Radboud Universiteit Nijmegen

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

Toewijzingsprobleem

Bachelorscriptie

Auteur: Veronique Rademaekers (s4155718)

Begeleiders: Dr. W. Bosma en dr. H. Don

juli 2018

Voorwoord

Voor u ligt eindelijk het eindresultaat van mijn bachelorscriptie. Na een lange periode van uitstellen en weer doorgaan ben ik trots dat het me toch gelukt is om deze scriptie te schrijven en aan u kan presenteren.

Grote dank aan mijn begeleiders en hun eeuwige geduld, zonder wie ik dit niet tot een goed einde had kunnen brengen. Het was niet makkelijk, het was niet altijd leuk en heb ik zeker wel eens getwijfeld of ik er goed aan deed wiskunde te gaan studeren aan een universiteit. Toch kijk ik terug op een ongelofelijk mooie tijd als student en heb ik het vertrouwen dat ik daar thuishoorde. De laatste loodjes wegen nou eenmaal het zwaarst en dat heb ik mogen ervaren.

Ik u veel leesplezier toe.

Veronique Rademaekers

Inleiding

Studenten van de Radboud Universiteit kiezen steeds vaker voor een buitenlandervaring in de vorm van een stage of het volgen van een aantal vakken aan een universiteit in het buitenland. Om al hun plannen te realiseren moet er van alles worden geregeld. Het International Office verstrekt alle informatie over beursprogrammaś en het aanmelden voor een internationaal studie- of stageverblijf. Studenten die kiezen voor een stage in het buitenland regelen vrijwel altijd zelf de plek waar ze terecht komen. Dit in tegenstelling tot de studenten die willen studeren in het buitenland. Deze studenten geven bij hun aanmelding aan waar ze graag zouden studeren. Dit doen ze door middel van het opgeven van drie voorkeursplekken aan bepaalde universiteiten. Het International Office is verantwoordelijk voor het indelen van de studenten aan de beschikbare universiteiten. Hierbij proberen ze zo goed mogelijk rekening te houden met de voorkeuren van alle studenten. Het streven is dat elke student geplaatst wordt op een universiteit die binnen zijn of haar keuze valt.

In de loop der jaren is het studeren aan een universiteit in het buitenland steeds populairder geworden, waardoor het maken van een goede indeling steeds complexer wordt en het meer tijd in beslag gaat nemen. De indeling die elk half jaar gemaakt moet worden wordt momenteel handmatig gemaakt door een medewerker van het International Office. Hij of zij legt een lijst met studenten en een lijst met beschikbare universiteiten naast elkaar en begint boven aan de lijst met het koppelen van studenten aan een universiteit van hun keuze. Op het moment dat een student niet meer gekoppeld kan worden aan één van zijn of haar keuzes, worden er wat toewijzingen verwisseld of verwijderd met als doel de indeling te verbeteren. Het kan op deze manier erg lang duren voordat alle studenten gekoppeld zijn aan een universiteit. Daarnaast is het niet bekend of de verkregen indeling daadwerkelijk de beste indeling is. Om het International Office te helpen met deze procedure gaan we in deze scriptie ontdekken of dit proces geautomatiseerd kan worden. Ook zullen we bekijken welk programma er nodig is om de theorie om te zetten in de praktijk, zodat er daadwerkelijk gebruik gemaakt kan worden van de oplossing op het moment dat die gevonden is en het probleem van het International Office wordt opgelost.

Om dit te bereiken bekijken we het probleem van het International Office als een algemeen toewijzingsprobleem. We beginnen met het uitleggen van een aantal basisbegrippen uit de grafentheorie. Het toewijzingsprobleem kunnen we namelijk opvatten als een bipartiete graaf waarbij de studenten de ene partitie zijn en de universiteiten de andere.

We gaan in deze scriptie twee verschillende manieren bekijken voor het vinden van een toewijzing tussen twee partities. De eerste manier, die aan bod komt in hoofdstuk 2, is het oplossen van het *maximum cardinality matching problem*. Bij dit probleem gaan we

op zoek naar een zo groot mogelijke toewijzing, waarbij groot staat voor een zo hoog mogelijke kardinaliteit van de toewijzing. Nadat we een zo groot mogelijke toewijzing gevonden hebben, weten we zeker dat alle andere toewijzingen die te vinden zijn hoogstens zo groot zijn als de toewijzing die we reeds gevonden hebben. De stelling van König is erg belangrijk bij het oplossen van het maximale kardinaliteit toewijzingsprobleem.

De tweede manier is het *minimum weight perfect matching problem* en wordt behandeld in hoofdstuk 3. Hier zijn we niet alleen op zoek naar een toewijzing die zo groot mogelijk is, maar is het bovendien van belang dat alle punten in de partities worden toegewezen. Dat wil zeggen dat de kardinaliteit van de toewijzing gelijk moet zijn aan het aantal elementen uit de afzonderlijke partities. Daarnaast zijn we pas tevreden wanneer het totale gewicht van de toewijzing zo klein mogelijk is. De wiskunde die hierbij aan bod komt is geheeltallig en lineair programmeren, waarbij relaxatie en primal-dual algoritme begrippen zijn die een grote rol zullen spelen bij het oplossen van een toewijzingsprobleem in dit hoofdstuk.

Nadat in hoofdstuk 3 op een wiskundige manier wordt getoond dat er een oplossing bestaat voor een algemeen toewijzingsprobleem, gaan we in hoofdstuk 4 bekijken hoe de theorie zich vorm geeft in het probleem van het International Office. Is het met alle voorwaarden die ze daar stellen nog steeds mogelijk om ook dit probleem om te lossen? Waar moeten ze rekening mee houden? Welke voorwaarden kunnen ze wel en niet toevoegen aan het probleem? We besluiten deze scriptie met een aanbeveling voor het International Office wat betreft het automatiseren van hun toewijzingsprobleem van studenten aan universiteiten. Er wordt antwoord gegeven op de vraag of het mogelijk is het toewijzingsprobleem te automatiseren.

Hoofdstuk 1

Grafentheorie

1.1 Basisdefinities

In dit hoofdstuk worden eerst de nodige basisbegrippen uit de grafentheorie gedefiniëerd, op deze manier ontstaat er duidelijkheid over definities en is de scriptie beter leesbaar voor iedereen die zich niet verdiept heeft in de stof. De begrippen die hieronder genoemd worden zijn belangrijk bij het in kaart brengen van een toewijzingsprobleem.

Definitie 1.1. Een *graaf* G is een paar (V, E) waarbij V een eindige verzameling is en E een verzameling van paren (u, v) uit V. We zullen spreken van *punten* als we het hebben over de elementen van V en over *zijden* als het gaat om elementen uit E.

Definitie 1.2. In een graaf is een *wandeling* te maken tussen twee punten a en b. Dat wil zeggen dat er een serie van verbonden punten bestaat, waarvan punt a het beginpunt en punt b het eindpunt van de serie is.

Definitie 1.3. Zo een wandeling noemen we een *pad* tussen een punt a en een punt b wanneer elk punt hoogstens één keer in de serie van punten voorkomt.

Voorbeeld 1.1. Als $a - c - d - g - a - c - b$ een wandeling is van een punt a naar een punt b, dan is $a - c - b$ het bijbehorende pad is, waarbij het punt a het beginpunt is van het pad en het punt b het eindpunt is van het pad. We kunnen hieruit concluderen dat een pad een efficiënte wandeling is van een punt a naar een punt b.

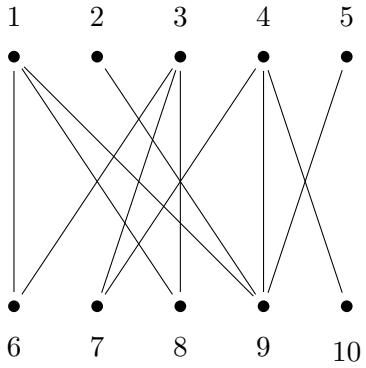
Opmerking. Een pad van een punt a naar een punt b heeft minstens twee punten vanwege het feit dat voor elk punt a en elk punt b geldt dat $a \neq b$.

Definitie 1.4. We noemen een graaf $G = (V, E)$ *bipartiet* als V op te splitsen is in twee niet-lege disjuncte verzamelingen A en B op een manier dat elke zijde in E een eindpunt heeft in A en een eindpunt heeft in B. M.a.w. een element e van E is te schrijven als (a, b) waarbij a in A en b in B. Oftewel:

$$E \subseteq \{(a, b) \mid a \in A \text{ en } b \in B\}$$

De verzamelingen A en B worden ook wel de delen van de graaf genoemd. Wanneer dit het geval is, is er sprake van een *bipartitie* tussen A en B.

Voorbeeld 1.2. Zij $G = (V, E)$ een bipartiete graaf op 10 punten. Met $\{1, 2, 3, 4, 5\} \in A$ en $\{6, 7, 8, 9, 10\} \in B$.



