Grafen met een optimale, perfecte matching om Blossom te controleren

Nienke Pastoor
Bachelorscriptie
Wiskunde en Toepassingen
Universiteit Utrecht

Begeleider: Rob Bisseling

Januari 2018

Contents

1 Introductie

Er zijn talrijke algoritmes die de optimale matching van bipartite grafen vinden. Hierentegen zijn er een stuk minder algoritmes die de optimale waarde van een niet-bipartite graaf vinden. Beperken we ons tot de gewogen, ongerichte grafen, vallen er nog een aantal af. Er zijn slechts enkele algoritmes die dit kunnen en deze leveren niet altijd de optimale oplossing.

Het doel van deze scriptie is om zelf grafen te creëren waarin we wel de optimale oplossing vinden en bewijzen. We zorgen ervoor dat de optimale oplossing tevens perfect is. De gecreëerde grafen zijn zo opgesteld dat deze oneindig ver uit te breiden zijn. We maken gebruik van gewogen grafen. De specificaties van de gewichten die we aan de zijden geven zijn zelf te bepalen. Wanneer we de optimale matching en waarde gevonden hebben, gaan we deze bewijzen.
Om op ideeën voor deze grafen te komen, hebben we Bondy en Murty [1] en daarnaast Harary [2] geraadpleegd. Gebaseerd op verscheidene figuren uit deze boeken hebben we zelf grafen opgesteld.

Hoewel over grafen veel te vinden is in literatuur, zijn er weinig bewijzen voor optimale matchings bekend. Wanneer een optimale graaf in een matching gevonden is, en deze ook bewezen is, kunnen we controleren hoe correct bestaande softwares of zelf geschreven programmas zijn. In deze scriptie doen we dit voor het Blossom-programma, waarover twijfels bestaan over de correctheid.

Het volgende deel is een korte inleiding over grafen en matching. In het derde deel gaan we twee bewijzen voor de optimale matching laten zien. Hierna in deel vier volgt een uitleg over Blossom. Vervolgens worden de optimale waarden die we gevonden vergeleken met de resultaten die Blossom geeft, waarna de conclusie volgt in het zesde en laatste deel.

2 Grafen en matching

2.1 Graaf

Een graaf kunnen we beschrijven als $G = (V, E)$ met V de verzameling knopen (Engels: vertices), en E de verzameling zijden (Engels: edges) [3]. Een zijde verbindt twee aangrenzende knopen met elkaar. Er bestaan gerichte en ongerichte grafen. Bij gerichte grafen hebben de zijden een bepaalde richting, bij ongerichte is dit niet het geval. In deze scriptie worden alleen ongerichte grafen behandeld. Daarnaast kan je onderscheid maken in gewogen en ongewogen grafen. In een gewogen graaf heeft elke zijde een bepaald gewicht. Deze kan zowel positief, negatief als 0 zijn. Hier gaan we werken met gewogen grafen.

2.2 Perfecte matching

Een matching M in een graaf $G = (V, E)$ is een set van zijden die niet dezelfde knoop delen [4]. We kunnen dus zeggen dat $M \subseteq E$, zo dat als $(u, v), (u', v) \in M \Rightarrow u = u'$. Een knoop is gematcht als dit een eindpunt is van een zijde die in de matching is opgenomen. Er wordt dus altijd een even aantal knopen opgenomen in een matching, met een maximum van het totale aantal n.

We spreken van een perfecte matching als iedere knoop i in de graaf gematcht is. Dit kan alleen wanneer n even is.

Een optimale matching is de matching die het maximale gewicht levert. Andere soorten matchings zijn hier niet relevant.

3 Optimale matchings die tevens perfect zijn

3.1 Wielgraaf

Een wielgraaf is een graaf met een knoop als middelpunt en daaromheen minstens 3 andere knopen. Elke knoop i is verbonden met $i-1$ en $i+1$ met $i = 2, 3, 4, ..., n-2$ en knoop $n-1$ is verbonden met knoop 1. Daarnaast is elke knoop in deze cirkelgraaf verbonden met knoop 0, het middelpunt.

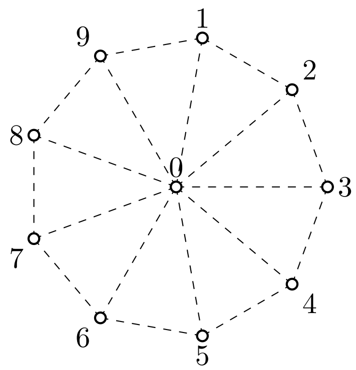

Figure 1: Wielgraaf met $n = 10$

Het bewijs voor de optimale waarde van de wielgraaf laten we zien aan de hand van de cirkelgraaf.

3.1.1 Cirkelgraaf met $n = 6k$

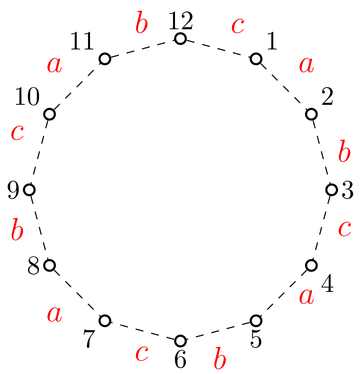

Figure 2: Cirkelgraaf met $n = 6k$ voor $k = 2$

Een cirkelgraaf is een graaf met n knopen voor $n \geq 3$. Hier is knoop i verbonden met knopen $i-1$ en $i+1$ met $i = 2, 3, 4, ..., n-1$ en knoop n is verbonden met knoop 1, waardoor er een cirkel ontstaat.

We geven de zijden van de cirkelgraaf drie verschillende gewichten, a, b en c. Specificaties van deze gewichten zijn $a, b, c \in \mathbb{R}^+$, $a < b < c$ en $a + b > c$.

Deze gewichten hebben de volgende volgorde:
$w(3i-2, 3i-1) = a,\ w(3i-1, 3i) = b,\ w(3i, 3i+1) = c$ met $i = 1, 2, 3, ..., \frac{n}{3} - 1$
Daarnaast is $w(n-2, n-1) = a,\ w(n-1, n) = b$ en $w = (n, 1) = c$.

Een periode definieren we als tweemaal de zijden met gewichten a, b en c, oftewel een periode bestaat uit zes zijden.
Neem $n = 6k$, wat ervoor zorgt dat de cirkelgraaf uit een geheel aantal periodes bestaat.

3.1.2 Stelling en bewijs

Stelling: De optimale matching voor een cirkelgraaf met $n = 6k$ met $k = 1, 2, 3...$ en gewichten $a < b < c$ met $a, b, c \in \mathbb{R}^+$, is een alternerende matching met zijden $(i-1, i)$ met $i = 2, 4, 6, ...n$, of een alternerende matching met zijden $(i, i+1)$ met $i = 2, 4, 6, ..., n-2$ daarbij zijde $(n, 1)$. De optimale matchings zijn in beide gevallen perfect en hebben een waarde van $\frac{n}{6}(a + b + c)$.

