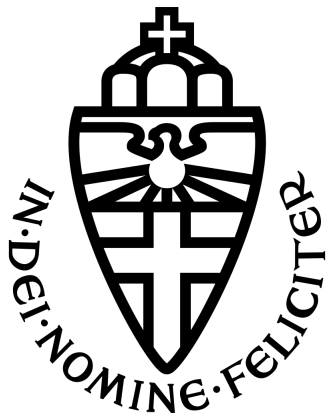
Verslag

Modellenpracticum

Royal Smit

Voorjaar 2016

Simone Clarisse
Iris van der Giessen
Rens Peters
Daja van de Vosse
Anita Kosman
Sterre den Breeijen



Inhoudsopgave

Inleiding

Voor het vak modellenpracticum hebben we een project doorlopen bij Royal Smit, gevestigd in Nijmegen. Royal Smit produceert transformatoren, deze worden gebruikt om wisselspanning te verhogen of te verlagen en/of het scheiden van primaire en secundaire stroomkringen. Ze bestaan uit drie spoelen die van metaal gemaakt zijn. Om een transformator te maken, moet veel metaal 'geslit' (gesneden) wordt uit enorme moederrollen, de rollen metaal die het bedrijf inkoopt. In één order van een transformator kunnen verschillende breedtes van het metaal nodig zijn. Na het slitten wordt het metaal in stukken geknipt en gaat het naar de stapelplaats. Hier worden de stukken metaal op elkaar gestapeld zodat een bij benadering ronde kern ontstaat.

Nu wordt het slitproces niet op een optimale manier ingepland. De verschillende breedtes worden uit een moederrol gehaald, zonder dat er rekening gehouden wordt met een al te groot materiaalverlies. Het beperken van dit verlies is een interessant vraagstuk voor wiskundestudenten.

Om het materiaalverlies te minimaliseren, hebben we een programma geschreven in C++. In dit verslag worden onze verschillende manieren van aanpak uiteengezet met uitleg van de theorie. Sommige manieren bleken niet te werken binnen aanzienbare tijd, dit zal ook toegelicht worden.

Het project is helaas niet af. Zoals in de probleemstelling uitgewerkt staat, is het project opgesplitst in drie fasen. De eerste fase is af, maar de laatste twee fasen waren dusdanig gecompliceerd dat het niet in een halfjaar is gelukt.

Graag zouden wij onze contactpersoon Mark Jansen, Torsten Geurtz en onze begeleider Wieb Bosma bedanken voor het meewerken en meedenken aan ons project.

Probleemstelling

Het probleem waar Royal Smit tegenaan loopt, is dat er veel materiaal verloren gaat bij het slitten van breedtes uit moederrollen. Het tweede deel van het probleem gaat over de kosten. Is voorgeslit materiaal inkopen goedkoper dan alles zelf te slitten?

Om dit probleem op te lossen, is het opgesplitst in drie fasen.

- Fase 1 "Proof of principle": dit is een basismodel van het programma met een werkbare oplossing, het geeft een oplossing met zo min mogelijk materiaalverlies uit een gegeven voorraad moederrollen.
- Fase 2 Inkoopadvies moederrollen: dit programma geeft aan welke kwaliteit en afmetingen moederrollen Royal Smit moet inkopen om het materiaalverlies te minimaliseren.
- Fase 3 "Make or buy": dit programma geeft aan of Royal Smit beter voorgeslitte materialen in kan kopen voor een orderbreedte of dat het goedkoper is deze zelf te slitten uit een ingekochte moederrol.

Fase 1

Fase 1 heet "Proof of principle". Wat we willen bereiken in fase 1 is een basismodel van het programma met een werkbare oplossing. Dit basismodel geeft voor één order de optimale keuze uit mogelijke moederrollen van verschillende breedtes. Voor dit basismodel hebben we dus gegevens nodig van deze order, namelijk de benodigde breedtes, het benodigde aantal kilo's, het benodigde materiaal en de start- en leverdatum van de order. Om tot het basismodel te komen, verdelen we fase 1 in verschillende deelfasen A, B en C.

In deelfase A bekijken we moederrollen van een gelijke breedte. Omdat we maar één soort moederrol bekijken is dit probleem vergelijkbaar met het knapzakprobleem.

Probleem 0.1 (Het knapzakprobleem)**.** *Gegeven een verzameling objecten met elk een gewicht en een waarde, bepaal een deelverzameling van objecten die samen in een knapzak passen. De knapzak kan een maximaal gewicht dragen en de objecten moeten samen een zo groot mogelijke waarde hebben.*

We kunnen dit als volgt wiskundig bekijken. Stel we hebben verzameling van objecten $\{1, \ldots, m\}$ waarbij elke $1 \leq i \leq m$ gewicht g_i en waarde w_i heeft. Zij a gegeven. Bepaal

$$\max\{\sum_{i \in I} w_i \mid I \subseteq \{1, \ldots, m\} \text{ en } \sum_{i \in I} g_i \leq a\}.$$

In ons geval is de moederrol een knapzak en de orderbreedtes zijn de objecten.

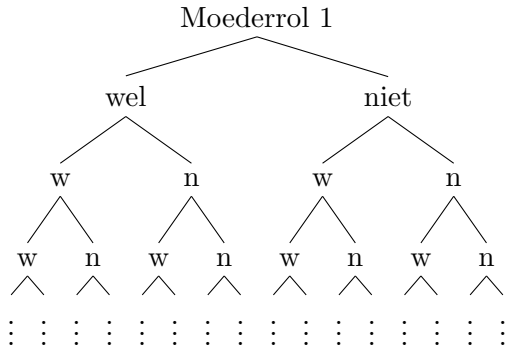
Het knapzakprobleem is *NP-volledig*. Dit betekent dat het probleem tot de klasse NP behoort en dat elk ander probleem in NP in polynomiale tijd tot dit probleem te reduceren is. De klasse NP bevat alle problemen die in polynomiale tijd te verifiëren zijn. Dit betekent dat in polynomiale tijd bepaald kan worden of een bepaalde waarde wel of geen oplossing is voor het probleem. Het hoeft niet zo te zijn dat er in polynomiale tijd een oplossing gevonden kan worden[1]. Omdat het knapzakprobleem NP-volledig is, is er geen betere oplossing dan de *brute force method*. Dit is een methode waarbij je alle mogelijke oplossingen afgaat. In dit geval zijn er 2^m mogelijkheden voor I, dus er zijn 2^m mogelijke oplossingen. Er is dus geen algoritme die in minder dan exponentiële tijd een goede oplossing geeft.