Figuur 1.1: Bipartiete graaf op 10 punten

Als we te maken hebben met een bipartiete graaf, dan kunnen we punten uit A en punten uit B waartussen zich een zijde bevindt aan elkaar toewijzen.

Definitie 1.5. Een *matching* M is het resultaat van het toewijzen van punten uit A aan punten uit B en bevat zijde uit E, $M \subseteq E$, zodanig dat geen enkele zijde uit M gemeenschappelijke eindpunten heeft met een andere zijde uit M. I.e. voor alle $(a, b), (c, d) \in M$ waarvoor geldt $a = c$ (of $b = d$) dan $(a, b) = (c, d)$.

Opmerking. In een matching, zoals hierboven beschreven, kunnen we geen wandeling maken. Een pad is een bijzondere wandeling, dus vanzelfsprekend bevat de matching ook geen paden.

Definitie 1.6. Een pad waarbij de zijden afwisselend wel en niet in de matching zitten wordt een *alternerend pad* genoemd.

Definitie 1.7. Wanneer een zijde $e = (a, b)$ uit E in een matching M zit, zijn a en b *gematcht* en noemen we a en b *gematchte* punten. Punten uit V die niet overdekt worden door een zijde uit M zijn niet gematcht. Deze punten noemen we *exposed*.

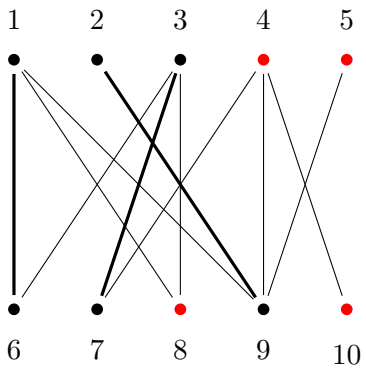
Definitie 1.8. Een *augmenting pad* is een speciaal geval van een alternerend pad waarbij het beginpunt en het eindpunt beiden exposed punten zijn.

Om een aantal begrippen te verduidelijken illustreren we ze in het volgende voorbeeld.

Voorbeeld 1.3. We bekijken opnieuw de bipartiete graaf in figuur 1.1 van voorbeeld 1.2 en nemen de volgend matching M:

$$M = \{(1, 6), (2, 9), (3, 7)\}$$

De dikgedrukte zijden in figuur 1.2 geeft deze matching weer en we zien dat de punten 4, 5, 8 en 10 niet overdekt worden door een zijde uit de matching en dus exposed punten zijn. $1 - 6 - 3 - 7 - 4$ is een alternerend pad waarbij zijden $(1, 6), (3, 7) \in M$ en $(3, 6), (4, 7) \notin M$. Dit alternerend pad is geen augmenting pad, want het punt 1 behoort niet tot de exposed punten. Voegen we het exposed punt 8 toe aan het begin van het pad, dan krijgen we het augmenting pad $8 - 1 - 6 - 3 - 7 - 4$, waarbij wederom $(1, 6), (3, 7) \in M$ en naast $(3, 6), (4, 7) \notin M$ nu ook $(1, 8) \notin M$.



Figuur 1.2: Bipartiete graaf

Definitie 1.9. Als elk punt uit verzameling A gematcht wordt aan een punt uit verzameling B en elk punt uit verzameling B gematcht wordt aan een punt uit verzameling A dan heet de matching *perfect*.

Definitie 1.10. Een matching waarvoor geldt dat elk punt uit A gematcht wordt aan een punt uit B, maar het niet noodzakelijk is dat elk punt uit B gematcht wordt aan een punt uit A noemen we een *semi-perfecte matching*.

Opmerking. Een semi-perfecte matching is perfect als $|A| = |B|$.

Definitie 1.11. Een graaf $G = (V \cup E)$ is een gerichte graaf als de zijden van E gericht zijn van een punt van V naar een ander punt van V.

Hoofdstuk 2

Maximum cardinality matching problem

Als we kijken naar de bipartiete graaf uit voorbeeld 1.2 (figuur 1.1), dan zien we dat er niet voor elke bipartiete graaf een perfecte matching bestaat, maar het is wel altijd mogelijk om zo veel mogelijk zijden in de matching te stoppen.

Definitie 2.1. Een *maximale matching* is een matching waarvoor de kardinaliteit van een matching de maximaal te behalen kardinaliteit voor een matching van de betreffende bipartiete graaf is. Voor elke matching $M' \neq M$ geldt dat $|M'| \leqslant |M|$.

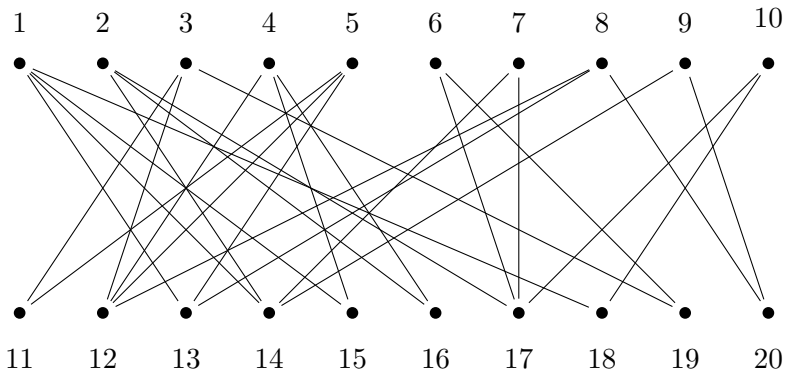
In figuur 1.1 wordt een bipartiete graaf weergegeven door 10 punten (5 per partitie) en 11 zijden die de punten van de partities verbinden. Het vergt niet veel moeite om te zien welke punten van de verzamelingen A en B gematcht moeten worden om een zo groot mogelijke matching te krijgen.

Voorbeeld 2.1. Zie figuur 1.1. In deze bipartiete graaf vormen de zijden $(1, 6), (2, 9), (3, 7)$ de matching M. Met één oogopslag zien we dat dit geen maximale matching is. Punt 4 is namelijk ook nog te matchen aan punt 10, zonder dat dit problemen geeft. De kardinaliteit van de nieuwe matching is dan 4. Groter dan 4 is niet mogelijk, want er zijn twee punten uit A die alleen gematcht kunnen worden met punt 9, wat onmogelijk is. De verzameling $M = \{(1, 6), (2, 9), (3, 7), (4, 10)\}$ is dus een maximale matching met kardinaliteit 4.

We zouden kunnen concluderen dat deze weergave van een bipartiete graaf overzichtelijk is en dat we in korte tijd een maximale matching kunnen vinden. Bij een graaf met relatief weinig punten is dit inderdaad vaak het geval, maar bij het oplopen van de kardinaliteit is het steeds lastiger te zien welke punten verbonden moeten worden om een maximale matching te krijgen en weten we bovendien niet eens of we een maximale matching hebben gevonden. Daarnaast zien we in figuur 2.1 dat een graaf met veel punten kan zorgen voor een onoverzichtelijke weergave, waardoor het niet eens duidelijk is welke punten allemaal verbonden zijn.

Om de weergave overzichtelijk te houden bij een bipartitie graaf van hoge kardinaliteit, gaan we de graaf omzetten naar een $n \times m$-matrix, waarbij n en m de kardinaliteiten van partitie A respectievelijk B zijn. Deze matrix wordt als volgt gedefiniëerd:

$$g_{ij} = \begin{cases} 1 & \text{als } (i, j) \in E; \\ 0 & \text{als } (i, j) \notin E. \end{cases}$$



Figuur 2.1: Bipartiete graaf op 20 punten

Voorbeeld 2.2. De verzameling van zijden die hoort bij bovenstaande graaf is de verzameling E.

$$\begin{aligned} E = \{&(1,13),(1,14),(1,15),(1/18),(2,14),(2,16),(2,17),(3,11),(3,12),\\ &(3,19),(4,12),(4,15),(4,16),(5,11),(5,12),(5,13),(6,17),(6,19),\\ &(7,14),(7,17),(8,12),(8,13),(8,20),(9,14),(9,20),(10,17),(10,18)\} \end{aligned}$$

Noteren we deze verzameling in een 10×10-matrix zoals zojuist gedefiniëerd wordt, dan levert dat de volgende matrix op:

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Figuur 2.2: 10×10 - matrix waarbij de rijen horen bij de punten 1 t/m 10 en de kolommen bij de punten 11 t/m 20.

Nu we een overzichtelijke en werkbare weergave van een graaf hebben gedefiniëerd zullen we de zoektocht naar een maximale matching voortzetten.

2.1 Stelling van König

Definitie 2.2. Een *puntenoverdekking* C van een bipartiete graaf G is een verzameling van punten uit V die op zijn minst één eindpunt bevat van alle zijden uit E. Een puntenoverdekking is minimaal op het moment dat alle mogelijke puntenoverdekkingen meer, of evenveel punten bevatten. I.e. C is minimaal als voor elke $C' \neq C$ geldt dat $C' \geq C$.