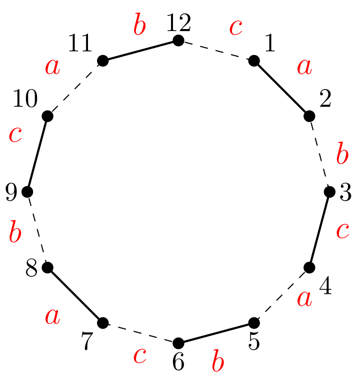

Figure 3: Een optimale matching voor een cirkelgraaf met $n = 6k$

Bewijs: We willen de optimale matching vinden. Meer dan twee zijden overslaan levert dit niet aangezien er dan *één* of meer tussengelegen zijden bestaan met twee aangrezende ongematchte knopen. Deze zouden we aan de huidige matching toe kunnen voegen zonder de bestaande matching te hoeven wijzigen. Wegens de positiviteit van de gewichten, verbetert dit de huidige waarde.

Kies er eerst voor om twee zijden over te slaan. Stel dat deze matching optimaal is. Wanneer we twee zijden overslaan hebben we drie verschillende gevallen, zie figuur 4.

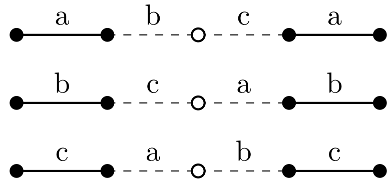

Figure 4: Twee zijden overslaan

In deze figuur zien we een lijngraaf dat onderdeel is van de cirkelgraaf. De bovenste graaf van figuur 4 kunnen we op twee manieren verbeteren, namelijk door zijde met gewicht c mee te nemen in de matching in plaats van a dan wel b. Vanwege $c > b > a$ verbetert dit de huidige matching respectievelijk met $c - a$ of $c - b$.
Ook de waarde van de tweede matching kan verhoogd worden met $c - b$, dit keer door de zijde met gewicht b te vervangen door die met c.
De matching in de onderste graaf is daarentegen wel optimaal en kan op geen enkele manier verbeterd worden.
Hieruit concluderen we dat wanneer er twee zijden worden overgeslagen, dat zijden met gewicht a en b zijn en er een zijde met gewicht c aan beide uiteinden moet zitten.
Nu willen we weten of twee zijden overslaan een betere waarde levert dan *één* zijde overslaan. Daarvoor gaan we kijken naar een langere lijngraaf, zie figuur 5.

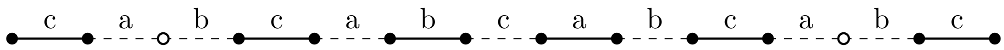

Figure 5: Lijngraaf met zijden a en b overgeslagen

We zien hier een lijngraaf met waarde $a+b+4c$. We gaan kijken of deze een augmenterend pad bevat, oftewel een pad dat een hogere waarde oplevert. Begin links en voeg nu na het toevoegen van zijde $(3i, 3i+1)$ met $i = 1, 2, 3...$ (de zijde met gewicht c), zijde $(3i+2, 3i+3)$ met $i = 1, 2, 3, ...$(zijde met gewicht b) toe. Voeg vervolgens in plaats van $(i-1, i)$, $(i, i+1)$ toe met $i = 2, 4, 6, ...$, oftewel, flip de matching, zie figuur 6.

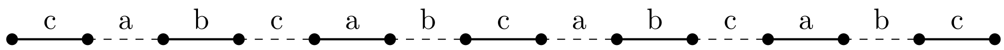

Figure 6: Geflipte matching

Na deze flip krijgen we een alternerende matching met een waarde van $2a + 2b + 3c$. Aangezien we zelf vastgelegd hebben dat $a + b > c$, is $2a + 2b + 3c > a + b + 4c$.

We hebben gezegd dat als we twee zijden overslaan, dit moet gebeuren tussen twee zijden met gewicht c. Nu hebben we laten zien dat wanneer we dat doen, een alternerende oplossing een beter resultaat geeft. Een alternerende matching is dus beter dan twee zijden overslaan, welke twee deze ook mogen zijn.

In algemenere vorm willen we weten of dit klopt voor grafen met $n = 6k$, dus ook voor grotere grafen. Om hier achter te komen kijken we eerst naar *één* periode met een alternerende matching, zie figuur 7.

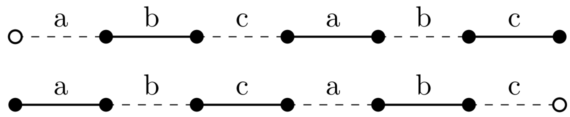

Figure 7: Periode met een alternerende matching

We zien dat de bovenste matching een waarde heeft van $b + a + c$. Wanneer we deze flippen, levert dit eveneens een matching met waarde $a + c + b$ op, wat we zien in de onderste graaf. Een alternerende matching in een gehele periode kunnen we flippen, zonder dat de waarde meer of minder wordt.

We hebben gezien dat de graaf uit figuur 5 geoptimaliseerd kan worden door in plaats van twee zijden over te slaan, de matching alternerend te maken. Doordat flippen van een alternerende matching in een gehele periode de waarde niet aantast, kunnen we een willekeurig aantal periodes met een alternerende matching toevoegen zonder dat de optimale waarde verandert als er eventueel geflipt wordt.

Aangezien een periode bestaat uit zes zijden en zes een even getal is, is knoop $n + 1$ ongematcht wanneer de matching begint bij knoop 1. Andersom, wanneer knoop $n + 1$ gematcht is, is knoop 1 ongematcht. We kunnen dus van de lijngraaf een cirkelgraaf maken zodat knoop n verbonden wordt met knoop 1, en de matching alternerend blijft.

Hiermee concluderen we dat wanneer we een cirkelgraaf hebben met een geheel aantal perioden, een alternerende matching optimaal is.
Per periode voegen we eenmaal $(a + b + c)$ toe. Aangezien we n knopen en dus n zijden hebben, en een periode uit zes zijden bestaat, is de optimale

waarde $\frac{n}{6}(a+b+c)$. Omdat we net hebben gezien dat flippen de waarde neutraal laat, maakt het geen verschil of we $(i-1,i)$ met $i=2,3,4,...,n$ in onze matching opnemen of $(i,i+1)$ met $i=2,3,4,...,n-2$ en daarbij zijde $(n,1)$.

3.1.3 Cirkelgraaf met $n = 6k+3$

Nu gaan we kijken wat er gebeurt wanneer een cirkelgraaf niet bestaat uit een geheel aantal periodes, maar eindigt met een halve. Dit krijg je wanneer $n = 6k+3$ met $k = 1,2,3,...$. Verder zijn alle specificaties voor de gewichten en de volgorde daarvan hetzelfde als graaf in 3.1.2.

3.1.4 Stelling en bewijs

Stelling: De optimale matching voor een cirkelgraaf met $n = 6k+3$ met $k = 1,2,3...$ en gewichten $a < b < c$ en $a,b,c \in \mathbb{R}^+$, is een alternerende matching. Er is 1 ongematchte knoop, knoop o, met $o \in \{3l-1: \ l = 1,2,3..., \frac{n-1}{3}\}$. De optimale matching is niet perfect en heeft een waarde van $\frac{n}{6}(a+b+c)+c$.