Vervolgens kijken we bij deelfase B naar meerdere moederrollen. Van een moederrol hebben we ook bepaalde gegevens nodig, zoals het gewicht, de breedte en materiaal. We moeten ermee rekening houden dat de minimale slitbreedte 30 mm is en dat het eerste mes van de messenbalk een standaardpositie heeft van 10 mm aan de linkerkant. Deze 10 mm is nodig omdat de zijkanten van de geslitte platen een scherpe rand moeten hebben. Verder moet er rekening meegehouden worden dat er tijdens het slitten en knippen materiaal verloren gaat (bijvoorbeeld doordat het materiaal enigzins gekreukt wordt bij het opstarten), hierdoor nemen we 10% van de benodigde hoeveelheid materiaal erbij. Het programma geeft de snijmogelijkheid met zo min mogelijk afval.

Daarna volgt deelfase C. Het programma van deelfase B wordt uitgebreid zodat het onbruikbare afval beperkt blijft. Offcuts zijn het onbruikbare gedeelte van de moederrol dat overblijft na het slitten. Bruikbare offcuts kunnen doorverkocht worden aan andere bedrijven. Een offcut van minimaal 50 mm is verkoopbaar. In deelfase B keken we nog naar het minimaliseren van het afval, in deelfase C kijken we naar het minimaliseren van de kosten en hebben we dus rekening gehouden met het mogelijk verkopen van bruikbare offcuts. Als al deze deelfasen geïmplementeerd zijn, is fase 1 afgerond.

Programma

Het programma dat we uiteindelijk in deelfase C hebben gemaakt (zie Bijlage A) is gebaseerd op de brute force method. De brute force method gaat alle mogelijke oplossingen af. Dit betekent dat het programma de oplossingen bekijkt waarbij orderbreedte i óf wel óf niet uit een bepaalde moederrol gesneden wordt. Dit kunnen we weergeven aan de hand van bomen, zie de figuur hieronder. We nummeren daarvoor de moederrollen en beginnen met moederrol 1. In de boom stelt elke hoogte een bepaalde orderbreedte voor. Deze kunnen we wel of niet uit de moederrol snijden. Deze keuze hebben we voor elke orderbreedte. Zo krijgen we een rij naar beneden met de keuzes wel en niet. Elke rij geeft een mogelijke oplossing weer. Na het bekijken van moederrol 1 doen we precies hetzelfde met moederrol 2, waarbij we rekening houden met de orderbreedtes die we al gesneden hadden uit moederrol 1. Dit kan beschouwd worden als een tweede boom die onder een oplossing van de eerste boom is geplakt. Dit kunnen we herhalen voor alle moederrollen.



Bij de uitleg van het programma verwijzen we naar regelnummers uit bijlage A. In het programma hebben we stucturen gemaakt voor orderbreedtes, moederrollen en zogeheten gesneden rollen, om de benodigde informatie voor deze dingen op te kunnen slaan. Zie hiervoor regels 1 t/m 21. Het hierboven beschreven algoritme is de functie `optimaliseer` uit regels 127 t/m 181. Hierbij is `io` de index van de te bekijken orderbreedte en `ib` de index van de te bekijken moederrol.

Als van alle orders genoeg gesneden is (regels 131 t/m 148) wordt gekeken of de nu behaalde oplossing beter is dan eerder gevonden oplossingen, zo ja, dan wordt deze oplossing opgeslagen in `best` (regels 141 t/m 145).

Als we niet klaar zijn maar wel de laatste orderbreedte bekeken hebben, gaan we door naar de volgende moederrol en beginnen we weer bij de eerste orderbreedte (regels 149 t/m 164). Als we ook alle moederrollen al hebben bekeken, blijkt dat de nu behaalde oplossing helemaal geen goede oplossing is en gaan we een tak terug in de boom (regels 129 en 130).

Als we niet klaar zijn en nog niet alle orderbreedtes bij de huidige moederrol bekeken hebben gaan we de oplossing met (regels 167 t/m 179) en zonder (regel 180) het snijden van deze orderbreedte uit deze moederrol uitrekenen (regels 165 t/m 181). Hierbij mag de orderbreedte er alleen uitgesneden

worden als hij qua breedte erin past, er nog gewicht nodig is van de orderbreedte en als er nog geen orderbreedtes van een andere order uit de moederrol zijn gesneden (regel 167). Wanneer we kijken naar de oplossing waarbij de orderbreedte er wel uit wordt gesneden, mag hij daarna best nog een keer eruit gesneden worden. Daarom wordt de functie `optimaliseer` aangeroepen met `io` en niet `io+1` (regels 173 en 175). In de functie `optimaliseer` worden steeds hulpfuncties gebruikt die te vinden zijn in regels 22 t/m 51.

In het bomenschema gaan we alle mogelijke oplossingen na, maar er zijn manieren om dat in te perken, waardoor we geen onnodig rekenwerk doen. In ons programma hebben we het aantal te berekenen mogelijkheden ingeperkt op drie manieren.

Ten eerste onthoudt het programma de mogelijke oplossing met het minste afval tot dan toe. Dit is handig, want zo kunnen we bij het berekenen van de volgende mogelijke oplossing nagaan of deze oplossing beter zal worden. Als bij het nagaan van deze mogelijkheid blijkt dat onderweg al meer afval wordt geproduceerd dan bij de voorgaande, dan is deze oplossing niet de beste. Dit betekent dat we niet verder hoeven te rekenen en dan kunnen we die tak van de boom afbreken (zie regels 129 en 130).

Ten tweede bekijken we alle orderbreedtes die zo breed zijn dat ze uitsluitend in hun eentje uit een moederrol gesneden kunnen worden. Zo'n orderbreedte snijden we uit de kleinste moederrol waar die inpast. Dit beperkt het aantal te berekenen mogelijkheden, want de gebruikte moederrollen voor deze orderbreedtes hoeven we niet meer te gebruiken in onze boomstructuur. De functie hierbij is `order_uitdunnen` (regel 87 t/m 112). Deze functie maakt weer gebruik van hulpfuncties die te vinden zijn in regels 52 t/m 86.