Stelling 2.1. (Stelling van König) Zij G een bipartiete graaf, M de maximale matching en C de minimale puntenoverdekking. Dan geldt voor elke bipartiete graaf G:

$$|M| = |C|$$

2.2 Fase 1: Augmenting pad vinden

Om een maximale matching te vinden met behulp van een algoritme gaan we in twee fases naar een gewenst eindresultaat werken. Elke fase bestaat uit de nodige stappen die moeten worden doorlopen. De eerste fase, het vinden van een augmenting pad, zal in deze sectie aan bod komen. Voordat we aan fase 1 beginnen, moeten we eerst nog een aantal matrices definiëren. We gaan gebruik maken van de *keuzematrix* K en de *toewijzingsmatrix* T ten opzichte van een graaf G, waarbij de rijen staan voor de punten van A en de kolommen voor de punten van B. Net zoals in figuur 2.2 hoort elke plek in de matrix bij een combinatie van een punt van A en een punt van B. Staat er op de plek een 1, dan staat de combinatie van de twee punten voor een zijden. Staat er op de plek een 0, dan is er geen verbinding (te maken) tussen de twee punten. Ter verduidelijking definiëren we de *keuzematrix* K zodanig dat:

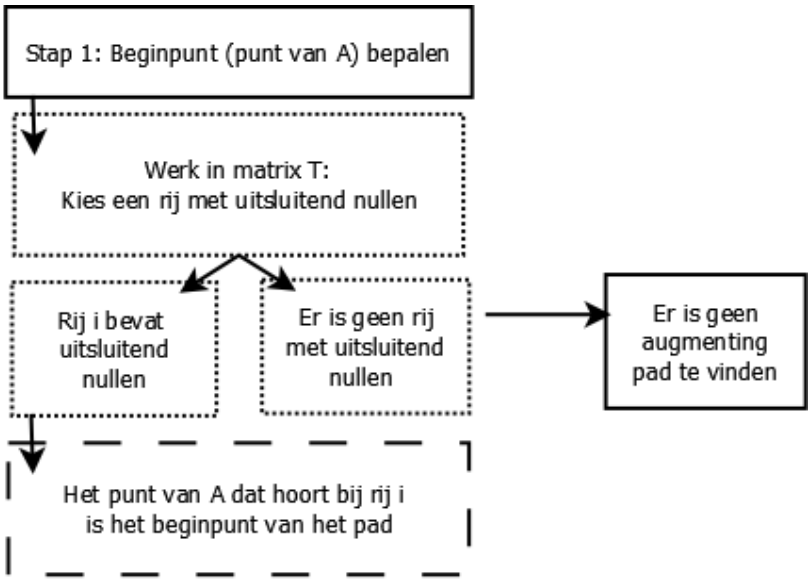
$$k_{ij} = \begin{cases} 1 & \text{als } i \text{ en } j \text{ gematcht kunnen worden;} \\ 0 & \text{als } i \text{ en } j \text{ niet gematcht kunnen worden.} \end{cases}$$

en de toewijzingsmatrix T zodanig dat:

$$t_{ij} = \begin{cases} 1 & \text{als de zijde } (i,j) \text{ in de matching zit;} \\ 0 & \text{als de zijde } (i,j) \text{ niet in de matching zit.} \end{cases}$$

Het feit dat twee punten gematcht kunnen worden en er dus op die plek in matrix K een 1 komt te staan betekent dat er in de bipartiete graaf een verbinding is tussen die twee punten. Een 1 in matrix T staat voor een zijde in de matching. In tegenstelling tot de toewijzingsmatrix kan er in de keuzematrix per rij of kolom meer dan één 1 staan. In de toewijzingsmatrix is dit dus niet het geval, daar geldt dat er in elke kolom en rij van matrix T hoogstens één 1 kan staan. Een rij of kolom kan daar bovendien uitsluitend uit nullen bestaan.

We starten de zoektocht naar een augmenting pad met een willekeurige beginmatching M. Daarna moeten we het eerste punt van een augmenting pad bepalen. Dit punt zit in A en moet exposed zijn, anders voldoet het pad niet aan de juiste eisen (definitie 1.8). Als een punt van A exposed is, dan betekent dit dat het punt niet gematcht is aan een punt uit B en er in de toewijzingsmatrix in de bijbehorende rij in geen enkele kolom een 1 staat. We zijn dus op zoek naar een rij met uitsluitend nullen. Het kan natuurlijk voorkomen dat geen enkele rij uitsluitend nullen bevat. Dan stopt het algoritme meteen, want dan is er geen augmenting pad te vinden. Het is niet uitgesloten dat er meer dan 1 rij bestaat met uitsluitend nullen, dus zal er een willekeurige keuze moeten worden gemaakt tussen deze rijen. Is de keuze gemaakt dan hebben we het beginpunt voor het augmenting pad gevonden, namelijk het punt van A wat hoort bij de gekozen rij.

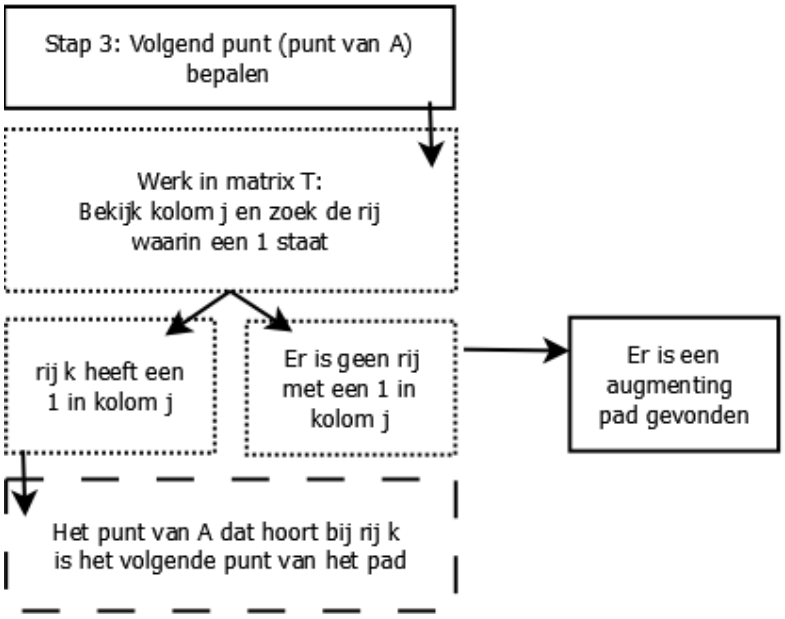


Figuur 2.3: Bepalen van het beginpunt

Een pad bestaat per definitie uit meer dan 1 punt, dus moet er sowieso nog een punt worden gekozen. Dit punt moet een punt uit B zijn en moet gematcht kunnen worden aan het zojuist gevonden punt van A. Dat wil zeggen dat we een kolom moeten vinden in keuzematrix K waarbij in de rij die we zojuist gekozen hebben een 1 staat. Ook nu is het mogelijk dat er meer dan één kolom voldoet aan deze voorwaarde. Het kiezen van één van deze kolommen levert het volgende punt van een augmenting pad op.

Figuur 2.4: Bepalen van het volgende punt

Nu het pad twee punten bevat, moeten we nagaan of het tweede punt een exposed punt is. Dit kunnen we nagaan in matrix T. Zoals al eerder genoemd hoort een punt van B altijd bij een kolom van de matrices. We bekijken daarom betreffende kolom. Staat er geen 1 in deze kolom, dan is het bijbehorende punt exposed en is er een augmenting pad gevonden, want het beginpunt en het eindpunt zijn beiden exposed. Als dit niet het geval is en het punt niet exposed is, maar gematcht aan een punt van A, dan is het nodig om nog minimaal twee punten toe te voegen aan het pad. Te beginnen met het punt van A wat gematcht is aan het zojuist gevonden punt van B. Dit is het punt van A wat hoort bij de rij waarin een 1 staat.