De opgenomen zijden in deze matching zijn ofwel $(i,i+1)$ met $i = o+1, o+3, o+5, ..., n-2$, zijde $(n,1)$ en zijden $(i,i+1)$ met $i = 2,4,6,...,o-2$ en $o \in \{3l-1: \ l = 1,3,5,..., \frac{n-4}{3}\}$ (o is even), ofwel zijden $(i,i+1)$ met $i = o+1, o+3, o+5, ..., n-1$ en zijden $(i,i+1)$ met $i = 1,3,5,...,o-2$ en $o \in \{3l-1: \ l = 2,4,6,..., \frac{n-1}{3}\}$ (o is oneven).

Bewijs: We kunnen nu verder gaan met waar we in 3.1.3 gebleven waren. We weten dat voor een geheel aantal periodes een alternerende matching optimaal is. Wanneer we een halve periode toevoegen blijft een alternerende oplossing optimaal.
Het verschil is dat n oneven is zodat er *één* knoop ongematcht blijft, en dus is er een moment waar twee zijden worden overgeslagen. We hebben drie plekken (periodiek gezien) waar dat kan gebeuren.
In figuur 4 hebben we gezien dat wanneer dit niet zijden met gewicht a en b zijn, de matching geoptimaliseerd kan worden door de matching te flippen. Sla dus de zijden over die grenzen aan knoop o, met $o \in \{3l-1: \ l = 1,2,3,..., \frac{n}{3}-1\}$ want deze knoop zit altijd tussen de zijden met gewichten a en b. Aangezien we nu *één* knoop overslaan, laat flippen de waarde van de matching niet neutraal. We moeten dus kijken welke zijden we op willen nemen in onze matching.

Na het overslaan van knoop o, nemen we zijde $(o+1, o+2)$ op in onze matching en we vervolgen met een alternerende matching. Wanneer o oneven is, ofwel $o \in \{3l-1: \ l = 2, 4, 6..., \frac{n-1}{3}\}$, alterneren we deze matching tot en met zijde $(n-1, n)$. Daarna alterneren we verder met zijden $(i, i+1)$ met $i = 1, 3, 5, ...o-2$. Echter, is o even, ofwel $o \in \{3l-1: \ l = 1, 3, 5..., \frac{n-4}{3}\}$, alterneren we na $(o+1, o+2)$ door tot en met zijde $(n-2, n-1)$. Vervolgens voegen we $(n, 1)$ toe en continueren we de alternerende matching met $(i, i+1)$ met $i = 2, 4, 6, ..., o-2$. We hebben nu opnieuw een alternerende matching maar dan met knoop o ongematcht. Aangezien de gewichten oplopend zijn en we de zijden met a en b overslaan, heeft de eerste zijde die we opnemen gewicht c. Vanwege de extra halve periode, eindigen we na alterneren ook met een zijde met gewicht c.

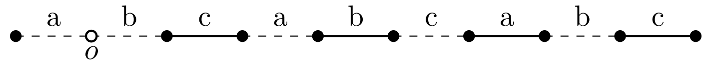

Figure 8: Lijngraaf met knoop o ongematcht die eindigt met een halve periode

De optimale oplossing voor een cirkelgraaf met $n = 6k+3$ is een alternerende matching met een knoop o ongematcht. Vanwege de laatste halve periode kunnen we nog *één* zijde extra toevoegen ten opzichte van de cirkelgraaf met $n = 6k$, en in de optimale oplossing is dat er een met gewicht c. De optimale waarde hiervan is $\frac{n}{6}(a+b+c)+c$. Aangezien flippen vanwege de overgeslagen zijden niet dezelfde waarde levert, moeten we vastleggen welke zijden we opnemen in de matching. Dit zijn de zijden $(i, i+1)$ met $i = o+1, o+3, o+5, ..., n-2$, zijde $(n, 1)$ en zijden $(i, i+1)$ met $i = 2, 4, 6, ..., o-2$, wanneer o even is. Is o oneven, zijn dit de zijden $(i, i+1)$ met $i = o+1, o+3, o+5, ..., n-1$ en zijden $(i, i+1)$ met $i = 1, 3, 5, ..., o-2$. Deze matching is niet perfect.

3.1.5 Wielgraaf

De wielgraaf is in principe hetzelfde als de laatste cirkelgraaf, maar nu met een knoop 0 toegevoegd, zie 3.1.1. Knoop 0 is met elke andere knoop, dus alle knopen uit de cirkelgraaf, verbonden. Dit zijn de zogeheten spaken. We gaan door met de cirkelgraaf uit 3.1.4. De wielgraaf bestaat uit $n = 6k+4$ knopen. Elke spaak heeft hetzelfde gewicht d, met $d \in \mathbb{R}^+$.

3.1.6 Stelling en bewijs

Stelling: De optimale matching voor een wielgraaf met $n = 6k+4$ met $k = 1, 2...$ en gewichten $a < b < c$ en d, met $a, b, c, d \in \mathbb{R}^+$, is een alternerende matching. Deze is perfect en heeft een waarde van $\frac{n}{6}(a+b+c)+c+d$. Deze waarde vinden we door de de volgende zijden op te nemen in onze matching: $(0, o)$ met $o \in \{3l-1: \ l = 1, 3, 5..., \frac{n-4}{3}\}$, $(i, i+1)$ met $i =$

$o+1, o+3, o+5, ..., n-2$, zijde $(n, 1)$ en zijden $(i, i+1)$ met $i = 2, 4, 6, ..., o-2$. De andere mogelijkheid om deze waarde te vinden is door deze zijden op te nemen; $(0, o)$ met $o \in \{3l - 1 : \ l = 2, 4, 6, ..., \frac{n-1}{3}\}$, de zijden $(i, i+1)$ met $i = o+1, o+3, o+5, ..., n-1$ en zijden $(i, i+1)$ met $i = 1, 3, 5, ..., o-2$.

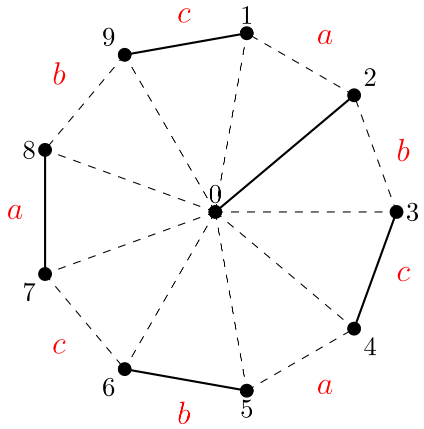

Figure 9: Een optimale matching voor de wielgraaf

Bewijs: Vanwege het oneven aantal knopen in de cirkelgraaf, is er altijd een knoop o ongematcht. Knoop 0 kan dus altijd met deze ongematchte knoop verbonden worden, zonder dat de optimale matching voor de cirkelgraaf met $n = 6k + 3$ veranderd hoeft te worden. Aangezien elke spaak hetzelfde gewicht heeft, maakt het voor de waarde van de matching niet uit welke spaak wordt toegevoegd. Voeg nu zijde $(0, o)$ toe aan de huidige optimale matching van de cirkelgraaf. De optimale waarde van de wielgraaf wordt de optimale waarde van de cirkelgraaf met daarbij opgeteld d. De optimale waarde met bijbehorende matching van de cirkelgraaf hebben we al gevonden in 3.1.5. De optimale waarde van de wielgraaf is dus $\frac{n}{6}(a + b + c) + c + d$. Omdat er altijd een spaak met waarde d toegevoegd kan worden, verandert deze de huidige matching nooit wat deze waarde ook exact is. We hoeven aan d dus geen eisen te stellen.