Ten derde kijken we steeds of er nog wel een orderbreedte is die in deze moederrol erbij past. Als dat niet zo is kunnen we al door naar het bekijken van de volgende moederrol (regels 149 t/m 164).

Herziening fase 1

Na het programmeren van fase 1 is gebleken dat het runnen van het programma met een reëel aantal moederrollen veel te lang duurt.

Zij o het aantal verschillende breedtes die voor de orders gesneden moeten worden en zij m het aantal beschikbare moederrollen. Het programma berekent dan voor elke moederrol en elke orderbreedte zowel de oplossing waarbij de orderbreedte wel uit die moederrol gesneden wordt als de oplossing waarbij dat niet gebeurt. Hierdoor krijg je een boomstructuur. Het aantal mogelijkheden dat het programma af moet gaan is daardoor $(2^o)^m$. Dit kunnen er zelfs meer zijn, want als we een orderbreedte uit een bepaalde moederrol snijden, zouden we die er nog een keer uit kunnen snijden (als er nog meer gewicht van die orderbreedte nodig is). Soms laten we het programma stoppen, waardoor hij een aantal oplossingen over slaat. Dit komt doordat hij de tak waar hij op dat moment is, niet meer verder naar beneden volgt. Als het programma stopt bij orderbreedte i en bij moederrol n, dan is het aantal mogelijke oplossingen dat wordt overgeslagen $(2^{o-i})^{m-n}$.

Als we het programma steeds laten uitgeven hoeveel oplossingen het al heeft behandeld, blijkt dat het programma ongeveer 10^9 oplossingen in 3 minuten bekijkt. Hierbij wordt rekening gehouden met afgekapte oplossingen. Dit komt neer op ongeveer $5,5 \cdot 10^6$ oplossingen per seconde. Bij een order van 17 breedtes en 58 moederrollen (wat een realitisch voorbeeld is) komt de totale tijd van het programma neer op:

$$\begin{aligned}
(2^{17})^{58}/5,5 \cdot 10^6 &\approx 6,5 \cdot 10^{296}/5,5 \cdot 10^6 \text{ seconden} \\
&\approx 1,2 \cdot 10^{290} \text{ seconden} \\
&\approx 3,3 \cdot 10^{286} \text{ uur} \\
&\approx 1,4 \cdot 10^{285} \text{ dagen} \\
&\approx 3,8 \cdot 10^{282} \text{ jaar}
\end{aligned}$$

Ter vergelijking: de aarde bestaat ongeveer $4,5 \cdot 10^9$ jaar.

Daarom hebben we gekeken naar manieren om het aantal mogelijkheden in te perken. Dit houdt in dat we hebben gezocht naar mogelijke takken van de boom die van te voren al geschrapt kunnen worden, net zoals de takken die geschrapt konden worden omdat ze een groter verlies geven dan al eerder bezochte oplossingen. Deze waren echter nauwelijks te vinden. De mogelijkheden die geschrapt konden worden waren niet voldoende om het programma snel genoeg te laten lopen. Daarom hebben we besloten het probleem anders aan te pakken, hierdoor moet er een ander programma komen.

Voor een andere oplossing hebben we twee mogelijkheden bekeken, namelijk het cutting stock problem en een variant op het knapzakprobleem. Eerst bekijken we het cutting stock problem. We zullen zien dat deze mogelijkheid niet toereikend is voor ons probleem.

Probleem 0.2 (Het cutting stock problem)**.** *Gegeven is een moedermateriaal (moederrol) met standaardafmetingen en stukken (orderbreedtes), met dezelfde lengte als de moederrol, die uit dit moedermateriaal gesneden moeten worden. Bepaal hoe de orders met zo min mogelijk materiaalverlies uit de moederrol gesneden kunnen worden.*

We kunnen dit op de volgende manier wiskundig bekijken. Zij I de verzameling met de verschillende orderbreedtes. Zij J de verzameling van de verschillende patronen die met deze breedtes kunnen worden gemaakt. Met een patroon bedoelen we een combinatie van orderbreedtes die samen in de breedte van de moederrol passen, dit is dan de messenstand. Zij nu a_{i_j} het aantal orderbreedtes met breedte i dat in patroon j gesneden wordt en zij b_i het aantal keer dat er een orderbreedte met breedte i nodig is. Noem nu x_j het aantal moederrollen dat gesneden wordt met patroon j. Dan kunnen we het probleem als volgt opschrijven:
Bepaal

$$\min\{\sum_{j\in J} x_j \mid \sum_{j\in J} a_{i_j}x_j \geq b_i \ \forall i \in I\}.$$

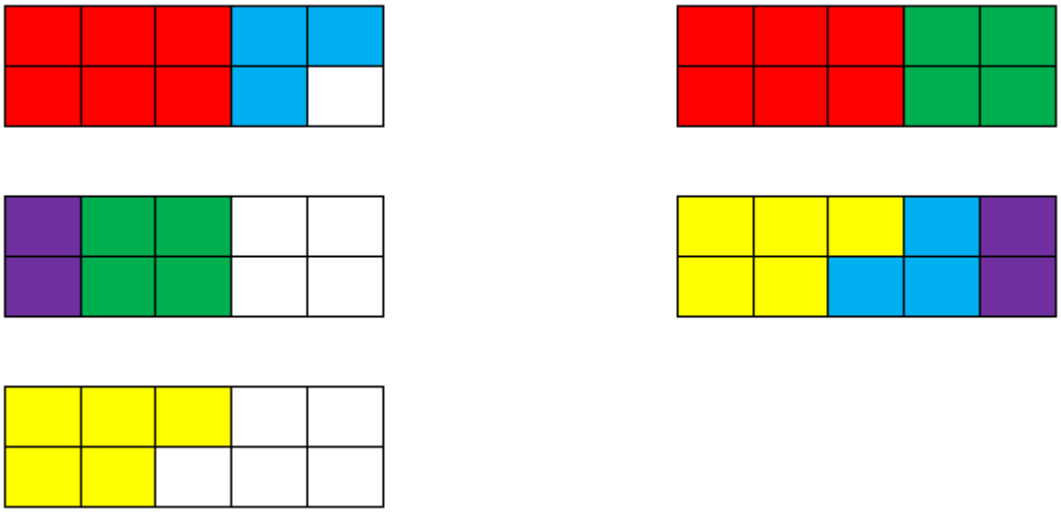
Hiermee wordt het minimum aantal benodigde moederrollen berekend.

