de knooppunten die we er hierboven bij gezet hebben in de eerste kolom. De knooppunten die we er nu bij gaan zetten, zetten we in een tweede kolom. De nieuwe pijlen die we gaan tekenen gaan lopen van een knooppunt uit de eerste kolom naar een knooppunt uit de tweede kolom. Bij het zetten van de nieuwe pijlen moeten we er wel op letten of item 2 nog wel in de koffer past.
We krijgen, bij het probleem met alleen de eerste twee items, de volgende graaf:

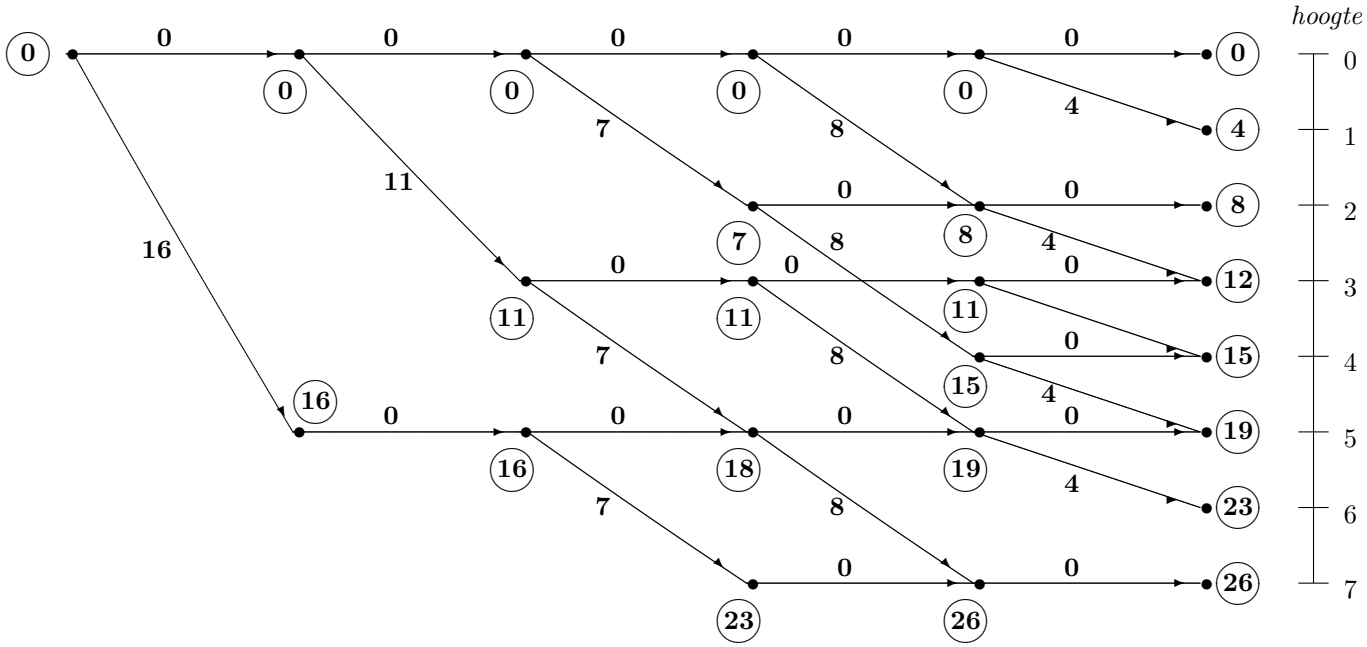


Zoals je in de graaf kunt zien hebben we er drie nieuwe pijlen bijgetekend die lopen van een knooppunt uit de eerste kolom naar een knooppunt uit de tweede kolom. De twee horizontale pijlen geven aan dat we item 2 niet meenemen en de pijl schuin naar beneden geeft aan dat we item 2 wel meenemen. Ook hebben we de pijlen en de knooppunten weer een waarde gegeven. De nieuwe pijlen geven de extra waarde aan van de koffer. Omdat we item 2 bij de twee horizontale pijlen niet meenemen is daar de extra waarde 0. Bij de pijl schuin naar beneden krijgt de koffer een extra waarde van 11, namelijk de extra waarde die item 2 toevoegt en dus krijgt die pijl een waarde 11.
Verder zie je in de graaf dat het tweede knooppunt uit de tweede kolom op een hoogte 3 zit, omdat we daar item 1 niet meenemen en item 2 wel en we dus een totale grootte van 3 krijgen. De waarde van dit knooppunt, oftewel de waarde van de koffer als we item 1 niet en item 2 wel meenemen, wordt gelijk aan 11.
Het derde en tevens onderste knooppunt uit de tweede kolom zit op hoogte 5, omdat we daar item 1 wel meenemen en item 2 niet en we dus een totale grootte van 5 hebben. Daar wordt de waarde van het knooppunt gelijk aan 16, want we nemen alleen item 1 mee.

Item 3 is aan de beurt. Op dezelfde manier als hierboven krijgen we de volgende graaf, behorende bij het probleem met de eerste drie items:



Zo kunnen we ook het vierde en het vijfde item afgaan. Met de dikgedrukte cijfers bij de pijlen geven we de waarde van de betreffende pijl aan. Het omcirkelde dikgedrukte getal geeft de waarde van het betreffende knooppunt aan. De uiteindelijke graaf die we krijgen, en die hoort bij het volledige probleem met alle vijf items, is de volgende:



OPGAVEN

9 Stel dat we de waarde van item 1 veranderden in 18. Wat verandert er dan in de graaf? Teken de graaf die hoort bij dit probleem.

10 Stel we voegen een extra item toe met waarde 3 en grootte 1. Teken de bijbehorende graaf.

Nu we de graaf hebben kunnen we het probleem op gaan lossen. De oplossing die we zoeken kunnen we nu uit de graaf halen: ga naar het knooppunt met de hoogste waarde (26) en loop terug in de graaf naar de nulde kolom.

11 Welke items neem je mee in de optimale oplossing?

12 Maak de graaf die hoort bij het volgende knapzakprobleem:

item	1	2	3	4	5	6	7
waarde	21	20	18	12	11	7	1
grootte	7	6	5	4	3	2	1

De items moeten in een koffer die 11 cm^3 groot is.
Wat is de maximale waarde van de koffer en welke items neem je daarvoor mee?