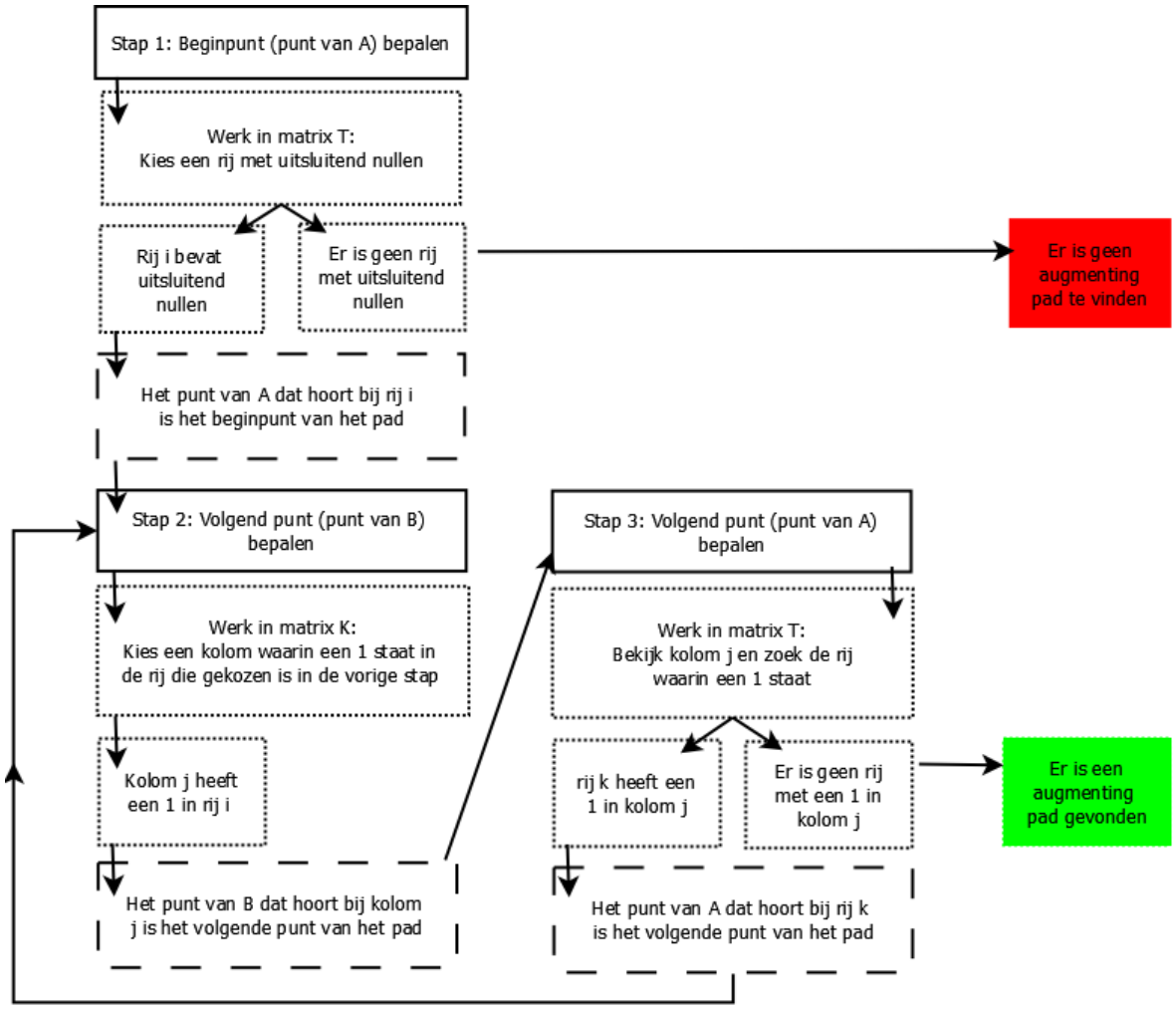


Figuur 2.5: Bepalen van het volgende punt

Het toevoegen van dit punt brengt het aantal punten van het pad op drie. Een augmenting pad heeft altijd een even aantal punten, dus we zoeken wederom een punt uit B en gaan na of dit punt een exposed punt is. Is dit niet het geval, dan moeten we opnieuw twee punten toevoegen aan het pad totdat het punt uit B wel een exposed punt is en er een augmenting pad gevonden is.

In een augmenting pad zijn alle punten verschillend van elkaar. Om te voorkomen dat er in een pad toch een punt zit dat meerdere malen wordt bereikt moeten de kolommen behorende bij de punten van B die aan het pad worden toegevoegd worden verwijderd van de matrix K nadat die punten aan het pad zijn toegevoegd.

In het schema in figuur 2.6 zijn alle stappen samengevoegd en wordt aan de hand van pijlen duidelijk hoe we het algoritme moeten doorlopen en wanneer het algoritme termineert. We zien heel duidelijk dat er twee vakjes zijn waar geen uitgaande pijl is. Komen we in het groene vakje, dan is het doel bereikt. Er is een augmenting pad gevonden. Het rode vakje betekent dat er geen augmenting pad bestaat ten opzichte van de matching die we vooraf bepaald hebben.



Figuur 2.6: Schema augmenting pad vinden

2.3 Fase 2: Augment M over Q

Er vanuit gaande dat het algoritme dat we zojuist hebben beschreven termineert in het groene vakje van het schema en dat er een augmenting pad is gevonden gaan we verder met fase 2: *Het augmenten van M over Q*. waarbij M de gevonden matching is en Q een gevonden augmenting pad. We zagen eerder al een bipartiete graaf G met bijbehorende matrix K en de matching M met bijbehorende matrix T. Om gebruik te kunnen maken van het gevonden augmenting pad Q is er ook nu een bijbehorende matrix nodig. Deze matrix noemen we de *padmatrix* P en definiëren we door:

$$p_{i,j} = \begin{cases} 1 & \text{als de zijde } (i,j) \text{ in het pad zit;} \\ 0 & \text{als de zijde } (i,j) \text{ niet in het pad zit.} \end{cases}$$

De stelling van König zegt ons dat bij het vinden van een augmenting pad de gebruikte matching niet maximaal is. Dit wil zeggen dat een andere matching, noem deze M', een kardinaliteit heeft die hoger is dan de kardinaliteit van de gebruikte matching M. Zo een *grotere* matching kunnen we maken met behulp van matching M en augmenting pad Q. Het is namelijk mogelijk om M te augmenten over Q. Dat wil zeggen dat alle

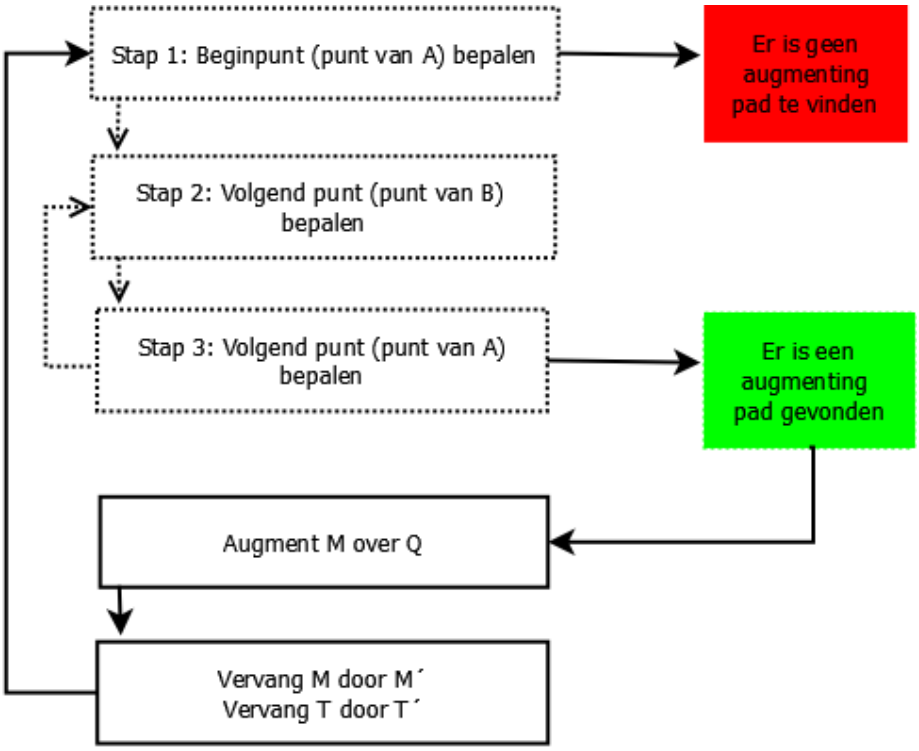
zijden die wel in het pad Q zitten, maar niet in matching M worden toegevoegd aan matching M. Daarnaast halen we alle zijden die zowel in de matching als in het pad zitten uit de verzameling M. Op deze manier zullen de zijden van het augmenting pad, die voorheen niet in de matching zaten de nieuwe matching M' vormen. Dit noteren we als volgt:

$$M' = M \triangle Q = (M - Q) \cup (Q - M) \tag{2.1}$$

Wanneer we dit omschrijven naar de matrices die we voorheen hebben gedefiniëerd krijgen we de nieuwe toewijzingsmatrix T' door:

$$T' = (T + P) \mod 2$$

We rekenen modulo 2 om een matrix te verkrijgen die uitsluitend uit nullen en enen bestaat. Op die manier kunnen we de nieuwe toewijzingsmatrix op dezelfde manier interpreteren als we deden bij de toewijzingsmatrix behorende bij de oude matching. Een 1 staat voor een verbinding en een 0 betekent geen verbinding tussen de twee punten.



Figuur 2.7: Algoritme maximum cardinality matching problem

Nu we weten hoe we een grotere matching kunnen maken, moeten we alleen nog de oude matching vervangen door de nieuwe en dus de oude toewijzingsmatrix vervangen door de nieuwe. Nadat we dat gedaan hebben beginnen we met de nieuwe matching weer van vooraf aan en doorlopen alle stappen in dezelfde volgorde waarbij we het doel hebben een augmenting pad te vinden en aansluitend de matching te augmenten over het gevonden pad.