Er zijn oplossingen voor het cutting stock problem bekend. Het is een 'geheeltallig' lineair programma. Voor weinig orderbreedtes is het op te lossen met een *standard branch-and-bound algorithm*. Het aantal mogelijke patronen neemt echter exponentieel toe als het aantal orders toeneemt. Hierdoor kan het probleem niet meer efficiënt worden opgelost met een standard branch-and-bound algorithm. Een methode die in zo'n geval gebruikt wordt heet *delayed column generation approach*. Hierbij kiezen we een verzameling patronen en passen daar het lineaire programma op toe [3].

Er zijn een aantal verschillen tussen het cutting stock problem en ons probleem. Eén daarvan is dat we te maken hebben met moederrollen die een verschillende lengte en een verschillende breedte hebben. In het cutting stock problem zijn de maten van het moedermateriaal standaardmaten. Verder hebben wij niet altijd de hele lengte van de moederrol nodig, maar meestal alleen maar een gedeelte. Hierdoor is het mogelijk om behalve verlies in de breedte, ook verlies in de lengte te hebben. Het probleem met het verschil in breedte tussen de verschillende moederrollen zou kunnen worden opgelost door het gemiddelde te nemen van alle breedtes van de moederrollen en dit gemiddelde te gebruiken als standaardbreedte van het moedermateriaal in het cutting stock problem. Als het probleem is opgelost kan daarna gecorrigeerd worden voor de fout die is gemaakt door het standaardiseren. Echter zitten we nu nog steeds met het probleem dat zowel de moederrollen als de orderbreedtes verschillen in lengte en dat daar vaak meer verlies op gemaakt wordt dan op de breedte. Het is dus zaak per orderbreedte een moederrol te vinden die ook qua lengte genoeg overeenkomt. Dit geeft de doorslag om niet met het cutting stock problem te proberen ons probleem op te lossen.

Een mogelijkheid die waarschijnlijk wel gaat werken is een variant die we hebben bedacht op het knapzakprobleem. De oplossing die we in fase 1 hadden bedacht voor onze probleemstelling is gebaseerd op de brute force method. Hierdoor werd de uitvoertijd van het programma veel te lang. Onze nieuwe oplossing voor fase 1 is gebaseerd op een andere aanpak van het knapzakprobleem. Voordat we dat toelichten bekijken we eerst het zogenaamde *bussenprobleem*. Dit is onze inspiratiebron voor de variant op het knapzakprobleem.

Probleem 0.3 (Het bussenprobleem)**.** *Gegeven is een aantal bushaltes met bij iedere bushalte een aantal kinderen. Deze kinderen moeten opgehaald worden met schoolbussen van dezelfde grootte. Bepaal een route langs de bushaltes zodat er zo min mogelijk bussen worden gebruikt. Hierbij moet rekening gehouden worden met het feit dat een groepje kinderen bij een bushalte in zijn geheel in één bus moet.*



Figuur 1: Links: groepjes kinderen niet gesorteerd. Rechts: groepjes wel gesorteerd.

Er is een bepaalde aanpak die ervoor zorgt dat we zo min mogelijk bussen gebruiken. We kunnen dit illustreren aan de hand van een voorbeeld. Stel dat we bussen met tien zitplaatsen hebben die langs vijf haltes moeten. Er staan groepjes van 6, 3, 2, 4 en 5 kinderen bij de haltes.

Als we de kinderen in deze volgorde in de bussen laten stappen hebben we drie bussen nodig, zoals links weergegeven in figuur 1. In de figuur zijn de bussen weergegeven door de rasters en elk hokje staat voor één zitplaats. We hebben drie bussen nodig, omdat we de groepjes kinderen niet mogen splitsen. Een betere oplossing verkrijgen we door de groepjes kinderen van groot naar klein te sorteren. Dit zorgt ervoor dat we aan het einde van de rit de kleine groepjes gemakkelijk kunnen verdelen over de bussen, want deze vullen gemakkelijk de gaten op. Hierdoor hebben we maar twee bussen nodig, zoals rechts weergegeven. Het idee van het sorteren van elementen gaan we gebruiken in de variant op het knapzakprobleem.

Nu komen we terug bij het knapzakprobleem. Zoals we in fase 1 zagen, is er geen algoritme dat in een redelijke tijd de beste oplossing geeft voor het knapzakprobleem. Dit zagen we ook aan de uitvoertijd van ons eerste programma. Er zijn wel algoritmes die de beste oplossing benaderen. Voorbeelden hiervan zijn *Greedy algoritmes*. Dit is een verzamelterm voor allerlei soorten algoritmes die lokaal de beste oplossing bepalen. Dit hoeft niet te betekenen dat je op deze manier globaal de beste oplossing krijgt, maar je kunt het wel benaderen. George Dantzig (1914-2005) heeft een Greedy algoritme voor het knapzakprobleem bedacht [2]. Sorteer hiervoor de objecten op waarde per gewicht (w/g) van groot naar klein. Ga nu per keer na of het volgende object qua gewicht nog in de knapzak past. Zo ja, stop het er dan in. Zo nee, bekijk het volgende object. Op deze manier verkrijg je bij elke stap de op dat moment beste mogelijkheid. Dus lokaal is dat de beste oplossing, maar globaal hoeft dat niet zo te zijn. We geven een voorbeeld:

Voorbeeld 0.4. Neem een knapzak met maximaal gewicht 6. Neem drie objecten met gewicht en waardes: $w_1 = 3$, $g_1 = 2$; $w_2 = 6$, $g_2 = 5$; en $w_3 = 3$, $g_3 = 5$. Dan zijn de ratios (w/g):

$$(w_1/g_1) = 1,5$$
$$(w_2/g_2) = 1,2$$
$$(w_3/g_3) = 0,6$$

Met het Greedy algoritme volgt nu dat we object 1 in de knapzak stoppen en dat er dan niks meer

bij past, want object 1 heeft gewicht 2 en de andere twee objecten hebben gewicht 5. We kunnen meteen zien dat dit niet de beste oplossing is, want we konden beter object 2 van waarde 6 in de knapzak doen. Een oplossing hiervoor is om te kijken of de waarde van de verkregen oplossing door het Greedy algoritme groter is dan de hoogste waarde die een object kan hebben. In dit geval is dat niet zo, dus doen we object 2 in de knapzak.