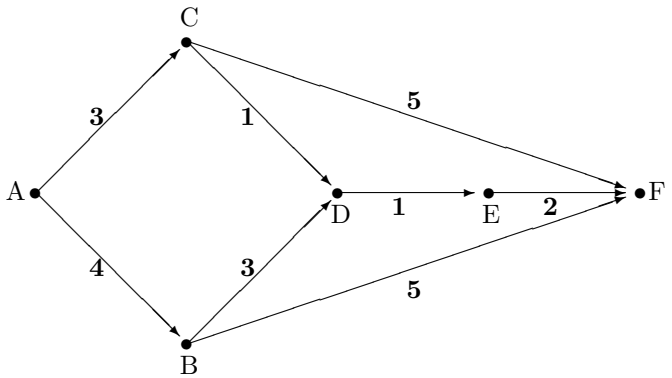
Bij het probleem uit ons voorbeeld konden we vrij gemakkelijk een graaf maken. Voor ons voorbeeld hadden we te maken met 5 items. Krijgen we problemen met 10 items, dan wordt het maken van een graaf al wat meer werk en zal de graaf al niet meer op één blaadje passen. Je kunt je voorstellen dat bij problemen met meer dan 10 items de graaf wel heel erg groot wordt. Bij zulke problemen gebruiken we liever een andere manier om het probleem op te lossen, waarbij we de graaf niet hoeven te tekenen. Een goede manier om problemen met veel items op te lossen is *dynamische programmering*. Deze manier van oplossen wordt uitgelegd in één van de afsluitende opgaven. Het is een formalisering van het vorige en kan dus eventueel overgeslagen worden.

4.3 Langste en kortste pad

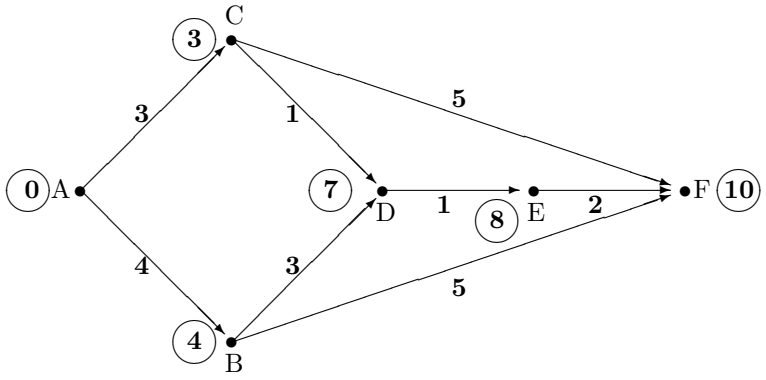
We gaan nog eens goed kijken naar het bepalen van de waarde van de knooppunten in de graaf. Die waarde krijgen we door bepaalde items wél en andere items niet mee te nemen. De waarde van het knooppunt krijgen we door alle waarden van de items die we wel meenemen bij elkaar op te tellen. Maar we zien ook dat de pijlen van de graaf de waarden hebben van de items. We kunnen in een bepaald knooppunt komen door over een aantal pijlen te 'lopen'; we kiezen ook wel een *pad* om van het beginknooppunt bij het gekozen knooppunt te komen. De som van alle waarden van de pijlen die in het pad zitten, noemen we ook wel de *lengte van een pad*. Omdat de pijlen de waarden van de items aangeven, krijgen we dat de waarde van een knooppunt gegeven wordt door de lengte van het gekozen pad. En omdat we op zoek zijn naar de maximale waarde van de knooppunten, zijn we op zoek naar het *langste pad* van het beginknooppunt naar het gekozen knooppunt. Als oplossing van het probleem zoeken we uiteindelijk naar het langste pad van het beginknooppunt naar de knooppunten in de laatste kolom.

We kunnen natuurlijk ook het langste pad bepalen van grafen die niet horen bij een knapzakprobleem. Het idee van het zoeken naar het langste pad blijft precies hetzelfde. Ga de knooppunten van links naar rechts af. Bekijk op welke manieren je in een knooppunt kunt komen vanuit het beginknooppunt en bepaal van alle paden die je vindt de lengte. Neem van alle gevonden lengtes het maximum en geef het knooppunt deze waarde. De waarde van knooppunten die je al gehad hebt kun je natuurlijk ook hier weer gebruiken. Ga zo alle knooppunen van links naar rechts af en zo vind je uiteindelijk het langste pad van het begin- naar het eindknooppunt.

Voorbeeld 2
We gaan van de onderstaande graaf het langste pad bepalen.



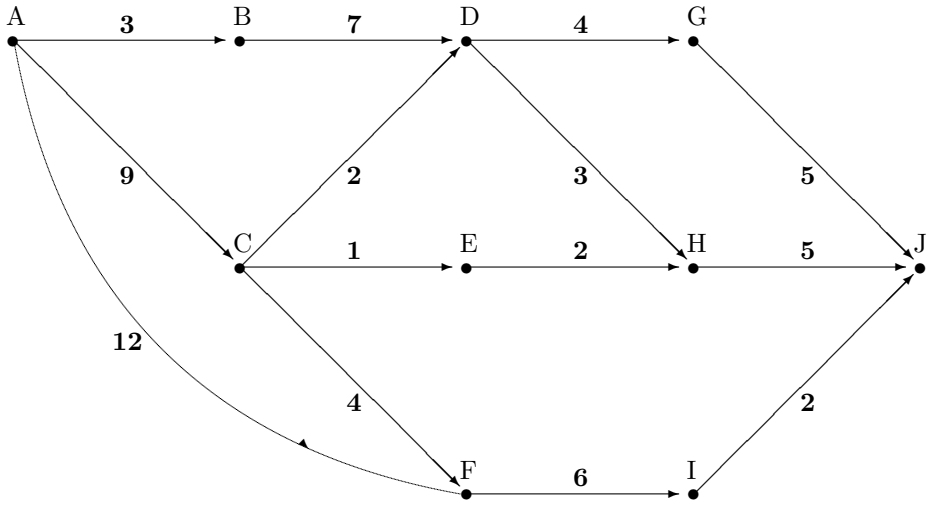
We beginnen bij het knooppunt waar alleen maar pijlen vandaan gaan, knooppunt A. Dit knooppunt geven we een waarde 0. Vervolgens gaan we naar knooppunt C. Omdat de pijl van A naar C een waarde 3 heeft, krijgt ook knooppunt C een waarde 3. Knooppunt B geven we om dezelfde reden waarde 4. Dan is knooppunt D aan de beurt. We kunnen in dat knooppunt komen vanuit B of C. We krijgen dus óf een waarde $4 + 3 = 7$ (als we vanuit B komen) óf een waarde $3 + 1 = 4$ (als we vanuit C komen). We nemen weer het maximum en dus krijgt knooppunt D een waarde 7. In knooppunt E kunnen we alleen komen vanuit D, dus dit knooppunt krijgt een waarde $7+1 = 8$. Als laatste krijgen we knooppunt F. We kunnen in F komen vanuit C of vanuit E en dus krijgt het knooppunt óf waarde $3 + 5 = 8$ óf $8 + 2 = 10$. Ook hier nemen we weer het maximum en dus krijgt knooppunt F een waarde 10. We zetten de waarden van de knooppunten weer in de graaf met een rondje er omheen.