3.1.7 Conclusie

We concluderen dat wanneer een cirkelgraaf uit een geheel aantal periodes bestaat, een alternerende matching optimaal is. Het maakt hierbij niet uit of zijden $(i-1, i)$ of $(i, i+1)$ met $i = 2, 4, 6, .., n-1$ worden gematcht. Wanneer we een halve periode toevoegen aan dezelfde cirkelgraaf is het wel van belang welke zijden worden toegevoegd aangezien er een knoop ongematcht blijft. De twee zijden die we niet toevoegen aan de matching liggen tussen twee zijden met gewicht c in. Maken we vervolgens van de cirkelgraaf een wielgraaf, krijgen we opnieuw een perfecte matching aangezien we knoop 0 matchen met de ongematchte knoop o uit de cirkelgraaf. De optimale waarde van deze matching in de wielgraaf is $\frac{n}{6}(a + b + c) + c + d$.

3.2 De ronde-laddergraaf

Een ronde-laddergraaf is een cirkelgraaf met meerdere lagen. De binnenste laag bestaat uit een bepaald aantal knopen, noem dit aantal m. Elke laag verder naar buiten bezit ook m knopen. Elke knoop i is verbonden met $i-1$ en $i+1$ met $i = 1, 2, 3, ..., n-2$. Knoop 3 is niet verbonden met knoop 4, knoop 7 niet met 8 enzovoort, oftewel i is niet gelijk aan $4l-1$ met $l = 1, 2, 3, ..., \frac{n}{4}$. We leggen dus de extra eis $i \neq j$ met $j = 4l-1$ met $l = 1, 2, 3, ..., \frac{n}{4}$ op i. Daarnaast is i ook verbonden met $i+m$ en $i-m$, voor $i = m, m+1, m+2, ..., n-m-1$. Deze laatste zijden noem je de diagonalen. De graaf bestaat uit n knopen met $n = mk$, en $m = 3, 4, 5....$ Aangezien we een graaf van minstens twee lagen willen, is $k \geq 2$.

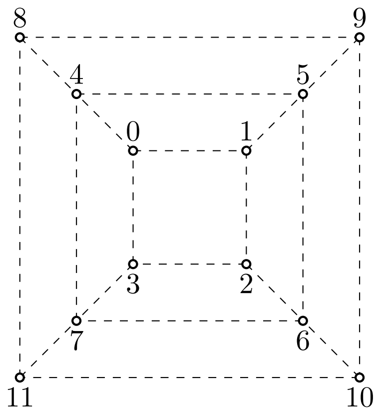

Figure 10: Ronde-laddergraaf met $m = 4$ en $k = 3$

We zien dat deze graaf een Cartesisch product is van G_1 en G_2. G_1 is een cirkelgraaf van m knopen, en G_2 is een lijngraaf met k knopen. Wanneer een graaf een Cartesisch product is, is $V(G) = V(G_1)\text{x}V(G_2)$. Knoop $u = (u_1, u_2)$ grenst dan aan knoop $v = (v_1, v_2)$ wanneer $u_1 = v_1$ en u_2 grenst aan v_2, of wanneer $u_2 = v_2$ en u_1 grenst aan v_1. Aan deze voorwaarden wordt voldaan dus we zien dat de ronde-laddergraaf een Cartesisch product [5].

3.2.1 Vierzijdig

Deze bestaat, wanneer we naar *één* laag kijken, uit vier knopen en vier zijden, oftewel $m = 4$. We geven de graaf vijf gewichten, $1 < a < b < c < d$. De volgorde hiervan is $w(4i, 4i+1) = a$, $w(4i+1, 4i+2) = b$, $w(4i+2, 4i+3) = c$, $w(4i+3, 4i) = d$ met $i = 0, 1, 2, ..., \frac{n-4}{4}$, en $w(i, i+4) = 1$ met

$i = 0, 1, 2, ..., n-5$. Verder geldt $a, b, c, d \in \mathbb{R}^+$.

3.2.2 Stelling en bewijs

Stelling: De optimale waarde van het vierzijdig ronde-laddergraaf met vijf gewichten $1 < a < b < c < d$ en $a, b, c, d \in \mathbb{R}^+$, is $\frac{n}{4}(b+d)$. Hiervoor nemen we zijden $(4i-3, 4i-2)$ en zijden $(4i-1, 4i-4)$ voor $i = 1, 2, 3...\frac{n}{4}$. Deze oplossing wordt geleverd door een perfecte matching.

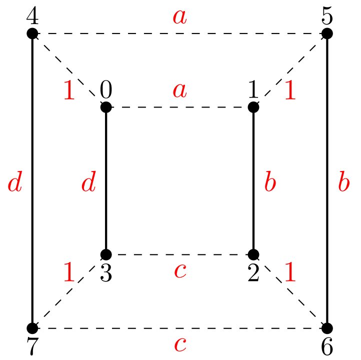

Figure 11: De optimale matching voor de vierzijdige ronde-laddergraaf

Bewijs: Laten we beginnen met acht knopen, oftewel twee lagen. We hebben de keuze om *één*, twee, drie of vier diagonalen in onze matching op te nemen. Stel je neemt er drie, zie figuur 12.

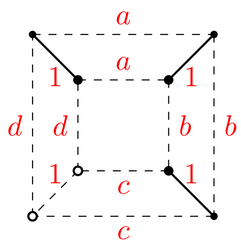

Figure 12: Drie diagonalen in matching

Naast deze drie zijden is er geen mogelijkheid om nog een zijde toevoegen zonder een diagonaal uit de huidige matching te verwijderen, behalve het laatste diagonaal. Je neemt dus nooit drie diagonalen op in je matching, want de laatste kan je toevoegen zonder de huidige matching te veranderen, en vanwege positiviteit van de gewichten levert dit een hogere waarde op. In plaats van drie hebben we nu vier diagonalen in de matching, wat een waarde van 4 oplevert.

Nu gaan we *één* diagonaal opnemen. We kunnen kiezen uit $(0,4)$, $(1,5)$, $(2,6)$ of $(3,7)$. Wanneer we een diagonaal $(i, i+4)$ met $i = 0,1,2,3$ toevoegen, kunnen we slechts twee van de vier tegenoverliggende zijden toevoegen, zie groene zijden figuur 13.

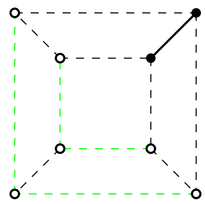

Figure 13: Mogelijke zijden om naast een diagonaal toe te voegen

Alle diagonalen hebben dezelfde waarde dus welk diagonaal toegevoegd wordt maakt voor de waarde van de matching niet uit. Omdat $2d > c+d > 2c > b+c > 2b > a+b > 2a$, kiezen we voor diagonaal (1,5), zodat we twee zijden met gewicht d toe kunnen voegen.