Deze techniek gaan we toepassen op ons probleem. In ons geval ziet de 'knapzak' er iets ingewikkelder uit. Alle moederrollen samen vormen de knapzak. De moederrollen verschillen in breedte en in lengte. Dus van een 1-dimensionale 'zak', waarbij we alleen een grens hadden voor het totale gewicht, gaan we naar een 2-dimensionale 'zak', waarbij we een grens hebben op de breedte én de lengte. Ook de objecten zijn iets ingewikkelder. In ons geval zijn de ordersbreedtes de objecten. Deze zijn, in tegenstelling tot de objecten in het gewone knapzakprobleem, te splitsen (niet qua breedte, maar wel qua lengte). Dit betekent dat een deel van de orderbreedtes uit de ene moederrol gesneden kan worden en een ander deel uit een andere moederrol. Ons doel is om zo min mogelijk afval te creeëren. Dit wordt dus een minimalisatieprobleem in plaats van een maximalisatieprobleem.

Programma

Met deze kennis kunnen we een concrete oplossing voor ons probleem geven. Door ons eerste programma voor fase 1 kwamen we erachter dat je het meeste verliest op de lengte van de moederrol. Om dit zo goed mogelijk te minimaliseren kijken we naar gewicht per breedte (g/b) van zowel de moederrollen als de orderbreedtes. Analoog aan het Greedy algoritme voor het knapzakprobleem sorteren we de rollen als volgt:

- sorteer eerst de moederrollen van klein naar groot ten opzichte van g/b;
- sorteer dan de moederrollen met hetzelfde g/b van klein naar groot ten opzichte de breedte b;
- sorteer de orderbreedtes van groot naar klein ten opzichte g/b;
- sorteer ten slotte de orderbreedtes met hetzelfde g/b van groot naar klein ten opzichte de breedte b.

Het programma voor onze nieuwe aanpak is te vinden in Bijlage B. Hierbij hebben we de structuren uit de eerste versie overgenomen en hier een paar extra eigenschappen aan toegevoegd (zie regels 1 t/m 26). Het sorteren doen we door gebruik te maken van het *quicksort algorithm* (regels 293 t/m 306) en die maakt op zijn beurt weer gebruik van het *DNF algorithm* (regels 268 t/m 292). De definities voor wanneer een orderbreedte/moederrol groter of kleiner is dan een andere orderbreedte/moederrol is terug te vinden in regels 27 t/m 50.

Als we dit gedaan hebben kunnen we aangeven wanneer we een orderbreedte uit welke moederrol snijden. We bekijken orderbreedte i met gewicht g_i en breedte b_i (regels 391 t/m 418). Het idee is als volgt:

1. Als g_i/b_i groter is dan g/b van de laatste moederrol waar hij qua breedte in zou passen, dan kan deze orderbreedte niet in zijn geheel uit één moederrol gesneden worden. Snijd orderbreedte i dan uit de eerste moederrol waarbij meer dan 40% van het gewicht gesneden wordt. Is er geen enkele moederrol waar minstens 40% uit gesneden kan worden, snijd dan orderbreedte i uit de laatste moederrol waar hij qua breedte inpast. Voeg het deel wat overblijft van de orderbreedte weer toe aan de lijst van orderbreedtes en ga door naar de volgende orderbreedte uit de lijst (regels 394 t/m 404).
2. Als g_i/b_i níet groter is dan g/b van de laatste moederrol, snijd de orderbreedte dan uit de eerste moederrol waar die in zijn geheel in past (regels 407 t/m 418).

Op deze manier kan het gebeuren dat we een moederrol voor minder dan 50% van de breedte gebruiken, terwijl het voordeliger was om een kleinere moederrol te pakken. Dit komt doordat we

de moederrollen van hetzelfde g/b sorteren van groot naar klein ten opzichte van de breedte (dit heeft als voordeel om zoveel mogelijk orderbreedtes uit één moederrol te snijden). Een andere reden waarom er moederrollen voor minder dan 50% gebruikt worden is dat we lokaal de beste oplossing nemen, waardoor het kan voorkomen dat we voor elke orderbreedte een nieuwe moederrol pakken, terwijl het mogelijk is om meerdere orderbreedtes bij elkaar in één moederrol te stoppen. Dan zouden we wel meer verliezen op de lengte, maar dan gebruiken we minder moederrollen.

Om te voorkomen dat we een te brede moederrol pakken, runnen we het programma opnieuw in het geval dat er een moederrol is waarvan minder dan 50% van de breedte gebruikt wordt, waarbij we deze moederrol buiten beschouwing laten (regels 431 t/m 443). Op deze manier wordt het aantal gebruikte moederrollen kleiner. Dit proces herhalen we totdat er een oplossing is waarbij er geen moederrol meer voor minder dan 50% gebruikt wordt. Als er niet zo'n oplossing bestaat, nemen we de tot dan toe beste oplossing. Bovendien onthouden we steeds de tot dan toe beste oplossing, want een moederrol buiten beschouwing laten zou ook juist nadelig kunnen zijn. Op deze manier maken we de oplossing van onze variant van het Greedy algoritme globaal beter. Dit betekent niet dat we de meest optimale oplossing verkrijgen.

Het algehele algoritme samen geeft de functie `slitten` (uit regels 387 t/m 449). Deze functie gebruikt hulpfuncties die te vinden zijn in regels 307 t/m 386.

Het programma dat we op deze manier hebben opgebouwd zal sneller een oplossing vinden dan het oude programma, omdat we nu niet alle mogelijkheden meer afgaan. Met de bestanden die we van Royal Smit hebben gekregen, doet het programma er ongeveer 2 seconden over.

Om te kijken hoe goed dit programma de optimale oplossing benadert, willen we het vergelijken met het oude programma. Voor moeilijke voorbeelden met veel orderbreedtes is de uitvoertijd van het oude programma te lang, waardoor we dit niet als vergelijkingsmateriaal kunnen gebruiken. We zijn wel eenvoudige voorbeelden nagegaan. Hierbij geven beide programma's hetzelfde uit. Dit betekent dat in die gevallen het nieuwe programma de optimale oplossing geeft.