Het langste pad van de bovenstaande graaf is dus het pad A-B-D-E-F van lengte 10.

OPGAVEN

13 Bekijk de onderstaande graaf. De knooppunten geven we aan met letters.
Bepaal het langste pad van knooppunt A naar het knooppunt J in de onderstaande graaf.



Net zoals er het langste pad bestaat in een graaf, bestaat er natuurlijk ook een kortste pad. Voor het bepalen van het langste pad nemen we in de knooppunten het maximum van een aantal waarden. Voor het bepalen van het kortste pad doen we verder precies hetzelfde, alleen nemen we in plaats van het maximum het minimum.

OPGAVEN

14 Bepaal het kortste pad van knooppunt A naar knooppunt J in de graaf van opgave 13.

Terugblik

De theorie die in de voorgaande hoofdstukken behandeld is geeft je een beeld hoe je optimaliseringsproblemen op kunt lossen. Van de problemen die we bekeken hebben, hadden we voor het oplossen van het probleem een graaf nodig. Voor de verschillende problemen is de manier gegeven om de graaf behorend bij het probleem te maken. Ook is bij de verschillende problemen gegeven hoe de optimale oplossing gevonden kan worden. De voorbeelden die in de verschillende hoofdstukken gebruikt zijn, zijn niet altijd realistisch geweest. Je zult dan ook begrijpen dat vergelijkbare problemen in werkelijkheid wat ingewikkelder zullen zijn. Dat neemt niet weg dat in het boekje wel duidelijk is gemaakt welke strategiën gebruikt kunnen worden om zo ook de ingewikkeldere problemen op te kunnen lossen. Het zal alleen wat meer tijd kosten om de optimale oplossing te vinden.

Hoofdstuk 5

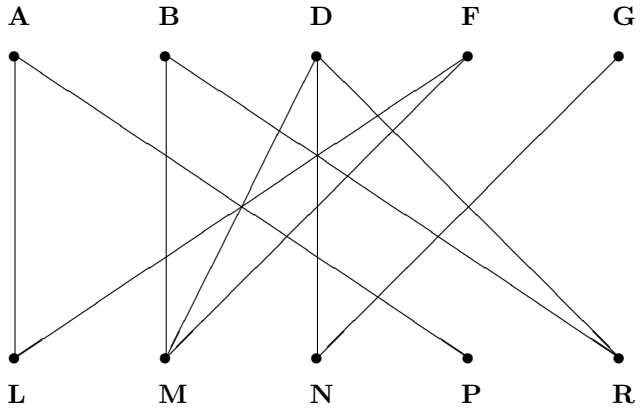
Afsluitende opgaven

5.1 Inleiding

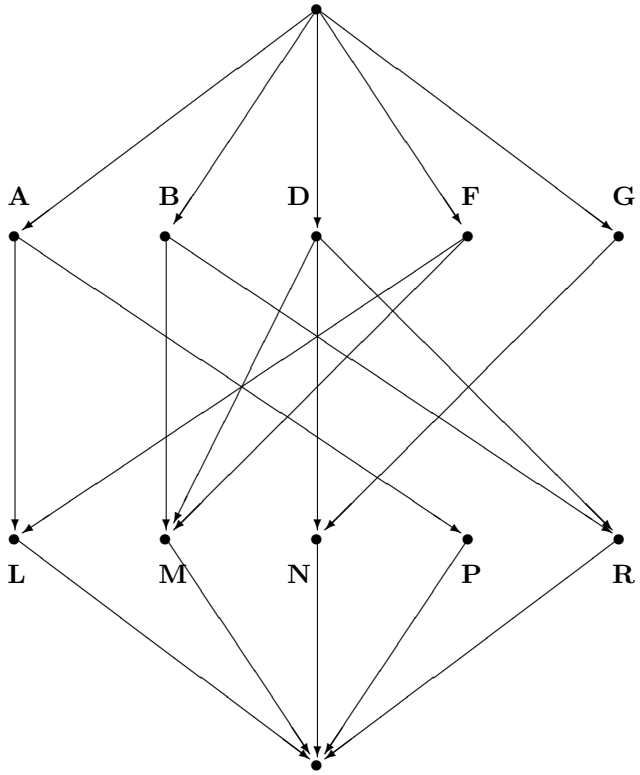
In dit hoofdstuk komen een aantal afsluitende opgaven naar voren die te maken hebben met de stof die in de rest van de hoofdstukken behandeld is. In de eerste afsluitende opgave komt het probleem uit hoofdstuk 3 over koppelingen weer terug, alleen wordt het probleem hier opgelost aan de hand van de stof uit hoofdstuk 2 over netwerkstromen. De tweede afsluitende opgave heeft te maken met de stof die behandeld is in paragraaf 4.3. De derde afsluitende opgave gaan we het probleem uit hoofdstuk 4 op een andere manier oplossen. De eerste en de tweede opgave zijn even moeilijk en zijn redelijk makkelijk te maken als de stof uit hoofdstuk 2 en 3 beheerst worden. De derde opgave is wat lastiger om te maken.