Bij het uitbreiden van het schema van figuur 2.6 worden de zojuist beschreven stappen toegevoegd. Daaruit wordt duidelijk dat het algoritme alleen termineert op het moment dat er geen augmenting pad gevonden kan worden en dat we dus in het rode vakje terecht komen.

Stelling 2.2. (van Berge) Er bestaat geen enkel augmenting pad ten opzichte van de matching $M \iff$ Een matching M is maximaal.

Bewijs. ($\Leftarrow$) Neem M als de maximale matching en stel dat er een augmenting pad Q bestaat ten opzichte van die matching M, dan kunnen we M over Q augmenten zoals beschreven staat in vergelijking 2.1. Voor de matching M' die we hierdoor verkrijgen geldt $|M'| = |M| + 1$. Tegenspraak, want M was maximaal.
($\Rightarrow$) Neem nu aan dat er geen augmenting pad Q bestaat ten opzichte van de matching M en stel dat M niet de maximale matching is. Dat wil zeggen dat er een matching M' bestaat zodanig dat $|M'| > |M|$. Neem nu de deelgraaf H van G, waarbij $H := M \triangle M'$. H bestaat dan uit de zijden van G die horen bij $M \setminus M'$ of $M' \setminus M$. Om de stelling daadwerkelijk te bewijzen willen we de volgende beweringen laten zien:

i) Elk punt van H heeft een graad van hoogstens 2;

ii) Elke samenhangscomponent is een cyckel of een pad met alternerende zijden tussen M en M';

iii) Elk pad heeft een oneven aantal zijden waarbij de buitenste zijden in M' zitten.

Bewijs van i) Neem een willekeurig punt w van H en stel dat de graad, d, van dat punt hoger is dan 2, $d(w) > 2$. Dan is dat punt verbonden met meer dan 2 zijden uit M en M', dus is w het eindpunt van meer dan 1 zijde in de matchings M of M'. Dit is in tegenspraak met de definitie van een matching, dus heeft elk punt van H een graad hoogstens 2. *bewijs van ii)* Uit de constructie van H volgt dat de zijden van H alterneren tussen M en M'. In combinatie met bewering i) geldt dat een samenhangscomponent of een cyckel of een pad is. Zou dit namelijk niet het geval zijn, dan moet er een punt bestaan waarvoor de graad groter is dan 2. *bewijs van iii)* Wederom gebruiken we de constructie van H en het feit dat de zijden alterneren tussen M en M'. Daaruit volgt namelijk dat elke cyckel even is. We weten ook dat $|M'| > |M|$ en dat er dus meer zijden van M' dan M in H zitten. Dan moet er een pad Q in H bestaan zodanig dat de uiteinden van Q geen uiteinden zijn van zijden uit de matching M. Dus zijn de uiteinden van Q exposed punten. Dus is Q een augmenting pad ten opzichte van matching M. Tegenspraak, dus M is maximaal. □

Gevolg 2.2.1. Het algoritme van figuur 2.7 geeft ons altijd een maximale matching en is dus een oplossing van het *maximal cardinality matching problem*.

Nu rest ons alleen nog het bewijzen van de stelling van König. Hiervoor hebben we een puntenoverdekking nodig van de graaf G. Neem een verzameling L van punten van A en B waarvoor geldt dat elk punt van L bereikt kan worden via een augmenting pad vanuit een exposed punt van A. Neem vervolgens het complement van A met L en de doorsnijding van B met L; $C = (A \setminus L) \cup (B \cap L)$.

Bewijs. (Stelling van König) Om de stelling van König te bewijzen moeten we eerst aantonen dat de zojuist gedefiniëerde verzameling C een puntenoverdekking is. Vervolgens is het voldoende om te bewijzen dat $|M| \geq |C|$. De omgedraaide ongelijkheid

kunnen we aannemen vanwege de definitie van een puntenoverdekking: van elke zijden van M zit namelijk minimaal één eindpunt in C. Om het eerste deel te bewijzen stellen we dat C geen puntenoverdekking is van G. Dan bestaat er een zijde $(a, b) \in E$ zodanig dat zowel a als b niet in C zit, maar $a \in A \cap L$ en $b \in B \setminus L$. Stel nu dat zijde (a, b) in de matching zit, dan geldt:

> Als $a \in A \cap L$, dan zit a in L en dus zit ook b in L, want a wordt bereikt vanuit b.
>
> Als $b \in B \setminus L$, dan zit b niet in L en dus kan a ook niet bereikt worden via een augmenting pad uit een exposed punt van A.

Dus zit (a, b) niet in de matching, maar in $E \setminus M$. Dat betekent dat de zijde gericht is van A naar B en dat b bereikt kan worden vanuit een exposed punt van A. Dus geldt dat $b \in B \cap L$. Dit is in tegenspraak met de aanname dat b niet in L zit, dus is C een puntenoverdekking van G.
Nu moeten we nog bewijzen dat $|M| \geqslant |C|$. Daarvoor bekijken we de constructie van puntenoverdekking, waarvoor geldt dat

> geen enkel punt van $A \setminus L$ is exposed, want dan zat het punt in L.
>
> geen enkel punt van $B \cap L$ is exposed, want dat zou betekenen dat een exposed punt van B te bereiken is vanuit een exposed punt van A. Dit impliceert een augmenting pad.
>
> Er geen gematchte zijde (a, b) bestaat zodanig dat $a \in A \setminus L$ en $b \in B \cap L$, want dan zat $a \in L$.

Bovenstaande beweringen bewijzen dat elk punt uit C het eindpunt is van een andere zijde uit de matching. Hieruit volgt dat $|M| \geqslant |C|$. □

Hoofdstuk 3

Minimum weight perfect matching

In dit hoofdstuk gaan we laten zien hoe we met behulp van geheeltallig en lineair programmeren een perfecte matching kunnen vinden voor een toewijzingsprobleem. Er wordt in kaart gebracht aan welke eisen de data moet voldoen voordat we daadwerkelijk gebruik kunnen maken van deze vorm van programmeren. De begrippen *relaxatie* en *primal-dual algoritme* zijn een belangrijke leidraad voor dit hoofdstuk.

3.1 Stelling van Hall

In voorbeeld 1.3 zagen we een bipartiete graaf waarvoor de maximale matching geen semi-perfecte matching is. Hieruit concluderen we dat er niet voor elke bipartiete graaf een semi-perfecte matching bestaat. Om na te gaan of er voor een bipartiete graaf wel een semi-perfecte matching te vinden is kunnen we het algoritme uit hoofdstuk 2 gebruiken. Met dat algoritme vinden we een matching M met bijbehorende matrix T. Indien alle rijen of alle kolommen van T een 1 bevatten is de maximale matching in het bijzonder een semi-perfecte matching.

Opmerking. In een bipartiete graaf met ongelijke partities kunnen we geen perfecte matching vinden, hoogstens een semi-perfecte matching.

Uiteraard is er een mogelijkheid om van tevoren na te gaan of er een semi-perfecte matching bestaat. Een bekend voorbeeld hiervan is het *huwelijksprobleem* waarbij een groep vrouwen wordt uitgehuwelijkt aan een groep mannen onder de voorwaarde dat elke vrouw aan één man gematcht wordt en elke man aan één vrouw. Daarnaast heeft elke vrouw de mogelijkheid om haar voorkeuren door te geven. Uiteindelijk moet elke vrouw aan een man van haar keuze worden gekoppeld.

Stelling 3.1. (Huwelijksstelling van Hall) Iedere vrouw kan worden uitgehuwelijkt aan een man van haar keuze als iedere k vrouwen samen met op zijn minst k mannen willen trouwen

Veralgemenen we deze stelling naar een toewijzingsprobleem dan beschouwen we de vrouwen als partitie A en de mannen als partitie B. Er zijn alleen verbindingen mogelijk tussen de vrouwen aan de ene kant en de mannen aan de andere. Dit is duidelijk een bipartiete graaf. De huwelijksstelling komt dan neer op het volgende:

Stelling 3.2. (Stelling van Hall) Gegeven een bipartiete graaf $G = (V, E)$ met bipartitie A, B, $(V = A \cup B)$. G heeft een matching van grootte $|A|$ dan en slechts dan als voor elke $S \subseteq A$ geldt dat $|N(S)| \geqslant |S|$, met $N(S) = \{b \in B \, \exists a \in S \text{ met } (a, b) \in E\}$

Voor het bewijs van de stelling van Hall nemen we aan de graaf G een gerichte graaf is en dat elke zijde van E maar één richting op gaat en wel op de volgende manier:

$A \to B$: Zijden die niet in de matching zitten

$B \to A$: Zijden die wel in de matching zitten

Bewijs. ($\Rightarrow$) Dat elke deelverzameling van A minimaal zoveel buren heeft als het aantal punten in die verzameling is makkelijk te beredeneren uit tegenspraak. Als er een deelverzameling bestaat die minder buren heeft dan dat de verzameling groot is kunnen we nooit al die punten matchen aan een buur. Simpel gezegd: als er meer vrouwen zijn dan gekozen mannen, kunnen we nooit elke vrouw tevreden stellen met een man. En blijven er vrouwen zonder man over.
($\Leftarrow$) Stel dat we een maximale matching hebben gevonden, maar dat die niet van grootte $|A|$ is. Dan heeft A een exposed punt. We noemen dit punt w. Met behulp van de gerichte graaf G kunnen we bepalen welke punten van A en B bereikt worden via een gericht pad uit het exposed punt w. De bereikte punten van A zitten in de deelverzameling W en de bereikte punten van B zitten in de deelverzameling V. Omdat de doorsnijding van A en B geen punten bevat, bevat ook de doorsnijding van W en V geen punten. Hiermee gaan we het volgende bewijzen:

i) Alle punten van V zijn gematcht aan een punt van W

ii) $|W| > |V|$

Bewijs van i: Stel dat er een exposed punt v in V bestaat. v zit in V, dus er bestaat een gericht pad van een exposed punt w naar een exposed punt v. Dit gerichte pad is in het bijzonder een augmenting pad. Stelling ?? laat zien dat de matching dan niet maximaal kan zijn en dat is in tegenspraak met de aanname. De punten die gematcht zijn met de punten van V liggen dus in W, want ook die punten worden bereikt via een gericht pad uit w. *Bewijs van ii:* Er zijn evenveel gematchte punten in V als in W en er is in W een exposed punt w wat het aantal punten in W groter maakt dan het aantal punten in V. Dit is in tegenspraak met de aanname dat elke deelverzameling minstens zoveel buren heeft als het aantal punten in de deelverzameling, dus bestaat er geen exposed punt w in A en is de maximale matching van grootte $|A|$. □

3.2 Geheeltallig programmeren

Het checken van een bipartiete graaf op de stelling van Hall is erg tijdrovend. Voldoet de graaf aan de voorwaarden van Hall, dan hebben we alleen nog maar zekerheid dat er een matching van grootte $|A|$ bestaat,het geeft ons nog niet de semi-perfecte matching waarnaar we op zoek zijn. Daarom gaan we het anders aanpakken. Om er voor te zorgen dat we een semi-perfecte matching vinden, maken we van de bipartiete graaf een volledig bipartiete graaf. Dat wil zeggen dat elk punt van A verbonden wordt met elk punt van B. Zo voldoet elke deelverzameling van A automatisch aan de voorwaarde van Hall, als we er vanuit gaan dat A minstens zoveel punten beval als B.

Om onderscheidt te maken tussen de verschillende zijden, maken we van de volledig bipartiete graaf een gewogen graaf, waarbij we elke zijden een gewicht meegeven. De zijden die voor het toevoegen van zijden al in de graaf zaten krijgen een laag gewicht in tegenstelling tot de toegevoegde zijden, die een hoog gewicht krijgen. De precieze hoogte van het gewicht kan afhangen van de grootte van de partities. Bekijken we de nieuwe graaf, dan staat op elk element in de matrix een getal, wat het gewicht van de bijbehorende zijde weergeeft. We willen uiteindelijk een semi-perfecte matching vinden waarbij het totale gewicht van de zijden zo laag mogelijk is, zodat er zo veel mogelijk zijden uit de oorspronkelijke graaf in de matching zitten.

Definitie 3.1. Een *beste matching* is een semi-perfecte matching waarvoor geldt dat de som van de gewichten van alle zijden in de matching zo klein mogelijk is. M.a.w. $\sum_{(i,j)\in M} c_{ij}$ is minimaal waarbij c_{ij} het gewicht van zijde (i, j) is.

Het vinden van een beste matching kunnen we opvatten als het oplossen van het *minimal weight perfect matching problem*:

$$\text{Min} \sum_{i,j} c_{ij}x_{ij} \tag{3.1}$$

onder voorwaarde:

$$\sum_{j} x_{ij} = 1 \qquad j \in B \tag{3.2}$$

$$\sum_{i} x_{ij} = 1 \qquad i \in A \tag{3.3}$$

$$x_{ij} \geqslant 0 \qquad i \in A, j \in B \tag{3.4}$$

$$x_{ij} \text{ integer} \qquad i \in A, j \in B. \tag{3.5}$$

Het minimum weight perfect matching problem wordt genoteerd als een *geheeltallig programma*, waarin de *doelfunctie* geminimaliseerd wordt 3.1. Dit gebeurt altijd onder bepaalde voorwaarden. Per definitie kan elk punt van $A \cup B$ een eindpunt zijn van hoogstens 1 zijde van de matching. Bij een geheeltallig programma geldt deze voorwaarde ook nog steeds, maar geldt bovendien dat elk punt van $A \cup B$ een eindpunt moet zijn van een zijde uit de matching. Dat wil zeggen dat elk punt van A verbonden moet worden met exact een punt van B 3.2 en dat elk punt van B verbonden moet worden met exact een punt van A 3.3. De variabelen x_{ij} mogen onder geen enkele omstandigheid negatief zijn 3.4 en bovendien moet voor een geheeltallig programma gelden dat alle variabelen geheeltallig zijn 3.5.

De doelfunctie samen met alle voorwaarden zal een perfecte matching van minimaal gewicht geven, waarbij alle punten van de bipartiete graaf gematcht worden. Dit laatste is alleen maar mogelijk als de partities van de graaf even groot zijn. Wanneer de partities van de graaf al even groot zijn, kunnen we de volledig gewogen bipartiete graaf gebruiken voor het oplossen van het geheeltallig programma. Zijn de partities van de graaf niet even groot, dan moeten we ze eerst even groot maken. We nemen hiervoor aan dat $|A| < |B|$. Met deze aanname kan er niet worden voldaan aan voorwaarde 3.3 van het geheeltallig programma. Er zijn punten van B die niet worden verbonden aan een punt van A door het gebrek aan punten in A. Om er voor te zorgen dat er geen gebrek meer is aan punten in A voegen we punten toe aan A totdat A en B even groot zijn.

Definitie 3.2. Een *dummy* is een denkbeeldig punt dat aan de verzameling A wordt toegevoegd om de kardinaliteit van A met 1 te verhogen. We voegen dummy's toe totdat de kardinaliteit gelijk is aan de kardinaliteit van de verzameling B. We definiëren de verkregen verzameling A' door:

$$A' = A \cup D$$

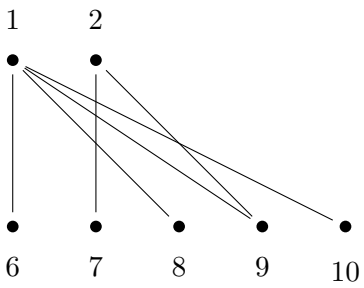
met

$$D = \{d : d \text{ een dummy}\} \qquad \text{en} \qquad |D| = \big||A| - |B|\big| .$$

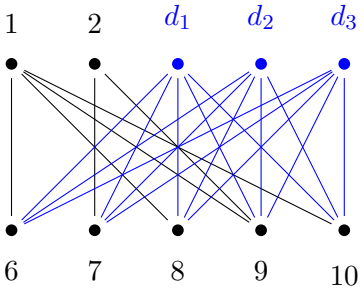
We hebben nu twee even grote partities A' en B. De dummy's die we toegevoegd hebben aan A worden net als alle andere punten van A met elke punt uit B verbonden. Deze zijden noemen we *dummy zijden* en krijgen allemaal hetzelfde gewicht. Dat wil zeggen:

$$c_{ij} = c_{kl};\ \forall i, k \in D$$

Voorbeeld 3.1. Een bipartiete graaf, waaraan 3 dummy's met bijbehorende zijden zijn toegevoegd op de manier die wordt hierboven beschreven. De gewichten van de zijden zijn weggelaten om de figuur overzichtelijk te houden.



(a) Bipartiete graaf



(b) Bipartiete graaf met dummy's en bijbehorende (blauwe) zijden

Nu de graaf voldoet aan alle voorwaarden van een geheeltallig programma rest ons alleen nog het toekennen van gewicht aan de dummy zijden. Het is van groot belang dat alle dummy zijden hetzelfde gewicht krijgen, aangezien de dummy's geen voorkeur hebben. Om te bepalen hoe hoog dit gewicht moet zijn bekijken we twee verschillende situaties. De eerste situatie is een volledig gewogen bipartiete graaf waarin n het gewicht van een dummy zijde is. De tweede situatie is dezelfde volledig gewogen bipartiete graaf, maar nu is het gewicht van een dummy zijde gelijk aan m, met $m \neq n$. Het totale gewicht van de zijden in de verschillende situaties definiëren we door $S = \sum c_{ij}x_{ij}$ en $S' = \sum c_{ij}x'_{ij}$. k is het aantal dummy's dat wordt toegevoegd aan A.
We zijn niet geïnteresseerd in het totale gewicht van de matching inclusief het gewicht van de k dummy zijden. We willen weten wat het voor invloed heeft op het totale gewicht van de matching exclusief de k dummy zijden.

lemma 3.1. $S - k \times n = S' - k \times m$, waarbij $S - k \times n$ en $S' - k \times m$ de totale gewichten van een matching zijn zonder de gewichten van de dummy zijden.