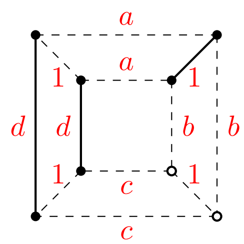

Figure 14: Een diagonaal in de matching

Nu zien we dat er geen zijde uit het vierkant meer toegevoegd kan worden, maar nog wel een diagonaal. Dit is diagonaal $(3,7)$. We vinden dus nooit een optimale oplossing wanneer we *één* diagonaal opnemen, aangezien er dan altijd ook een tweede kan worden toegevoegd. Nu hebben we twee diagonalen en tweemaal de zijde met waarde d. In totaal hebben we dus een waarde van $2d + 2$.
We concluderen hieruit dat wanneer we drie diagonalen meenemen in de matching, de vierde als enige zijde toegevoegd kan worden zonder een ander diagonaal eruit te halen. Ook wanneer we in eerste instantie *één* diagonaal toevoegen, komt er voor de optimale oplossing nog een tweede bij. De keuze is dus om twee of vier diagonalen op te nemen in de matching, en dit levert respectievelijk waardes $2 + 2d$ en 4 op.

Natuurlijk hebben we ook de mogelijkheid om geen diagonalen toe te voegen. We richten ons nu alleen op de zijden in het vierkant.
Kijken we eerst alleen naar het binnenste, dan zien we dat er in een vierkant maar twee zijden toegevoegd kunnen worden, namelijk de tegenover elkaar gelegen zijden.

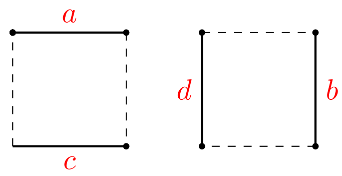

Figure 15: Matchingsmogelijkheden in een vierkant

We hebben dus de keuze om $a + c$ of $b + d$ toe te voegen. We weten $b > a$ en $d > c$, dus onafhankelijk van wat de exacte waardes zijn, levert $b + d$ toevoegen een hogere waarde op. Aangezien het buitenste vierkant losstaat van het binnenste wanneer we de diagonalen niet meenemen, geldt hetzelfde verhaal. We voegen dus tweemaal $b + d$ toe.

Kortom, wanneer we acht knopen hebben, hebben we de keuze om geen, twee of vier diagonalen op te nemen. Aangezien $2b + 2d > 2 + 2d > 4$, maken we de keuze om geen van de diagonalen mee te nemen, maar de zijden $(2, 3)$, $(4, 1)$, $(6, 7)$ en $(8, 5)$. De optimale waarde van een vierzijdig ronde-laddergraaf voor $n = 8$ is dus $2b + 2d$ en deze wordt geleverd door een perfect matching.

We gaan nu kijken wat er gebeurt als we een extra laag toevoegen. Ook in deze laag kunnen we twee tegenovergelegen zijden nemen, en net hebben we al gezien dat dan $b + d$ de beste oplossing geeft. Wanneer we gaan proberen of diagonalen toevoegen wellicht een hogere waarde levert, zien we dat we eigenlijk in dezelfde positie zitten als net. Wanneer we *één* diagonaal opnemen in de matching, vervallen er twee zijden met gewicht b of twee zijden met gewicht d. We kunnen dan wel een extra diagonaal toevoegen, maar dit levert een lagere waarde op. Zo kunnen we doorgaan, maar met dezelfde redenering die we net gebruikt hebben, zien we dat geen diagonalen opnemen in de matching het beste resultaat levert. Hetzelfde geldt voor alle lagen die we daarna nog toevoegen. Aangezien per laag alle knopen al gematcht zijn, voegen we dus nooit een diagonaal toe maar iedere laag de zijden met gewichten b en d. Elke laag heeft vier zijden, wat betekent dat we $\frac{n}{4}$ lagen hebben. Zo vinden we de optimale waarde van $\frac{n}{4}(b + d)$. We nemen hiervoor de zijden $(4i - 3, 4i - 2)$ en zijden $(4i - 1, 4i - 4)$ met$i = 1, 2, 3...\frac{n}{4}$.

3.2.3 Vijfzijdig

De vijfzijdige ronde-laddergraaf is dezelfde als die met vier zijden, maar dan met een extra knoop per laag, dus $m = 5$. Er is nu per laag een extra zijde en deze heeft gewicht e, met $e > d$ en ook $e \in \mathbb{R}^+$. De rest van de eigenschappen zijn hetzelfde. We willen een even aantal lagen, dus neem $n = 10k$. Leg hiernaast de eis op e dat $a + d > e$.
De volgorde van de gewichten is: $w(5i, 5i + 1) = a$, $w(5i + 1, 5i + 2) = b$, $w(5i + 2, 5i + 3) = c$, $w(5i + 3, 5i + 4) = d, w(5i + 4, 5i) = e$ met $i = 0, 1, 2, ..., \frac{n-5}{5}$, en $w(i, i + 5) = 1$ voor $i = 0, 1, 2, ..., n - 6$

3.2.4 Stelling en bewijs

Stelling: De optimale waarde van een vijfzijdig ronde-laddergraaf met gewichten $1 < a < b < c < d < e$ en $a, b, c, d, e \in \mathbb{R}^+$, is $\frac{n}{5} \cdot (e+c) + \frac{n}{10} \cdot 1$. Deze wordt gegeven door zijden $(5i - 3, 5i - 2)$ en $(5i - 1, 5i - 5)$ met $i = 1, 2, 3... \frac{n}{5}$, en daarnaast zijden $(10i + 1, 10i + 6)$ met $i = 0, 1, 2... \frac{n-10}{10}$ op te nemen in je matching.

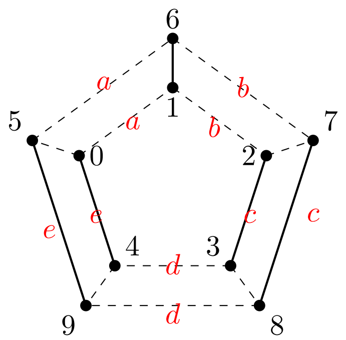

Figure 16: De optimale matching in een vijfzijdig ronde-laddergraaf

Bewijs: Neem eerst tien knopen. We gaan dit op dezelfde manier bekijken als de graaf met $4k$ knopen. Vanwege het oneven aantal knopen per laag, nemen we altijd minstens *één* diagonaal mee in de matching. Er is namelijk in beide lagen een ongematchte knoop, en aangezien de volgorde van de gewichten per laag hetzelfde is, zullen dit knopen i en $i + 5$ voor $i = 0, 1, 2, 3, 4$ zijn. Deze kunnen dus verbonden worden met elkaar.

Hetzelfde geldt voor vier diagonalen. De laatste diagonaal zal ook altijd toegevoegd kunnen worden en aangezien dit de enige zijde is die dit kan zonder het verwijderen van een diagonaal in de huidige matching, nemen we in plaats van vier altijd vijf diagonalen mee. We nemen dus nooit geen of vier diagonalen mee in onze matching.

Nu gaan we kijken of we dan *één* diagonaal willen toevoegen. De vier zijden die aan deze twee knopen grenzen, vallen dan af om toe te kunnen voegen. De zes zijden die tegenover de opgenomen diagonaal liggen, zijn nog wel toe te voegen, zie groene zijden in figuur 17.