5.2 Koppeling omzetten naar maximale stroom

In hoofdstuk 2 hebben we gezien hoe we in een netwerk een maximale stroom kunnen bepalen. Dit hebben we gedaan aan de hand van het algoritme van Ford en Fulkerson. In hoofdstuk 3 is het koppelingsprobleem aan bod gekomen.
We nemen de graaf van voorbeeld 1 uit hoofdstuk 3 er nog eens bij:



In deze afsluitende opgave gaan we een maximale koppeling in deze graaf proberen te vinden door het probleem op te vatten als een maximale-stroom-probleem. Om een maximale stroom te vinden in een graaf hebben we eerst een begin- en eindknooppunt nodig en we willen een maximale stroom vinden van het begin- naar het eindknooppunt. Aan de bovenstaande graaf voegen we daarom twee knooppunten toe; één knooppunt boven de graaf en de ander eronder. Als we dat gedaan hebben maken we van alle takken in de graaf pijlen en we laten alle pijlen van boven naar beneden lopen. Van het beginknooppunt zetten we pijlen naar alle knooppunten aan de bovenkant en van alle knooppunten aan de onderkant zetten we een pijl naar het eindknooppunt. Verder moeten we de pijlen capaciteiten geven om een maximale stroom te vinden. We geven alle pijlen een capaciteit 1. De graaf waarin we een maximale stroom willen vinden komt er dan als volgt uit te zien (de capaciteiten staan er niet bij, want die is toch voor alle pijlen hetzelfde):



Volgens het algoritme van Ford en Fulkerson moeten we nu een pad zoeken van het begin- naar het eindknooppunt. Van de capaciteiten van de pijlen die in dat pad zitten moeten we het minimum nemen. Omdat in dit geval alle pijlen een capaciteit 1 hebben, zal dit minimum altijd gelijk zijn aan 1. Dus $\Delta w = 1$. Als we vervolgens de nieuwe graaf gaan maken, dan moeten we voor de pijlen die in het pad zitten pijlen in beide richtingen tekenen. De pijl in tegengestelde richting krijgt een waarde $\Delta w = 1$ en van de pijl in de oorspronkelijke richting trekken we Δw af. Maar de waarde van de pijl in de oorspronkelijke richting wordt dan gelijk aan 0 en we laten de pijl dan ook weg.
In deze graaf is dus het volgende het geval; voor het maken van de nieuwe graaf laten we de pijlen die niet in het gekozen pad zitten onveranderd en de pijlen die wel in het pad zitten draaien we om en geven we weer capaciteit 1.

OPGAVEN

1 Pas het algoritme van Ford en Fulkerson toe op de bovenstaande graaf. Wat is de waarde van de maximale stroom?

2 Laat in de laatste graaf die je in opgave 1 hebt gekregen het volgende weg:
- het begin- en eindknooppunt
- alle pijlen die van het beginknooppunt vandaan komen
- alle pijlen die naar het eindknooppunt toe lopen

Wat valt je op aan de pijlen die in tegengestelde richting staan? (hint: vergelijk deze pijlen met de oplossing die we voor dit koppelingsprobleem hebben gevonden in hoofdstuk 3)

5.3 Projectplanning

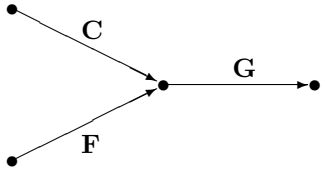
In deze opgave gaan we kijken naar een probleem dat alles te maken heeft met het maken van een goede planning. Bij het maken van een planning heb je natuurlijk te maken met een aantal activiteiten of opdrachten die je uit moet voeren. Maar je kunt die activiteiten of opdrachten niet in een willekeurige volgorde doen. Daar komt een aantal eisen bij kijken. Als we dus een planning willen maken zullen we rekening moeten houden met deze eisen. We nemen het volgende probleem waar we graag een planning voor willen maken.

Voorbeeld: **Huis bouwen**
Je wilt graag een huis bouwen en daarvoor moet je de volgende werkzaamheden uitvoeren (deze lijst van werkzaamheden is natuurlijk sterk vereenvoudigd): Fundering maken, muren metselen, dak vervaardigen, electriciteit aanleggen, ramen plaatsen, schilderen en behangen. Van elk van deze werkzaamheden weet je hoe lang je daar mee bezig bent. Maar zoals je begrijpt zul je wel eerst de fundering aan moeten leggen voordat je kunt beginnen met het metselen van de muren. En het vervaardigen van het dak kan natuurlijk niet voordat je de muren gemetseld hebben. Iedere werkzaamheid heeft dus zijn voorgangers. We zetten alles netjes in een tabel:

Werkzaamheid	Voorganger	Tijdsduur
A. Fundering maken	-	5
B. Metselwerk muren	A	8
C. Dak vervaardigen	B	12
D. Electriciteitswerk	B	5
E. Ramen plaatsen	B	4
F. Behangen	E	6
G. Schilderen	C, F	3

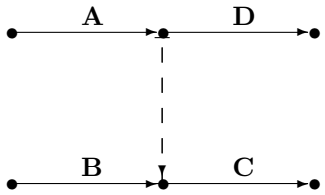
We zijn nu vooral geïnteresseerd in vragen als:
- Wat is de kortste tijdsduur van het totale project?
- Wat is het vroegste en wat is het laatste tijdstip waarop een activiteit kan starten zonder dat daardoor de totale tijdsduur van het project verandert?