Bewijs. Stel dat $S - k \times n \neq S' - k \times m$.

$$S - k \times n < S' - k \times m$$

Dat wil zeggen dat

$$\sum c_{ij}x_{ij} < \sum c_{ij}x_{ij} \qquad \forall i \in A \backslash D$$

en dus dat

$$\sum c_{ij}x_{ij} + k \times m < S'$$

Dit is in tegenspraak met de aanname dat S' minimaal is. Dus $S - k \times n = S' - k \times m$. □

We kunnen een willekeurig gewicht hangen aan de dummy zijden, want het heeft geen invloed op het totale gewicht exclusief het gewicht van de k dummy zijden.

3.3 Relaxatie

De theorie is gevonden, de nodige aanpassingen aan de bipartiete graaf zijn gedaan, maar toch lukt het niet om met geheeltallig programmeren het toewijzingsprobleem op te lossen. Dit is het gevolg van de oplosbaarheid van een geheeltallig programma. Een geheeltallig programma is namelijk niet oplosbaar in polynomiale tijd, wat betekent dat het oneindig lang kan duren voordat het programma ons een perfecte matching geeft. Ondanks de onoplosbaarheid gebruiken we wel een geheeltallig programma. Niet om direct een perfecte matching te vinden, maar om een *lineair programma* te verkrijgen. Het lineaire programma moet de volgende vorm hebben:

$$\text{Min} \sum_{i,j} c_{ij}x_{ij}$$

onder voorwaarde:

$$\sum_{j} x_{ij} \qquad j \in B$$

$$\sum_{i} x_{ij} \qquad i \in A$$

$$x_{ij} \geqslant 0 \qquad i \in A, j \in B$$

We zien dat de doelfunctie en de eerste twee voorwaarden niet zijn veranderd ten opzichte van het geheeltallige programma. Ook bij lineair programmeren minimaliseren we de som van de gewichten in de matching onder de voorwaarde dat elk punt het eindpunt is van precies één zijde. De voorwaarde dat alle variabelen van een lineair programma niet negatief zijn is ook onveranderd. Het verschil zit hem in de laatste voorwaarde, 3.5, deze komt te vervallen. Dit proces noemen we *relaxatie*. De variabelen kunnen nu elke waarde van 0 t/m 1 aannemen in plaats van alleen de waarden 0 en 1. Doordat het domein van de variabelen vergroot is, zijn er veel meer mogelijke oplossingen en zal de optimale waarde van de doelfunctie in het lineaire geval hoogstens zo groot zijn als de optimale waarde van de doelfunctie in het geheeltallige geval.

Voorbeeld 3.2.

$$\text{Max } x + y \tag{3.6}$$

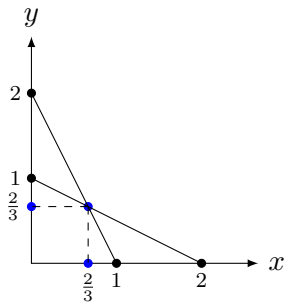
onder voorwaarde:

$$2x + y \leqslant 2 \tag{3.7}$$

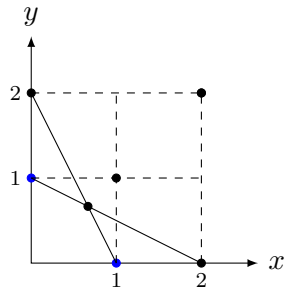
$$\frac{1}{2}x + y \leqslant 1 \tag{3.8}$$

$$x, y \geqslant 0 \tag{3.9}$$

$$(x, y \text{ integer}) \tag{3.10}$$



(a) lineair programma



(b) geheeltallig programma

We bekijken bovenstaand lineair programma (voorbeeld 3.2 zonder 3.10). Dit programma bestaat uit een doelfunctie en drie voorwaarden. Deze voorwaarden worden afgebeeld in figuur 3.2a door middel van de $x-$as en $y-$as (voorwaarde 3.9 en de twee snijdende lijnen (voorwaarden 3.7 en 3.8. Het snijpunt van deze lijnen is de optimale waarde van het lineaire programma en zal dus de grootste waarde geven bij het optellen van het x- en het y-coördinaat. Het oplossen van dit maximalisatieprobleem geeft de oplossing:

$$x = \frac{2}{3}, \qquad y = \frac{2}{3} \qquad \Longrightarrow \qquad x + y = 1\frac{1}{3} \tag{3.11}$$

We noteren de optimale waarde voor het lineair programmeer probleem door z_{LP} en de optimale waarde voor het geheeltallige programmeer probleem door z_{IP}. Gaan we terug naar het oorspronkelijke geheeltallige programma, dan voegen we voorwaarde 3.10 toe aan het lineaire programma dat we zojuist gebruikt hebben om een oplossing te vinden. Als de oplossing z_{LP} geheeltallig is, dan hebben we een optimale waarde gevonden voor het geheeltallig programmeer probleem. We zien in 3.11 dat dat niet het geval is, dus we gaan het geheeltallig programma ook oplossen. De oplossing van dit probleem zal zich bevinden op een roosterpunt (snijpunten van assen en de stippellijnen in figuur 3.2b) en de variabelen moeten geheeltallig zijn volgens voorwaarde 3.10. De mogelijke oplossingen zijn dan $(0, 0), (0, 1)$ en $(1, 0)$. Alle andere roosterpunten vallen buiten het oplossingsgebied. Een makkelijke rekensom geeft dat 1 de maximale waarde is voor de doelfunctie en dat $z_{LP} > z_{IP}$. Dit betekent dat de oplossing van het lineair programma optimaler is dan de oplossing van het geheeltallig programma.

Hoe kan toch de oplossing van een lineair programma een matching geven als de oplossing niet geheeltallig hoeft te zijn? Voordat we antwoord kunnen geven op deze vraag beschrijven we eerst een nieuw algoritme: Het *primal-dual-algoritme*. Vrij vertaald het

oorspronkelijke lineaire programma tegenover het omgeschreven lineaire programma. We gebruiken dezelfde gegevens, maar bekijken het probleem die vanuit een ander perspectief.
Het oorspronkelijke probleem beschrijft een doelfunctie die geminimaliseerd moet worden onder bepaalde voorwaarden, waarbij we de gewichten van de zijden als uitgangspunt nemen. Schrijven we het lineaire (oorspronkelijke) probleem om, dan krijgen we het *duale probleem*. We gaan niet meer uit van de gewichten van de zijden, maar koppelen gewichten aan de punten van de twee partities van G zodanig dat de som van deze gewichten maximaal het gewicht van de zijden is.
Neem aan dat u_i het gewicht is van $i \in A$ en v_j het gewicht van $j \in B$. Dan geldt voor alle i, j dat $u_i + v_j \leqslant c_{ij}$. Dit zijn de voorwaarden van het duale probleem. De doelfunctie komt voort uit de volgende ongelijkheid:

$$\sum_{i \in A} u_i + \sum_{j \in B} v_j \leq \sum_{(i,j) \in M} c_{ij} \tag{3.12}$$

Daarom zoeken we een maximale ondergrens en is de doelfunctie een maximalisatie van het linker gedeelte van 3.12. Deze voorwaarden en doelfunctie geven samen het duale probleem van het oorspronkelijk lineaire probleem.

$$\text{Max} \sum_i u_i + \sum_j v_j \tag{3.13}$$

onder voorwaarde:

$$u_i + v_j \leq c_{ij} \qquad\qquad i \in A, j \in B \tag{3.14}$$

Omdat het een convex probleem is gaat het hier om sterke dualiteit. Dit betekent dat de duale van het duale probleem het oorspronkelijke lineaire probleem geeft. De optimale waarde van het lineaire probleem is daarom gelijk aan de optimale waarde van het duale probleem. We kunnen dus het duale probleem oplossen om de optimale waarde te vinden.

Op een oplossing te vinden van het duale probleem introduceren we *complementary slackness*. Hiervoor schrijven we het duale probleem om na de vorm $w_{ij} \geq 0$ met $w_{ij} = c_{ij} - u_i - v_j$. Complementary slackness is de eis dat $w_{ij} = 0$ voor alle zijden (i, j). We zijn op zoek naar een perfecte matching M waarvoor geldt dat de ongelijkheid van 3.12 een gelijkheid is. Het totale gewicht van de perfecte matching is dan gelijk aan de waarde van de duale oplossing. Dit betekent dat de matching de optimale waarde bereikt heeft en dus de beste matching is, want de oplossing van het duale probleem is een bovengrens.

Het kan voorkomen dat voor een gegeven (u, v) niet voor iedere (i, j) geldt dat $w_{ij} = 0$ en dat we dus ook geen perfecte matching kunnen vinden.