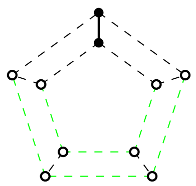

Figure 17: Mogelijke zijden om naast een diagonaal toe te voegen

Aangezien elke laag dezelfde volgorde van gewichten heeft, en deze los van elkaar staan vanwege het gebrek aan opgenomen diagonalen in dit gedeelte, kunnen we voor *één* laag bekijken welke zijden we meenemen in onze matching. Voor de andere laag geldt exact hetzelfde. Eerst gaan we laten zien dat je altijd een tweetal zijden neemt. Neem als voorbeeld het drietal zijden met gewichten a, b en c.

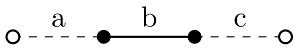

Figure 18: Deel van de graaf

Stel je voegt de middelste zijde toe. Vanwege het feit dat de gewichten oplopend zijn, zien we dat we zo nooit de optimale oplossing vinden, aangezien de zijde rechts altijd groter is. Wanneer we de rechter zijde toevoegen in de matching, kunnen we de linker meenemen aangezien beide aangrenzende knopen nog ongematcht zijn. in het voorbeeld kiezen we dus nooit voor de zijde met gewicht b, maar voegen we eerst zijde met gewicht c toe waarna we dat ook met a kunnen doen.

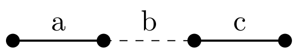

Figure 19: Verbeterde matching

Uitzondering is wanneer zowel de zijde met gewicht a als e in het drietal zit, oftewel wanneer diagonaal $(2, 7)$ of $(3, 8)$ in de matching opgenomen is. De volgorde van de gewichten is dan niet meer oplopend. Dit gebeurt in twee gevallen, die we nu gaan bekijken.

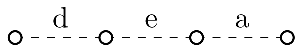

Figure 20: Eerste geval met zowel a als e in de toe te voegen zijden

Hier zien we dat we in eerste instantie niet perse de twee buitenste zijden kiezen, maar dat deze keuze afhankelijk is van de grootte van e. Daarentegen hebben we aan het begin vastgesteld dat $a + d > e$. Wanneer we ons beperken tot de matching waarbij we *één* diagonaal toevoegen, zien we dat $1 + a + d > 1 + e$. Ook hier kiezen we een tweetal van zijden.

Het andere geval waar zowel de zijde met gewicht a als met gewicht e in zit, is het volgende.

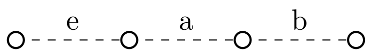

Figure 21: Tweede geval met zowel a als e in de toe te voegen zijden

In plaats van de zijde met gewicht a toe te voegen aan onze matching, voegen we de andere twee toe, aangezien zowel de rechter als de linker zijde een gewicht heeft die groter is dan a.
We kunnen concluderen dat in zo'n graaf wanneer we *één* diagonaal toevoegen, we altijd de buitenste vier zijden toevoegen gekozen uit de zes tegenoverliggende. Aangezien alle diagonalen hetzelfde gewicht w hebben, en vanwege de oplopende gewichten, zien we nu dat $e + c$ meenemen de hoogste oplossing levert. Dus kiezen we ervoor wanneer we *één* diagonaal toe voegen, de zijden $(5i - 3, 5i - 2)$ en $(5i - 1, 5i)$ met $i = 0, 1$ daarnaast meenemen, wat een oplossing geeft van $2e + 2c + 1$

Wanneer we twee diagonalen toevoegen hebben we de keuze om twee naast elkaar gelegen diagonalen te nemen, of er eentje over te slaan. Wanneer we twee of drie overslaan levert dit na een rotatie hetzelfde resultaat als geen of *één* overslaan. Wanneer we *één* overslaan, kan de tussengelegen diagonaal toegevoegd worden zonder dat een andere mogelijkheid verloren gaat. Als we twee naast elkaar gelegen diagonalen kiezen, hebben we nog vier zijden waarvan we twee toe kunnen voegen aan de huidige matching, zie groene zijden figuur22.

We zien dat we twee verschillende gewichten kunnen kiezen of tweemaal dezelfde. Aangezien alle gewichten per laag dezelfde volgorde hebben, nemen we tweemaal dezelfde. Als $w_1 > w_2$ kiezen we in zowel de eerste als de tweede laag de zijde met w_1, en vice versa.

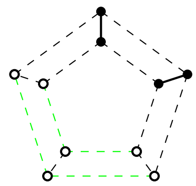

Figure 22: Mogelijke zijden om naast twee diagonalen toe te voegen

Hieruit volgt dat er nog een diagonaal 'vrij' is. We zien hier dat we nooit voor twee diagonalen kiezen. Laten we dus kijken naar de gevallen met drie diagonalen.

Naast drie diagonalen kan je nog *één* zijde per laag toevoegen. Vanwege $e > d > c > b > a$, en de uniciteit van de gewichten op de diagonalen, kiezen we ervoor om zijden met gewicht e toe te voegen en de overstaande drie diagonalen. Dit levert een waarde van $2e + 3$.

We kunnen vaststellen dat we altijd *één*, drie of vijf diagonalen in onze matching opnemen, wat respectievelijk waardes $2e + 2c + 1$, $2e + 3$ en 5 oplevert. We weten $c > 1$, en dus $2e + 2c + 1 > 2e + 3$. Vanwege deze zelfde eigenschap is $e > c > 1$ en dus is $2e + 2c + 1 > 5$. We concluderen dat *één* diagonaal meenemen in de matching het beste resultaat levert.

Omdat we een perfecte matching willen vinden is $n = 10k$. Voeg toch voor het bewijs nu 1 extra laag toe, zodat $n = 15$. We kunnen nu in de buitenste laag maximaal twee zijden toevoegen, los van de diagonalen. Vanwege dezelfde redenering als net kiezen we dan voor de zijden met gewichten c en e. Echter kunnen we ook kijken of we diagonalen toe willen voegen. Wanneer we een diagonaal $(i, i + 5)$ met $i = 5, 7, 8, 9$, worden zijden verwijderd en net hebben we gezien dat de zijden toevoegen een betere waarde levert dan de diagonalen. Dit doen we dus niet. Wanneer we diagonaal (6,11) willen opnemen, moeten we diagonaal (1,6) verwijderen. Nu krijgen we in principe dezelfde graaf waarmee we begonnen, maar nu binnenstebuiten. We vervangen dus geen zijden door diagonalen aangezien dit, vanwege dezelfde redenatie als net, een lagere waarde levert. Wanneer we (6,11) in plaats van (1,6) nemen verbeteren we de matching ook niet. In de buitenste laag voegen we de zijden met gewichten c en e toe aan de huidige matching.

Voeg nu nog een laag toe. Weer met dezelfde redenatie als net zien we dat we het beste in alle lagen zijden met gewicht c en e kunnen toevoegen. Er zijn dan nog vier aangrenzende ongematchte knopen. Match deze. We zien dat wanneer we een laag toevoegen, we exact een kopie krijgen van het eerste gedeelte. Wanneer we meer diagonalen opnemen gaat dat ten koste van de zijden en we hebben gezien dat dit geen betere waarde oplevert. We zien dat wanneer we tien knopen toevoegen, we een kopie krijgen van de matching met $n = 10$. Dit zal ook gebeuren voor elke volgende tien knopen die we toevoegen. We zien dus dat we deze laag uit kunnen breiden en dat we elke keer een exacte kopie krijgen van de matching met $n = 10$.