Om antwoord te kunnen geven op deze vragen, gaan we ook van dit probleem eerst een graaf maken. Of beter gezegd, we maken een netwerk dat hoort bij het probleem. De pijlen corresponderen met de verschillende werkzaamheden en de getallen bij de pijlen geven de tijdsduur aan. De knooppunten van het netwerk komen overeen met bepaalde stadia waarin het project verkeert. Zo is er een knooppunt dat correspondeert met het begin en een knooppunt dat correspondeert met het einde van het project. Bij de pijlen van het netwerk zetten we de letters van de werkzaamheden.
Werzaamheden zonder voorganger worden gerepresenteerd door een pijl afkomstig van het beginknooppunt, in ons voorbeeld dus A. Werkzaamheden die geen opvolger hebben worden gerepresenteerd door een pijl naar het eindknooppunt, in ons voorbeeld dus D en G. In de tabel zien we dat de werkzaamheden C en F eerst gedaan moeten zijn voordat we aan werkzaamheid G kunnen beginnen. In de graaf zorgen we er dan voor dat de pijlen die horen bij C en F in hetzelfde knooppunt uitkomen. In de graaf ziet dat er dus zo uit:

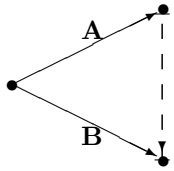


In sommige situaties is het nodig om dummy-activiteiten te gebruiken om de relaties met directe voorgangers te representeren. Een dummy-activiteit vereist geen tijd. Bijvoorbeeld, stel dat activiteit C niet gestart kan worden voordat A en B zijn afgerond en activiteit D pas kan worden gestart zodra A voltooid

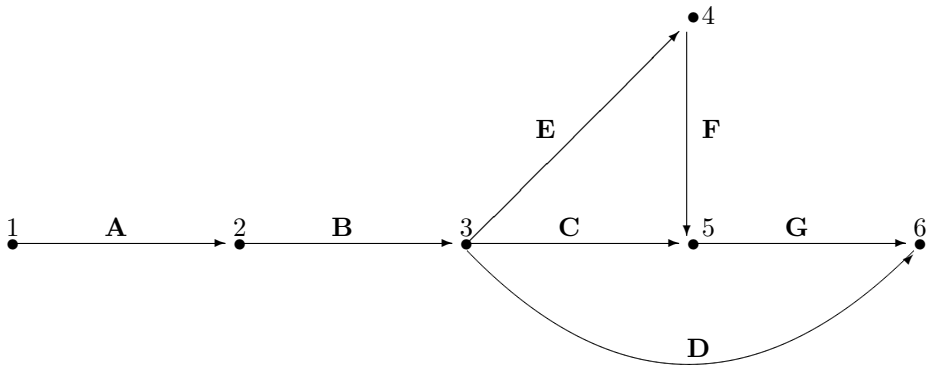
is. Hiervoor hebben we een dummy-activiteit nodig en in de graaf zou dit er dan als volgt uit zien:



Het gebruik van een dummy-activiteit kan ook nodig zijn vanwege de afspraak dat wtee verschillende activiteiten niet hetzelfde begin- en eindpunt mogen hebben. In dat geval ziet de graaf er zo uit (waar activiteit A en B dus tegelijkertijd uitgevoerd kunnen worden):



Als we de graaf gemaakt hebben gaan we de knooppunten nummeren. Dit doen we zodanig dat alle pijlen van een lager naar een hoger nummer lopen. Voor bovenstaand probleem krijgen we de volgende graaf.



Met behulp van het bovenstaande netwerk kunnen we een antwoord geven op de vragen waar we in geïnteresseerd zijn:
- Wat is de kortste tijdsduur van het totale project?
- Wat is het vroegste en wat is het laatste tijdstip waarop een activiteit kan starten zonder dat daardoor de totale tijdsduur van het project verandert?

Om de kortste tijdsduur van het totale project te bepalen gaan we in de graaf op zoek naar het langste pad. Eerder kan het project niet klaar zijn. In paragraaf 4.3 hebben we gezien hoe we het langste pad van een graaf bepalen. In de projectplanning noemen we dit ook wel het *kritieke pad*. Dat betekent dat de taken die in het kritieke pad zitten precies op tijd moeten beginnen, omdat de werkzaamheden anders uitlopen. De taken die niet in het kritieke pad zitten hebben een zekere speling en kunnen als het nodig is iets later beginnen dan gepland.

OPGAVEN

3 Bepaal het kritieke pad van de bovenstaande graaf van knooppunt 1 naar 6.

4 Bekijk het volgende probleem:
Je wilt een nieuw product op de markt brengen. Maar voordat je dat kunt doen moet je eerst een aantal activiteiten uitvoeren. We moeten natuurlijk een ontwerp van het nieuwe product maken en we moeten een marktonderzoek doen om te kijken of er wel vraag is naar je nieuwe product. Verder moeten er grondstoffen besteld worden en moet je die ontvangen van de leverancier. Ook het maken van proefmonsters

is belangrijk. Je moet een verkoopcampagne starten, het productieproces moet gestart worden en uiteindelijk moeten de winkels bevoorraad worden. Natuurlijk kun je je grondstoffen niet ontvangen voordat je ze besteld hebt en kun je niet beginnen met het productieproces als je grondstoffen nog niet binnen zijn. We hebben dus te maken met wat voorgangers. We zetten alle activiteiten, met zijn voorgangers en zijn tijdsduur, netjes in een tabel:

Activiteit	Voorganger	Tijdsduur
A. Ontwerp maken	-	6
B. Marktonderzoek	-	5
C. Grondstof bestellen	A	3
D. Grondstof ontvangen	C	2
E. Proefmonster maken	B, D	3
F. Verkoopcampagne	B	4
G. Productie starten	E	4
H. Winkels bevoorraden	F, G	2

Maak een netwerk bij het probleem en bepaal het kritieke pad.

5.4 Dynamische programmering

Het knapzakprobleem van voorbeeld 1 uit hoofdstuk 4 gaan we oplossen met dynamische programmering. Het idee achter de methode van dynamische programmering is dat we het grote probleem gaan splitsen in een aantal verschillende kleinere deelproblemen. Die kleinere deelproblemen gaan we eerst goed bekijken en voor die kleinere deelproblemen gaan we op zoek naar de optimale oplossing. De optimale oplossingen van die kleinere problemen kunnen we dan weer gebruiken om de optimale oplossing van het grote originele probleem te vinden.
We gaan eens kijken naar ons voorbeeld.
Voorbeeld
We hebben de 5 items met de volgende waardes en groottes:

item	1	2	3	4	5
waarde	16	11	7	8	4
grootte	5	3	2	2	1

Deze items moeten in een koffer met grootte 7.