Fase 2

Fase 2 bestaat uit twee delen. Ten eerste gaan we het programma uit fase 1 verbeteren en uitbreiden, en ten tweede gaan we een ander programma maken dat een inkoopadvies geeft voor de moederrollen.

Verbetering fase 1

Om het programma te verbeteren zullen een aantal wijzigingen in het programma (zie bijlage B) geïmplementeerd worden, waarbij de uitvoer niet verandert. Ten eerste moeten er meerdere orders tegelijk berekend worden, waarbij er rekening gehouden moet worden dat er geen breedtes van verschillende orders uit één moederrol gesneden mogen worden. Hiervoor is een extra conditie in regel 414 toegevoegd.

Ten tweede gaat het programma berekenen hoe lang het slitproces gaat duren. Hieruit kan Royal Smit zelf een planning maken, waarbij ze rekening houden met de leverdatum. Om uit te rekenen hoe lang het slitproces duurt moeten we ook rekening houden met het verstellen van de messen. De stand van de messen hangt af van de breedtes die gesneden moeten worden én van de dikte van de rol. De ombouw van de machine duurt 1,5 uur. Het slitten van de rol zelf duurt 10 minuten. Het liefst heeft Royal Smit dat alles kort voor de levertijd geslit is, ze hebben liever niet tien weken voor de levertijd alles geslit in het magazijn liggen. Hiervoor is de functie `print_tijden` uit regels 99 t/m 127 gemaakt.

Ten slotte moeten we rekening houden met de kwaliteit van het materiaal. Dit is niet zo'n probleem om toe te voegen, want dit hadden we al in onze eerste versie van fase 1. Het idee is dat de klant doorgeeft welke kwaliteit van het materiaal van de moederrollen wordt vereist, wat vervolgens kan worden aangegeven in het programma. Bij het inlezen van de orders (regels 212 t/m 267) worden deze per materiaal in een vector gestopt en daarna wordt voor elk gevonden materiaal de beschikbare moederrolen ingelezen (regels 173 t/m 211). Vervolgens wordt in de `main` functie per materiaal de oplossing berekend en worden deze oplossingen samengevoegd (regels 469 t/m 497). Daarnaast laten we het programma het oorspronkelijke gewicht van een moederrol en het percentage dat uit de moederrol gesneden is uitgeven. Hiervoor zijn er wat regels bijgekomen in de functie `print_oplossing` uit regels 128 t/m 172.

Inkoopadvies

Naast deze aanpassingen zullen we een ander programma maken dat een inkoopadvies geeft. Dit programma doet precies het tegenovergestelde als hetgeen we in fase 1 hebben bekeken. Het doel is om te kijken of het voordeliger is om moederrollen op maat in te kopen. De prijs van de moederrollen gaat hierdoor omhoog, maar de hoeveelheid afval kan wel gereduceerd worden.

De bedoeling is om, gegeven een aantal orders, de beste moederrollen in te kopen en daarmee zo min mogelijk afval te hebben. De moederrollen zijn te verkrijgen tussen de 960 mm en 1050 mm met tussenstappen van 5 mm. Daarnaast kun je aangeven welk gewicht de moederrol moet hebben. Dit kan tussen de 0 kg en 5000 kg met tussenstappen van 1 kg. Ook is het mogelijk om een gewicht aan te geven van een aantal rollen samen. We hebben nagedacht over hoe we de moederrollen het beste kunnen indelen, hiervoor hebben we verschillende voorbeelden behandeld. In deze voorbeelden hebben we alleen gekeken naar de orderbreedte en niet naar het gewicht van de rol. Het is helaas niet gelukt om hier een programma voor te schrijven, dit is slechts een idee.

Voorbeeld 0.5. We bekijken een order met de volgende orderbreedtes: 900 mm, 650 mm, 370 mm en 250 mm. Nu moeten we deze breedtes in de moederrollen stoppen en daarbij zo min mogelijk afval creëren. De meest voor de hand liggende manier is om de orderbreedtes van de breedste naar de minst brede te ordenen. We stoppen nu de eerste orderbreedte in een moederrol, daarna gaan we de rij af en kijken of er nog een orderbreedte bij past. Op deze manier stoppen we elke keer de grootst mogelijke orderbreedte erbij. We kunnen moederrollen kopen met maximale breedte 1050 mm (inclusief de 10 mm aan de randen). In ons voorbeeld geeft dit het volgende:

1. In de eerste moederrol stoppen we breedte 900 mm, dan gaan we de rij naar beneden af. Er is geen breedte die we bij deze 900 mm kunnen stoppen zonder de 1050 mm te overschrijden. We stoppen deze 900 mm in een moederrol van 960 mm en gaan door naar de volgende moederrol.
   afval: $960 - 920 = 60$ mm (exclusief randen)

2. In de tweede moederrol stoppen we breedte 650 mm, dan gaan we de rij naar beneden af. De volgende breedte die we tegenkomen is 370 mm en die kunnen we bij deze 650 mm stoppen. Dit geeft een breedte van 1020 mm. Samen met de randen is er dus een rol van 1040 mm nodig.
   afval: $1040 - 1040 = 0$ mm (exclusief randen)

3. In de derde moederrol stoppen we breedte 250 mm, deze is namelijk als enige over. We moeten hiervoor een moederrol van 960 mm gebruiken.
   afval: $960 - 270 = 690$ mm (exclusief randen)

De totale hoeveelheid afval is nu 750 mm. Om dit te reduceren hebben we een kleine verandering toegepast. We gaan nu de orderbreedtes niet meer van de breedste naar de minst brede ordenen, maar we splitsen de lijst in tweeën. We ordenen de breedtes van groter dan 515 mm van groot naar klein en de breedtes van kleiner dan 515 mm van klein naar groot. In ons voorbeeld geeft dit de ordening: 900 mm, 650 mm, 250 mm en 370 mm. Nu passen we hetzelfde principe toe als net.