Dit levert in totaal een gewicht van $\frac{n}{5}(e + c) + \frac{n}{10}(1)$ op als we zijden $(5i - 3, 5i - 2)$ en $(5i - 1, 5i)$ met $i = 0, 1, 2, ..., \frac{n}{5}$ en $(10i + 1, 10i + 6)$ met $i = 0, 1, 2, ..., \frac{n-10}{10}$ opnemen. Dit is de optimale matching en deze waarde kan niet verbeterd worden. De optimale matching is tevens een perfecte matching.

4 Blossom

4.1 Algoritme

Het Blossom algoritme is in 1965 bedacht door Jack Edmond. Het vindt een perfecte matching met minimale kosten in een ongerichte gewogen graaf in polynomiale tijd. Het is met de tijd erg verbeterd. Huidig is de beste oplossing met de minste runtijd gevonden door Gabow[6]. Het programma wat voor deze scriptie gebruikt is, is BlossomV[7].

4.1.1 Werking

In deze sectie geven we een schets van het bewijs van het Blossom-algoritme, om hier een indruk van te krijgen. Voor details, zie [7]. Deze informatie komt uit $BlossomV:\ a\ new\ implementation\ of\ a\ minimum\ cost\ perfect\ matching\ algorithm$.

Onthoud dat V de verzameling knopen is, en E de verzameling zijden in een graaf. We hebben een ongerichte, gewogen graaf G. De kosten van een zijde geven we aan met c_e. Het doel is om een perfecte matching M te vinden met minimale kosten $c(M)$. Neem S als een deelverzameling van V, $S \subseteq V$. Neem vervolgens $\delta(S) = \{(u,v) \ \in E,\ u \in S,\ v \in V - S\}$, oftewel $\delta(S)$ is een verzameling begrenzingszijden. Hebben we een enkele knoop $v \in V$, nemen we $\delta(v) = \delta(\{v\})$. Laat O de verzameling zijn van alle deelverzamelingen van V met een oneven aantal elementen en minstens drie knopen. Vector x is een dummy-variabele die alleen waarde 0 of 1 aanneemt, x$\in \{0,1\}$. Wanneer zijde e opgenomen wordt in de matching, is $x_e = 1$. Zo niet, dan is $x_e = 0$.
Met deze informatie gaan we nu kijken naar de LP-formulering van het Blossom-algoritme.

PRIMAL

min $\sum_{e \in E}\ c_e x_e$

s.t. $x(\delta(v)) = 1 \qquad \forall v \in V$
$x(\delta(S)) \geq 1 \qquad \forall S \in O$
$x_e \geq 0 \qquad \forall e \in E$

DUAL

max $\sum_{v \in V}\ y_v + \sum_{S \in O} y_s$

s.t. $slack(e) \geq 0 \qquad \forall e \in E$
$y_S \geq 0 \qquad \forall S \in O$

met $slack(e) = c_e - y_u - y_v - \sum_{S \in O: e \in \delta(S)} y_s$

Hier betekent *slack* de gereduceerde kosten van zijde $e = (u, v)$.

Zijden e waar de *slack* 0 is, worden *tight* genoemd. Dit betekent dat de zijde volledig gebruikt is. De vectoren met de naam x en y worden op zo'n manier aangepast zodat x uiteindelijk maximaal is waardoor er sprake is van een perfecte matching. Wanneer dit gebeurt, wordt er aan de volgende voorwaarden voldaan:
$slack(e) > 0 => x_e = 0$
$y_s > 0 => x(\delta(S)) = 1$.

Een *blossom* is een cirkelgraaf die onderdeel is van de gehele graaf. Deze bezit een oneven aantal knopen.

Elke knoop v heeft een label $l(v) \in \{+, -, \varnothing\}$. Knopen waarvoor de label leeg is worden vrije knopen genoemd. Deze zijn altijd verbonden met een andere vrije knoop, via een *tight* zijde, en deze zijde is opgenomen in de matching. Als het label niet leeg is, behoort v tot een alternerende boom. Wanneer $l(v) = -$, dan heeft de voorliggende knoop $l(v) = +$. Deze twee knopen zijn met elkaar verbonden met een *tight* zijde, die niet in de matching is opgenomen. Wanneer $l(v) = +$, dan heeft de voorliggende knoop een negatief label, $l(v) = -$. Deze zijn verbonden met een *tight* gematchte zijde. De enige uitzondering is de *root* van de boom. Deze heeft $l(v) = +$, maar deze is niet gematcht. Een knoop met $l(v) = -$ is altijd verbonden met twee knopen $l(v) = +$. Andersom kan een knoop met $l(v) = +$ verbonden zijn met meerdere knopen of juist met geeneen.

Het algoritme itereert tussen de "primal updates" en de "dual updates".

We gaan nu eerst kijken naar de "primal updates". Hierin worden vier handelingen gebruikt.
GROW: Als zijde (u, v) *tight* is, $l(u) = +$ en $l(v) = \emptyset$, dan kan de boom waar u in zit 'groeien', door zijde (u, v) toe te voegen aan de boom.
AUGMENT: Als zijde (u, v) *tight* is, $l(u) = l(v) = +$ en u en v behoren beide tot een verschillende boom, dan de som van matching x worden vergroot door x_e te flippen, oftewel door de alternerende bomen beide te flippen. Hierna zijn alle knopen zogeheten vrije knopen.
SHRINK: Als zijde (u, v) *tight* is, $l(u) = l(v) = +$, en u en v behoren tot dezelfde boom, dan is er een cirkelgraaf onderdeel van de gehele graaf met een oneven aantal knopen en zijden. Deze kan dan samengetrokken worden tot een *blossom*. De knopen in deze *blossom* zijn nu vrij.
EXPAND: Als knoop v een blossom is met $y_v = 0$ en $l(v) = -$, dan kan deze knoop uitzetten.

De enige handeling die daadwerkelijk de huidige matching x verandert, is AUGMENT. De som van x wordt dan uitgebreid met 1, dus er zijn maximaal $\frac{n}{2}$ *augmentations* (verbeteringen).

Vervolgens gaan we kijken naar de "dual updates".
Geef een boom aan met de letter T. De "dual updates" laten de bomen en *blossoms* intact. Voor elke boom T kiezen we een hoeveelheid voor de duale verandering $\delta(T) \geq 0$. Voor iedere knoop $v \in T$ verandert de duale variabele y_v als volgt: Als $l(v) = -$, dan wordt $y_v = y_v - \delta_T$. Als $l(v) = +$, dan wordt $y_v = y_v + \delta_T$. De waarde van de duale functie wordt dan vergroot met $\sum_T \delta_T$. We krijgen op deze manier de volgende beperkingen op δ_T:

1. $\delta_T \leq slack(u, v)$	(u, v) is een $(+, \emptyset)$ zijde, $u \in T$
2. $\delta_T + \delta'_T \leq slack(u, v)$	$(u, v)$is een $(+, +)$ zijde, $u \in T, v \in T', T \neq T'$
3. $\delta_T \leq \frac{slack(u,v)}{2}$	(u, v) is een $(+, +)$ zijde, $u, v \in T$
4. $\delta_T \leq y_v$	v is een " $-$ "knoop die een *blossom* is, $v \in T$
5. $\delta_T - \delta'_T \leq slack(u, v)$	(u, v) is een $(+, -)$ zijde, $u \in T, v \in T', T \neq T'$

Stel dat δ_T voor boom T de maximale waarde bereikt zodat *één* van de beperkingen *tight* wordt. Als dit het geval is voor zijde van de eerste beperking, dan volgt de GROW handeling. Worden de tweede, derde of vierde zijden *tight*, dan volgt respectievelijk AUGMENT, SHRINK, EXPAND. De laatste beperking heeft geen directe progressie in boom T. De tijd die dit algoritme heeft een grens $O(n^2 m)$ [7].