Om een perfecte matching te vinden starten we met een duale mogelijke oplossing. Als we $u_i = 0$ nemen voor alle i en $v_j = min_i c_{ij}$ voor alle j dan voldoet de oplossing duidelijk aan de ongelijkheid van 3.12. We hebben nu een oplossing (u, v), ofwel de oplossing (u, v, w) als we de complementary slackness waarde ook meenemen in de oplossing. Zoals gezegd zijn we alleen nog maar geïnteresseerd in deelverzamelingen met zijden van de graaf waarvoor geldt dat $w_{ij} = 0$. Definiëer deze verzameling door

$E' = \{(i,j) : w_{ij} = 0\}$. Om te weten te komen of E' een perfecte matching heeft kunnen we het algoritme uit hoofdstuk 2 gebruiken. We vinden daarmee een maximale matching. Is de matching zo groot als de kardinaliteit van de partities, dan is de maximale matching de perfecte matching. Is dit het geval en bevat E' een perfecte matching, dan is geet de toewijzingsmatrix van die matching een mogelijke oplossing van het primaire probleem en voldoet de matching aan complementary slackness ten opzichte van de meest recente duale oplossing. Dit betekent dat de oplossing die we gevonden hebben optimaal moet zijn.

Het is natuurlijk mogelijk dat de duale oplossing waarmee we beginnen er niet meteen voor zorgt dat we een beste matching vinden. Zo kan het zijn dat we geen perfecte matching kunnen vinden in E'. Dan moeten we de duale oplossing aanpassen op een manier dat de waarde van de duale oplossing vergroot wordt. Dit zorgt er voor dat er meer oplossingen mogelijk zijn. Hiervoor gebruiken we de optimale puntenoverdekking. Neem het minimum van alle w_{ij} voor alle $i \in (A \cap L)$ en $j \in (B \setminus L)$ met $A \cap L$ en $B \setminus L$ zoals in het bewijs van de stelling van König wordt beschreven. Geef dit minimum de waarde δ en pas de duale oplossing aan op de volgende manier.

$$u_i = \begin{cases} u_i & \text{als} \quad i \in A \setminus L \\ u_i + \delta & \text{als} \quad i \in A \cap L \end{cases}$$

en

$$v_j = \begin{cases} v_j & \text{als} \quad j \in B \setminus L \\ v_j - \delta & \text{als} \quad j \in B \cap L \end{cases}$$

Nadat we de duale oplossing hebben aangepast checken we of de verzameling E' nu wel een perfecte matching bevat. Het algoritme zal doorgaan tot dat er een perfecte matching is gevonden die een optimale waarde geeft voor een toewijzingsprobleem.

Hoofdstuk 4

probleemstelling International Office

Het optimaliseren van een verdeling is het onderwerp wat in deze scriptie wordt onderzocht. Is het mogelijk om de optimalisatie van een verdeling te automatiseren? Deze vraag stelde het International Office een tijd geleden. Daar hebben ze namelijk te maken met het volgende *probleem*: Een groep studenten moet verdeeld worden over beschikbare universiteitsplekken zodanig dat zo veel mogelijk studenten ingedeeld wordt op een plek van hun keuze.

Op de vraag of het te automatiseren is kunnen we een kort antwoord geven. *Ja, het is mogelijk om de optimalisatie te automatiseren.* De data moet daarvoor in een matrix staan waar elke rij van deze matrix een student weergeeft en elke kolom een universiteit. Elk element van de matrix bevat een getal. Dit getal, wat we het gewicht van het element noemen, geeft aan of de student een voorkeur heeft voor de bepaalde universiteitsplek. Een 1 betekent dat de student het liefst naar deze plek gaat om te studeren, 2 en 3 geven de tweede en de derde keuze van een student aan. Naast het koppelen van de gewichten 1, 2 en 3 wordt er ook nog het gewicht 1000 toegekend aan alle overgebleven universiteitsplekken, welke niet behoren tot de voorkeur van de student.

Het is voor de studenten niet verplicht om 3 voorkeuren door te geven. In dat geval krijgt elke universiteitsplek die niet behoort tot de voorkeur van de student het gewicht 3 respectievelijk 2 bij het doorgeven van 2 respectievelijk 1 voorkeur(en). In desbetreffende rijen wordt hierdoor nergens het gewicht 1000 toegekend.

De partities in dit toewijzingsprobleem zijn niet even groot. Opgeteld zijn er meer plekken op de universiteiten te vergeven dan dat er student zijn die aanspraak willen maken op één van deze plekken. Om gebruik te kunnen maken van geheeltallig of lineair programmeren is dit wel noodzakelijk. Zoals in hoofdstuk 3 staat beschreven moeten we dus dummy-studenten met bijbehorende dummy-zijden gaan toevoegen totdat de partities even groot zijn en alle studenten verbonden zijn met alle universiteitsplekken. Zoals bewezen is in hoofdstuk 3 maakt het voor de toewijzing niet uit welk gewicht we hangen aan de dummy-zijden, zolang elke dummy-zijden maar hetzelfde gewicht heeft. Omdat ze niet van waarde zijn geven we deze zijden het symbolische gewicht 0.

Het toewijzingsprobleem van het International Office bestaat uit een dataset waar 150

studenten een keuze maken tussen 47 universiteiten over de hele wereld. In totaal zijn er op deze universiteiten 250 plekken te verdelen. Dit betekent dat we maar liefst 100 dummy-studenten moeten toevoegen om de matrix compleet te maken.

In Microsoft Office Excel is het mogelijk om met behulp van lineair programmeren een toewijzingsprobleem op te lossen. We gebruiken daarvoor *OpenSolver*. Nadat we de goede matrix hebben moeten we alleen nog het lineair programma invullen. Daarvoor moeten we de matrix kopiëren en plakken en de cellen leegmaken. Daarna gebruiken we een aantal formules om de voorwaarde van het lineair programma aan te geven. Te beginnen bij de doelfunctie.

Doelfunctie: =SOMPRODUCT([matrix1];[matrix2])

Waarbij matrix1 de matrix is met gewichten en matrix2 de gekopiëerde matrix zonder gewichten. Daarna gebruiken we de SOM-formule, om de cellen uit een rij of kolom bij elkaar optellen. Deze formule staat in de cel rechts van elke rij en onder elke kolom. Nu de cellen van de matrix nog leeg zijn, worden deze cellen opgevuld met het getal 0.

Nu alle aspecten voor het lineair programmeren klaarstaan kunnen we de matrix gaan optimaliseren met behulp van een aantal voorwaarden. We vullen eerst de cel die de doelfunctie aangeeft en het bereik (matrix2) in op de juiste plek van de *oplosser*. We moeten drie voorwaarden opgeven. De eerste twee voorwaarden geven aan dat de som van elke rij en elke kolom precies 1 moet zijn. De derde, en laatste voorwaarde, geeft aan dat alle cellen in de matrix binair moeten zijn. Nadat we alle voorwaarden hebben ingevoerd kan het programma beginnen met het zoeken naar een optimale verdeling.

Na een tijdje verschijnt in matrix2 een verdeling. In de cellen van de matrix die eerst nog leeg waren staan nu voornamelijk nullen, met in elke rij en elke kolom een 1. Deze 1 geeft aan dat de student en universiteit behorende bij de rij en kolom waar deze 1 staat gekoppeld zijn. De weergave van deze verdeling is niet overzichtelijk. We moeten nu bij alle enen gaan kijken in welke rij en kolom ze staan. Om een overzichtelijke verdeling te maken gebruiken we weer een aantal formules. Dit keer een combinatie van de INDEX-formule en de VERGELIJKEN-formule (engels: MATCH-formule). In de cellen rechts naast de matrix komt de naam van de universiteit te staan wanneer we de formules als volgt invullen:

=INDEX([matrix2];1;VERGELIJKEN(1;rij;0))

Dit hoeft alleen ingevuld te worden naast de rijen die horen bij de *echte* studenten. Het doet er namelijk niet toe aan welke universiteiten de dummy-studenten gekoppeld zijn.

Na het testen op de data van het International Office verkregen we een verdeling met totale som 181. De conclusie die we hieruit kunnen trekken is dat alle studenten die 3 voorkeuren hebben doorgegeven gekoppeld zijn aan één van deze voorkeuren. Er is dus een optimale verdeling te maken met behulp van lineair programmeren in Excel. De vraag van het International Office is hiermee beantwoord en opgelost. De nodige (basis)kennis van Excel is van belang, maar daarmee is het haalbaar om het probleem te automatiseren en scheelt het ieder jaar een hoop werk omdat het toewijzen niet meer handmatig gedaan hoeft te worden.

Bibliografie

[1] Goemans, M. X. (2007). *Assignment Problem.* Massachusetts Instituut voor Technologie.

[2] , Dantzig, G. B.& Thapa, M. N. (1914). *Linear Programming: 1 Introduction.* Stanford Universiteit.

[3] Schrijver, A. (2013). *Grafen: Kleuren en Routeren.* CWI, Amsterdam.

[4] Trevisan, L. (2011). *Linear Programming Relaxation.* Stanford University.

[5] Cameron, P. J. (1994). *Combinatorics: Topics, Techniques and Algorithms.* Cambridge Universiteit.

[6] Mason, A. (2012). *OpenSolver, An Open Source Add, in to Solve Linear and Integer Programmes in Excel.* Berlin.