1. In de eerste moederrol stoppen we breedte 900 mm, dan gaan we de rij naar beneden af. Er is geen breedte die we bij deze 900 mm kunnen stoppen zonder de 1050 mm te overschrijden. We stoppen deze 900 mm in een moederrol van 960 mm en gaan door naar de volgende moederrol.
   afval: $960 - 920 = 60$ mm (exclusief randen)

2. In de tweede moederrol stoppen we breedte 650 mm, dan gaan we de rij naar beneden af. De volgende breedte die we tegenkomen is 250 mm en die kunnen we bij deze 650 mm stoppen. Dit geeft een breedte van 900 mm. Hier kunnen we niks meer bij stoppen en er is dus een rol van 960 mm nodig.
   afval: $960 - 920 = 40$ mm (exclusief randen)

3. In de derde moederrol stoppen we breedte 370 mm, deze is namelijk als enige over. We moeten hiervoor een moederrol van 960 mm gebruiken.
   afval: $960 - 390 = 570$ mm (exclusief randen)

De totale hoeveelheid afval is nu gereduceerd tot 670 mm. Dit komt omdat we in het eerste voorbeeld een rol nodig hadden van meer dan 960 mm, in het tweede voorbeeld konden we deze breedte in een andere moederrol stoppen waardoor de uitkwamen op alleen maar rollen van 960 mm en dus minder afval.

Bovenstaand idee heeft echter nog een aantal verbetermogelijkheden. Wij komen er echter niet aan toe om deze verbeteringen door te voeren.

Verbeteringen project

Het afmaken van het project is een mooie opdracht voor het vak 'Modellenpracticum' tijdens een volgend studiejaar. Dit hebben we ook aangegeven bij het bedrijf. We hebben nog een aantal verbeterpunten, welke we op zullen splitsen in een organisatorisch en een inhoudelijk deel. We sluiten af met wat tips die uit ons samenwerkingsverband nuttig bleken.

Organisatorisch

- In het begin van het project spraken we niet zoveel met onze begeleider. Later deden we dit wel en stelden we hem inhoudelijke vragen, dit is erg nuttig.
- Wij kregen helaas pas laat de contactgegevens van het bedrijf, waardoor we pas laat konden beginnen.

Inhoudelijk

- Voor het inkoopadvies hebben wij de orderbreedtes gesorteerd op breedte. In het programma van fase 1 is juist op gewicht/breedte gesorteerd, zodat minder afval op de lengte van een moederrol wordt verkregen. Het gewicht van een orderbreedte is bij het inkoopadvies compleet buiten beschouwing gelaten. Wanneer het gewicht van een orderbreedte ook wordt meegenomen, kan wellicht bespaard worden op afval van de lengte van een moederrol.
- Wij hebben in ons programma niet meegenomen dat eventuele orderbreedtes achter elkaar uit de moederrol geslit kunnen worden.

Tips

- Onderhoud wekelijks contact met elkaar en leg uit wat iedereen gedaan heeft.
- Zorg dat er meerdere mensen in het groepje zitten met programmeerervaring en zorg dat minstens twee personen precies weten hoe het programma in elkaar steekt.

Evaluatie

We hebben het project bij Royal Smit eind juni afgerond. Daarbij hebben we fase 1 afgerond en we hebben een idee bedacht voor het inkoopadvies van fase 2. Het verbeteren en uitvoeren van dit idee en het uitvoeren van fase 3 is een mooi project voor volgend jaar.

In mei hadden we een programma voor fase 1 gemaakt. Echter, het bleek dat dit programma er veel te lang over deed om een oplossing te geven, zelfs langer dan de aarde bestaat! Dat betekende dat we een compleet nieuw idee voor een nieuw programma moesten bedenken. Het punt is dat we te maken hebben met een tweedimensionaal probleem en dat is niet zomaar op te lossen. In ons project is veel tijd gaan zitten in het eerste programma, dus dat is jammer. Dit hadden we van tevoren niet kunnen voorzien. Maar we hebben het wel voor elkaar gekregen om een nieuw werkend programma te maken. Dit programma berekent niet de beste oplossing, maar geeft een goede oplossing voor het probleem.

Toen we ons project toegewezen kregen, duurde het even voordat we de contactgegevens van het bedrijf hebben gekregen. Als we deze gegevens sneller gekregen hadden, konden we eerder aan de slag. We hebben meteen een afspraak gemaakt met Royal Smit, waarna het project van start ging. We kwamen als groepje elke week een keer bij elkaar om door te spreken hoe het ging, wat we nog moesten doen en wie dat dan ging doen. Deze samenwerking is prima verlopen. De taakverdeling was vrij eenzijdig, aangezien Anita in ons groepje het beste kan programmeren. Dat betekende dat zij vaak dit werk op zich heeft genomen.

Tijdens onze eindpresentatie hebben we ons programma gepresenteerd. Royal Smit was erg tevreden met het werkende programma voor fase 1. Het contact met het bedrijf is gedurende de periode van het project goed verlopen.

Bibliografie

[1] Bosma. W., *Cryptografie*, (najaar 2014).

[2] Onbekend, *Knapsack problem*, (z.d.). Geraadpleegd op 13-06-2016 van https://en.wikipedia.org/wiki/Knapsack_problem#Greedy_approximation_algorithm.

[3] Onbekend, *The cutting stock problem*, (2016). Geraadpleegd op 09-06-2016 van http://www.neos-guide.org/content/cutting-stock-problem.

Bijlage C: Logboek

24 februari

Het heeft een tijd geduurd voordat we de contactgegevens kregen van Smit Transformatoren. Vandaag hebben we de contactgegevens gekregen en we hebben gelijk een afspraak kunnen maken met het bedrijf: donderdag 3 maart om 14.00u. We gaan met z'n allen naar het bedrijf, samen met onze begeleider Wieb Bosma. Om ons voor te bereiden voor het bezoek hebben we de volgende vragen opgesteld. We hebben ze verdeeld onder twee kopjes: vragen van organisatorische aard en inhoudelijke vragen.

Organisatorisch:

1. Wat verwachten ze van ons? Willen ze een goed werkend programma (in welke taal) of moeten wij een programma aanpassen? Of gaat het alleen om een idee?

2. Wat zijn de verwachtingen naar ons toe? Hoe vaak moeten we contact houden? Feedback geven en verwachten ze een eindpresentatie?