5 Resultaten

Nu gaan we Blossom gebruiken en kijken of deze op de juiste optimale oplossing komt. In C heb ik voor de verschillende grafen een programma geschreven zodat zelf het getal k te bepalen en in te vullen is, en dat Blossom deze kan gebruiken. Blossom laat dan het algoritme los op de grafen en deze vindt zo de minimale kosten. Aangezien het doel van deze scriptie was om de maximale waarde te vinden, vermenigvuldig ik alle gewichten met -1. Nu betekenen de minimale kosten de maximale waardes vermenigvuldigd met -1.

5.1 Cirkelgraaf met $n = 6k$

De optimale oplossing voor de cirkelgraaf met $n = 6k$ is $\frac{n}{6}(a + b + c)$.
De cirkelgraaf heeft gewichten $a, b, c \in \mathbb{R}^+$ met $a < b < c$ en $a + b > c$.
Ik heb hier gekozen voor $a = 2$, $b = 3$ en $c = 4$, zodat aan alle voorwaarden wordt voldaan. In het C-programma zijn hier dus de negatieve waarden ervan genomen.

De optimale waarden volgens Blossom met k zijn:
$k = 20$: 180 berekend in een verwaarloosbare tijd.
$k = 5000$: 45000 berekend in 4,429 seconden.
$k = 8500$: 76500 berekend in 15,770 seconden.

5.2 Wielgraaf met $n = 6k + 4$

De optimale oplossing voor de wielgraaf met $n = 6k+4$ is $\frac{n}{6}(a+b+c)+c+d$.
Deze graaf heeft gewichten $a, b, c, d \in \mathbb{R}^+$ met $a < b < c$ en $a + b > c$ en d is vrij te kiezen.
Ik heb hier gekozen voor $a = 2$, $b = 3$, $c = 4$ en $d = 1$.

De optimale waarden volgens Blossom met k zijn:
$k = 30$: 275 berekend in een verwaarloosbare tijd.
$k = 1500$: 13505 berekend in 0,438 seconden.
$k = 8000$: 72005 berekend in 15,770 seconden.

5.3 Vierzijdige ronde-laddergraaf met $n = 4k$

De optimale oplossing voor de vierzijdige ronde-laddergraaf met $n = 4k$ is $\frac{n}{4}(b + d)$.
Deze graaf heeft gewichten $1, a, b, c, d \in \mathbb{R}^+$ met $1 < a < b < c < d$.

Ik heb hier gekozen voor $a = 2$, $b = 3$, $c = 4$ en $d = 5$.

De optimale waarden volgens Blossom met k zijn:
$k = 400$: 3200
$k = 7000$: 56000
$k = 9000$: 72000

Deze waarden zijn allemaal berekend in een verwaarloosbare tijd.

5.4 Vijfzijdige ronde-laddergraaf met $n = 5k$

De optimale oplossing voor de vijfzijdige ronde-laddergraaf met $n = 5k$ is $\frac{n}{5}(c + e) + \frac{n}{10} \cdot 1$.
Deze graaf heeft gewichten $1, a, b, c, d \in \mathbb{R}^+$ met $1 < a < b < c < d < e$ en $a + d > e$.
Ik heb hier gekozen voor $a = 2$, $b = 3$, $c = 4$, $d = 5$ en $e = 6$.

De optimale waarden volgens Blossom met k zijn:
$k = 60$: 630 berekend in een verwaarloosbare tijd.
$k = 4500$: 47250 berekend in 0,003 seconden.
$k = 9500$: 99750 berekend in 0,008 seconden.

6 Conclusie

Aangezien er in de huidige literatuur amper bewijzen te vinden zijn voor optimale matchings in grafen, wilden we hier onderzoek naar doen en zelf deze bewijzen vinden voor zelf opgestelde grafen. De eerste graaf die we gecreëerd hebben is de wielgraaf met $n = 6k + 4$. De optimale matching hebben we bewezen en daarbij de optimale waarde $\frac{n}{6}(a + b + c) + b + d$ gevonden. Dit hebben we gedaan aan de hand van de cirkelgraaf met $n = 6k$ en deze vervolgens uitgebreid naar $n = 6k + 3$. De tweede graaf die we gecreëerd hebben is de vierzijdige ronde-laddergraaf met $n = 4k$. Ook deze optimale matching is bewezen en deze levert een waarde van $\frac{n}{4}(b + d)$. Deze ronde-laddergraaf hebben we uitgebreid naar *één* met vijf zijden, zodat $n = 5k$. Deze optimale matching geeft een waarde van $\frac{n}{5}(c + e) + \frac{n}{10} \cdot 1$. Vervolgens hebben we deze grafen omgeschreven in een C-programma zodat we het Blossom-algoritme hierop los kunnen laten. Voor elke graaf die we getest heben in Blossom en voor elke k die we daarvoor gekozen hebben, komt Blossom op het juiste resultaat uit. Voor deze grafen en de bijbehorende gewichten kunnen we dus niet aantonen dat Blossom niet altijd correct is. Nu hoeven we niet meteen uit te sluiten dat Blossom nooit fout zit. In de toekomst kunnen er meer grafen worden bedacht met bijbehorende gewichten zodat er wellicht wel aangetoond kan worden dat Blossom soms naast de optimale oplossing zit. Verder kunnen nu ook andere algoritmes getest worden met de grafen die hierboven genoemd zijn. Voor deze grafen is de optimale oplossing nu bewezen.

7 Referenties

1) Bondy, J. A., & Murty, U. S. R. (2008). Graph theory (graduate texts in mathematics). Graduate Texts in Mathematics, 44.

2) Barnes, J. A., & Harary, F. (1983). Graph theory in network analysis. Social networks, 5(2), 235-244.

3) Weisstein, Eric W. "Graph." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/Graph.html

4) Weisstein, Eric W. "Perfect Matching." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/PerfectMatching.html

5) Weisstein, Eric W. "Graph Cartesian Product." From MathWorld–A Wolfram Web Resource. http://mathworld.wolfram.com/GraphCartesianProduct.html

6) Gabow, H.N.: Data structures for weighted matching and nearest common ancestors with linking. In: Proceedings of the 1st Annual ACM-SIAM Symposium on Descrete Algorithms, pp. 434-443 (1990)

7) Kolmogorov, Vladimir (2009), "Blossom V: A new implementation of a minimum cost perfect matching algorithm", Mathematical Programming Computation, 1 (1): 4367