Om ons probleem met de vijf items op te lossen gaan we de optimale oplossing van het probleem met de eerste vier items gebruiken. Maar ook het kleinere probleem met de eerste vier items kunnen we niet zomaar oplossen. Daarvoor gaan we de optimale oplossing van het probleem met de eerste drie items gebruiken. Maar ook voor dit kleinere deelprobleem geldt dat we hiervan niet zomaar de optimale oplossing kunnen vinden. Een deelprobleem waar we de optimale oplossing wél van kunnen vinden is het probleem met alleen het eerste item. Als we daar de optimale oplossing van weten kunnen we die oplossing gebruiken om een optimale oplossing te vinden van het deelprobleem met de eerste 2 items. Als we er steeds één item bijnemen, komen we uiteindelijk uit bij het originele probleem met alle vijf items.
Bij het maken van de graaf namen we een vast volume voor de koffer en zijn we gaan variëren met het wel of niet meenemen van een item. Bij dynamische programmering gaan we ook weer variëren in het wel of niet meenemen van een item, maar ook het volume van de koffer gaan we variëren. We gaan in ons voorbeeld kijken naar een koffer met een volume tussen de 0 en de 7. Vanaf nu noemen we v het volume van de koffer. In ons voorbeeld kan v dus een getal tussen de 0 en de 7 zijn, omdat de koffer in ons voorbeeld een maximale inhoud heeft van 7 cm^3.

We gaan beginnen met het vinden van een optimale oplossing van het probleem in ons voorbeeld. Zoals we gezien hebben moeten we dus allereerst het deelprobleem met alleen het eerste item gaan oplossen.
We willen nu dus de maximale waarde van de koffer gaan bepalen als we alleen naar item 1 kijken en de koffer een volume v heeft. Deze waarde geven we aan met $F(1, v)$. Verder geven we met $x_1 = 0$ en $x_1 = 1$ aan of we item 1 niet respectievelijk wel meenemen. en deze waarde is dus afhankelijk van het volume v. De waarden van $F(1, v)$ en x_1 zetten we netjes in een tabel:

	$F(1,v)$	x_1
$v = 0$	0	0
$v = 1$	0	0
$v = 2$	0	0
$v = 3$	0	0
$v = 4$	0	0
$v = 5$	16	1
$v = 6$	16	1
$v = 7$	16	1

OPGAVEN

5 Formuleer zelf wat de betekenis is van de regel met v=6.

In de bovenstaande tabel staat nu voor elke mogelijke inhoud v van de koffer de maximale waarde van de inhoud als we alleen kijken naar het eerste item. Deze optimale oplossing gaan we gebruiken om een optimale oplossing te vinden van het deelprobleem met de eerste twee items. Dit doen we als volgt; stel we hebben een inhoud v van de koffer. We willen de maximale waarde van de koffer met inhoud v vinden als we alleen kijken naar eerste twee items. Deze maximale waarde geven we aan met $F(2, v)$. Met x_2 geven we weer op dezelfde manier als bij x_1 aan of we item 2 wél of niet meenemen.

Stel we hebben een koffer met volume v. Dan hebben we twee mogelijkheden:

- kies $x_2 = 0$ (item 2 niet mee).
  Dan is de waarde van de koffer gelijk aan $F(1, v)$.
- kies $x_2 = 1$ (item 2 wel mee).
  Dan is de waarde van de koffer 'waarde item 2'+$F(1, v - 3)$.

Bij de tweede mogelijkheid moeten we er wel op letten dat we er voor kunnen kiezen om het tweede item mee te nemen. Als je x_2 gelijk aan 1 wilt kiezen, moet dat wel betekenen dat item 2 ook in de koffer past. Past dit niet, dan blijft alleen mogelijkheid 1 over.

OPGAVEN

6 Je ziet bij de tweede mogelijke keuze het volgende staan: *'waarde item 2'*+$F(1, v - 3)$.
Hoe komen ze aan $v - 3$ in $F(1, v - 3)$?

Nu willen we van deze twee mogelijke waarden het maximum hebben. Het maken van een keuze komt dus neer op het volgende:

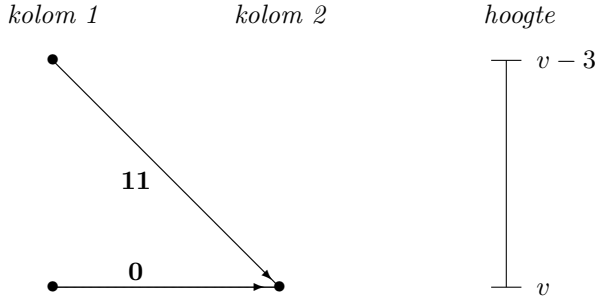
$$F(2, v) = max\Big\{F(1, v); \text{'waarde item 2'} + F(1, v - 3)\Big\},$$

waarbij

$$x_2 = \begin{cases} 0 & \text{als } F(1, v) \text{ het maximum geeft;} \\ 1 & \text{als 'waarde item 2'} + F(1, v - 3) \text{ het maximum geeft.} \end{cases}$$

We kunnen nu met de bovenstaande formule voor alle mogelijke waarden van v de maximale waarde van de koffer bepalen als we alleen kijken naar de eerste twee items. Ook kunnen we met de bovenstaande formule bepalen of we item 2 wél of niet meenemen. Maar we willen natuurlijk ook weten of item 1 mee gaat in de koffer. Dat kunnen we zien aan de waarden $F(1, v)$ of $F(1, v - 3)$, afhankelijk van wat het maximum is in de formule. Is het maximum gelijk aan $F(1, v)$, dan moeten we in de eerste tabel kijken wat de waarde van x_1 is bij $F(1, v)$. Is het maximum gelijk aan 'waarde item 2' + $F(1, v - 3)$, dan kijken we in de eerste tabel wat de waarde is van x_1 bij $F(1, v - 3)$.

We pakken ook de graaf er eens bij en gaan eens kijken wat we in de graaf doen als we dynamische programmering toepassen. Bij elk deelprobleem dat we bekijken hoort ook een deelgraaf. We zijn nu aan het kijken naar het deelprobleem met alleen de eerste twee items. De deelgraaf die daar bij hoort is de volgende; neem alle knooppunten uit de 'nulde', eerste en tweede kolom (het knooppunt uit de 'nulde' kolom is het beginknooppunt). Alle takken die bij de deelgraaf horen zijn alle de takken die tussen de genoemde knooppunten lopen. Bij dynamische programmering kijken we naar een koffer met volume v. In de graaf betekent dit, dat we gaan kijken naar een knooppunt uit de tweede kolom (want we kijken alleen naar de eerste twee items) die op een hoogte v zit. Zie het plaatje hieronder:



Het kan zijn dat we helemaal geen knooppunt hebben op een hoogte v. Dan betekent dat dat we geen keuze kunnen maken uit de eerste twee items zó dat de inhoud van de koffer precies v is. Ook kan het zijn dat één van de twee of beide pijlen niet bestaan. Dat heeft dan te maken met het bestaan van de knooppunten in de eerste kolom.
Ook hebben we bij het maken van de graaf de knooppunten in de tweede kolom een waarde gegeven. En dat voor elke hoogte v van een knooppunt. De waarde van het knooppunt op hoogte v en kolom 2 komt bij dynamische programmering overeen met de waarde van $F(2, v)$. Het bepalen van de waarde van het knooppunt komt bij dynamische programmering overeen met het bepalen van het maximum in de formule

$$F(2, v) = max\Big\{F(1, v); \text{'waarde item 2'} + F(1, v - 3)\Big\}.$$

We hebben nu de link gezien tussen het maken van de graaf en het toepassen van dynamische programmering. Voordat we verder gaan met het deelprobleem met de eerste drie items, zetten we voor alle mogelijke v de waarden $F(2, v)$ eerst nog even netjes in een tabel. Ook we waarden van x_1 en x_2 zetten we erbij.

	$F(2, v)$	x_2	x_1
$v = 0$	0	0	0
$v = 1$	0	0	0
$v = 2$	0	0	0
$v = 3$	11	1	0
$v = 4$	11	1	0
$v = 5$	16	0	1
$v = 6$	16	0	1
$v = 7$	16	0	1

Nu hebben we de optimale oplossing van het deelprobleem met de eerste 2 items en kunnen we de optimale oplossing van het deelprobleem met de eerste 3 items gaan bepalen. En met die oplossing kunnen we de optimale oplossing vinden van het probleem met de eerste 4 items en vervolgens kunnen we het originele probleem oplossen.
Als we de grootte van item i nu g_i noemen en de waarde van item i nu w_i, dan kunnen we een algemene formule geven voor het bepalen van de maximale waarde van een koffer met volume v als we kijken naar het deelprobleem met de eerste i items. i zal in ons voorbeeld dus variëren van 1 t/m 5.
De algemene formule die we krijgen is als volgt:

$$F(i, v) = max\Big\{F(i - 1, v); w_i + F(i, v - g_i)\Big\}.$$

Door deze formule toe te passen vinden voor het originele probleem de volgende tabel:

	$F(5, v)$	x_5	x_4	x_3	x_2	x_1
$v = 0$	0	0	0	0	0	0
$v = 1$	4	1	0	0	0	0
$v = 2$	8	0	1	0	0	0
$v = 3$	12	1	1	0	0	0
$v = 4$	15	0	1	1	0	0
$v = 5$	19	0	1	0	1	0
$v = 6$	23	1	1	0	1	0
$v = 7$	26	0	1	1	1	0

OPGAVEN

7 Om de bovenste tabel te kunnen maken heb je de tabellen van $F(3, v)$ en $F(4, v)$ nodig. Maak deze tabellen en controleer of je hiermee ook de bovenstaande tabel krijgt.

Uit de tabel van het originele probleem kunnen we nu de maximale waarde van de koffer halen als we kijken naar de 5 items die we mee willen nemen. De koffer heeft namelijk een inhoud van 7 cm^3, dus als we kijken in de tabel bij $v = 7$ vinden we dat de maximale waarde van de koffer gelijk is aan 26. Ook vinden we dat we uiteindelijk de items 2, 3 en 4 mee op reis nemen om een totale waarde van 26 in de koffer te krijgen.

We zien nu dat dit een betere oplossing van het probleem geeft dan die we gezien hebben bij opgave 3 en 4 uit hoofdstuk 4.

OPGAVEN

8 Aan het begin van het hoofdstuk hebben we het knapzakprobleem gezien met de volgende gegevens:

item	1	2	3	4	5	6	7	8	9	10
waarde	16	22	12	8	20	6	10	25	14	13
grootte	5	7	4	3	6	3	3	9	4	4

De koffer heeft in dit geval een inhoud van 30 cm^3. Pas de methode van dynamische programmering toe op dit knapzakprobleem. Je zult zien dat dit een ander en beter antwoord oplevert dan de antwoorden die je hebt gevonden bij opgave 1 en 2 uit hoofdstuk 4.

We hebben nu gekeken naar het probleem dat we een koffer met bepaalde inhoud hebben en dat we de waarde van de koffer willen optimaliseren. We kunnen ook kijken naar een alternatief probleem waarin we een bepaalde waarde mee willen hebben en dat we dit met een zo klein mogelijk volume willen doen. Wél moet het volume passen in de koffer die we hebben. In het voorbeeld met de vijf items moeten het dus in een koffer van volume 7 passen.
Voor het oorspronkelijke probleem hebben we de formule $F(i, v)$ gemaakt. Ook voor dit alternatieve probleem kunnen we een formule maken zoals we bij het oorspronkelijke probleem ook hebben gedaan.

OPGAVEN

9 Geef de formule die hoort bij het alternatieve probleem. Noem de formule $G(i, W)$, waarbij W de totale waarde van de koffer is.

10 Bekijk nogmaals het voorbeeld. Welke items moet ik meenemen om een waarde van 20 met een zo klein mogelijk volume te krijgen? En wat is dat volume dan?

Bijlage

A Korte Uitwerkingen en Antwoorden

Hoofdstuk 2: Netwerkstromen

4 De maximale stroom heeft een waarde van 6. Er zijn meerdere mogelijkheden om de stroom van het begin- naar het eindknooppunt te sturen. Eén daarvan is de volgende:

De stroom gaat over de volgende verbindingen:

$S-A:2$	$A-D:0$	$B-C:1$	$D-T:0$
$S-B:2$	$A-T:2$	$B-T:1$	
$S-C:2$	$A-B:0$	$C-T:3$	

5 De maximale stroom heeft een waarde van 12, dus er kunnen 12 mailtjes per minuut van X naar Y verstuurd worden.

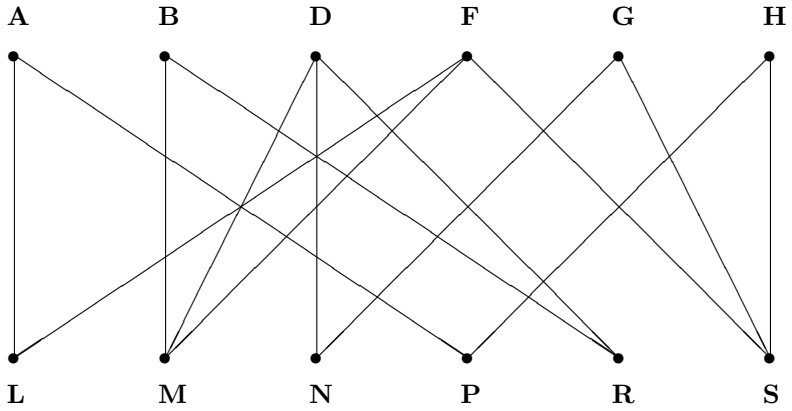
De stroom gaat over de volgende verbindingen:

$X-A:8$	$A-D:3$	$D-B:0$
$X-C:4$	$C-B:2$	$B-Y:7$
$A-B:5$	$C-D:2$	$D-Y:5$

6 De minimale snede zijn de knooppunten S, B en C. De pijlen die van deze punten naar andere punten lopen zijn de pijl van S naar A, de pijl van B naar T en de pijl van C naar T. De capaciteit van de snede is dus gelijk aan $2+1+3=6$.

Hoofdstuk 3: Koppelingen

4 De graaf komt er als volgt uit te zien:



5 Een volmaakte koppeling is ook een maximale koppeling.
Een maximale koppeling hoeft geen volmaakte koppeling te zijn.

6 In de maximale koppeling krijgen we de volgende koppels:
$1 \longleftrightarrow 10$, $2 \longleftrightarrow 13$, $3 \longleftrightarrow 9$, $4 \longleftrightarrow 8$, $5 \longleftrightarrow 14$, $6 \longleftrightarrow 11$, $7 \longleftrightarrow 12$.
Deze koppeling is uniek, er is dus geen andere koppeling mogelijk.

7 In de maximale koppeling krijgen we de volgende koppels:
$1 \longleftrightarrow 13.30$, $2 \longleftrightarrow 14.00$, $3 \longleftrightarrow 15.30$, $4 \longleftrightarrow 16.00$, $5 \longleftrightarrow 14.30$, $6 \longleftrightarrow 13.00$, $7 \longleftrightarrow 15.00$.
Deze koppeling is uniek, er is dus geen andere koppeling mogelijk.

8 Begin met de volgende koppels:
$A \longleftrightarrow 1$, $B \longleftrightarrow 5$, $C \longleftrightarrow 2$, $D \longleftrightarrow 6$, $E \longleftrightarrow 4$, $F \longleftrightarrow 8$ en $I \longleftrightarrow 3$.

Neem vervolgens het volgende pad: $G \rightarrow 1 \rightarrow A \rightarrow 8 \rightarrow F \rightarrow 2 \rightarrow C \rightarrow 9$. We krijgen dan de volgende nieuwe koppels:
$A \longleftrightarrow 8$, $B \longleftrightarrow 5$, $C \longleftrightarrow 9$, $D \longleftrightarrow 6$, $E \longleftrightarrow 4$, $F \longleftrightarrow 2$, $G \longleftrightarrow 1$ en $I \longleftrightarrow 3$.

Er is nu geen pad meer te vinden in de graaf die hoort bij de bovenstaande koppels en dus hebben we een maximale koppeling gevonden. We zien dus dat we opdracht H en werknemer 7 niet kunnen koppelen.

Dit is meer één mogelijkheid van koppels die gemaakt kunnen worden. Er zijn hier meerdere antwoorden mogelijk. Het zal alleen wel altijd zo zijn dat niet alle opdrachten uitgevoerd kunnen worden.

10 We krijgen de volgende vrouwenoptimale stabiele koppeling:

Naam jongen	Gekoppeld aan
Laura	Bart
Mariska	Dirk
Natascha	Frank
Pauline	Gerard
Romy	Arnold

Hoofdstuk 4: Stapsgewijze Optimalisatie

4 We moeten eerst van alle items de waarde per cm^3 bepalen, zie de onderstaand tabel.

item	1	2	3	4	5	6	7	8	9	10
waarde per cm^3	3.2	3.14	3	2.67	3.33	2	3.33	2.78	3.5	3.25
grootte	5	7	4	3	6	3	3	9	4	4

We nemen nu de items 9, 5, 7, 10, 1 en 2 mee (in volgorde van hoogste waarde per cm^3). Deze items hebben bij elkaar een waarde van 95.

5 Neem item 5,4 en 2 mee met totale waarde van 23.

6 Neem item 1 en 4 mee met totale waarde van 24.

7 Neem item 5, 4 en 2 mee met totale waarde van 23.

11 In de laatst verkregen graaf zien we dat de maximale waarde van de koffer 26 kan worden. Om te zien welke items we daarvoor mee moeten nemen, moeten we in de graaf 'teruglopen' naar het beginknooppunt. Als we dat doen krijgen we dat we de items 4, 3 en 2 mee moeten nemen om een waarde van 26 te krijgen.

12 De maximale waarde van de koffer is 38 en daarvoor nemen we de items 2, 5 en 6 mee of de items 2 en 3 (dit levert allebei een waarde van 38 op!).

13 Het langste pad is: $A \rightarrow C \rightarrow F \rightarrow I \rightarrow J$ en dit pad heeft een lengte van 21.

14 Het kortste pad is: $A \rightarrow C \rightarrow E \rightarrow H \rightarrow J$ en dit pad heeft een lengte van 17.

Index