3. Welke methodes hanteren jullie momenteel?

4. Wat moeten wij verbeteren aan het huidige programma?

5. Willen zij zelf het programma nog aan kunnen passen?

Inhoudelijk:

1. Welke afmetingen hebben de platen?

2. Hebben de verschillende platen ook verschillende diktes?

3. Hebben de platen snijranden en moeten we rekening houden met afwijkingen?

4. Kan het restafval bewaard worden? En kunnen hier opnieuw nieuwe platen uitgesneden worden?

5. Snijden ze de platen in een keer door? Zijn het stroken of kunnen er vormen gesneden worden?

6. Welke breedtes kunnen doorverkocht worden?

7. Welke maten hebben de verschillende rollen?

8. Hoeveel verschillende bestellingen? Zijn de bestellingen hetzelfde/overeenkomstig qua maten?

9. Wat zijn de kosten van dergelijke platen/snijmachines en restafval?

10. Hoe werkt de machine? (Moeten we bepaalde afmetingen eerst snijden?)

3 maart

Vandaag zijn we naar het bedrijf geweest. Onze contactpersoon, Mark Jansen, heeft ons samen met een aantal medewerkers een rondleiding gegeven door de fabriek. We hebben het proces van de bouw van een transformator van achter naar voren gezien. We begonnen bij het stapelen van de platen en gingen daarna naar het knippen van de juiste lengtes. Deze onderdelen van het proces zijn niet van belang voor ons project, maar het geeft wel een goed beeld van het werkveld. Hetgeen wat wel van belang is voor ons project is het zogenaamde 'slitproces'. Dit proces bekeken we in het tweede gedeelte van de rondleiding. Hierbij gaat het om het snijden van de geleverde rollen staal in de juiste breedtes.

Na de rondleiding gaf Mark Jansen ons een presentatie over hoe zij het project voor zich zien en wat er van ons verwacht wordt. De Powerpoint presentatie hebben we ook via de mail ontvangen.

De opdracht is dat er een programma geschreven moet worden om zo efficiënt mogelijk met het materiaal om te gaan. Het is een minimalisatieprobleem. Het komt erop neer dat alles wat we kunnen doen mooi meegenomen is. Nu kiezen ze namelijk de moederrollen, zonder een minimalisatieprogramma. De probleemstelling is opgedeeld in drie fasen. Deze worden in het verslag uitgelegd. Tijdens de presentatie hebben we de vragen gesteld die we hadden opgesteld (zie 24 februari). De gegevens die we vroegen in de inhoudelijke vragen staan voor een deel in de Powerpoint en voor een deel worden deze nog opgestuurd naar ons.

De drie fasen zijn een inhoudelijke opdeling van het project. Qua werkwijze zijn we redelijk vrij gelaten. Het is mogelijk om op locatie te werken zodat we dan dicht bij het productieproces aanwezig zijn, maar dat is niet noodzakelijk. De tijdsplanning kunnen we zelf bepalen. Na elke fase verwachten ze een korte presentatie over de vorderingen van ons project. Dat zijn in totaal 3 presentaties waarbij steeds een tweetal van ons presenteert.

8 maart

Vandaag is een taakverdeling gemaakt voor het hele project. We proberen in Matlab te programmeren, mocht dit niet lukken dan gaan we over tot C++.

Met z'n allen: het probleem wordt uitgedacht, welke zaken belangrijk zijn en hoe we dit gaan aanpakken. Wat we hier hebben besproken, werken de programmeurs uit.
Daja: maken van het programma
Anita: maken van het programma
Sterre: maken van het programma, verslag controleren
Iris: problemen vastleggen en dit opschrijven in het verslag
Simone: problemen vastleggen en dit opschrijven in het verslag
Rens: contactpersoon

Daarna zijn de volgende deadlines voor fase 1, 'proof of principle', voor onszelf opgesteld.
Deadline programma fase 1: 1 april
Deadline presentatie fase 1: 7 april
Ook hebben we een uiterste deadline opgesteld, waarbij we in ieder geval fase 2 af willen hebben.
Deadline gehele project: 20 juni.
De deadline voor fase 3 is niet gepland.

9 maart

Rens heeft via de mail de Powerpoint ontvangen, maar nog geen lijst met gegevens.
Deze voorraadgegevens van de moederrollen met een lijst met benodigde materiaal moeten we nog ontvangen. Het programma moet het materiaal opsplitsen in kwaliteit en breedte, en moet dit onthouden. Enkele eisen zijn de minimale slitbreedte (30 mm), zo min mogelijk afval en een courante offcut moet minimaal 50 mm zijn.

Het probleem wordt opgesplitst in meerdere stappen. Voor de deelfasen, zie verslag. In fase 1 besluiten we ook het artikelnummer uit te geven, wat eigenlijk onder fase 2 zou vallen.

Het zal een recursief programma worden, wat voorlopig nog kijkt naar één enkele order. Later zal er natuurlijk gekeken worden naar meerdere orders.

Wat we nu nodig hebben is de Excel sheet met gegevens van de rollen, de groottes van orders, welk afval bruikbaar is. Omdat het begin en het eind van elke rol niet bruikbaar is, moet er meer kilo metaal gesneden worden (10%) dan daadwerkelijk nodig is voor de order. Ook willen we graag weten of andere bedrijven wel de rollen op de snijmachine kunnen leggen die bij Royal Smit niet meer zouden kunnen passen.

16 maart

Matlab bleek lastig te zijn, dus het programma is geschreven in C++.
Het is niet helemaal duidelijk wat het grootste verschil is tussen fase 2 en fase 3. We denken dat het verschil is dat fase 2 inkoopadvies levert van moederrollen en dat fase 3 ook kijkt naar voorgeslitte rollen. Ook is het niet duidelijk of de offcuts doorverkocht worden met of zonder de goede rand. Daarnaast kijkt het programma nu naar zo min mogelijk afval en niet naar zo min mogelijk verlies. Of dit goed is, is niet helemaal duidelijk. In fase 2 zal ook rekening gehouden worden met een start- en leverdatum. Dit zal nog gevraagd worden.
Anita legt uit hoe het programma werkt. Het is een recursief programma en het werkt met moederrollen van verschillende breedtes. Het programma bekijkt nu één order met verschillende breedtes.

22 maart

Het is tijd voor de tussenpresentatie van fase 1. Iris en Simone leggen kort uit wat het programma doet, waarna er wat vragen komen vanuit Royal Smit. Ze zijn blij verrast dat het programma nu al zo goed werkt.

23 maart

Om het probleem dat een deel van de rol niet bruikbaar is voor de transformator op te lossen, wordt besloten 10 % bij de order op te tellen.
Rens en Simone geven nog een update over hun bezoek aan Royal Smit. Gedurende zeven dagen per week wordt er 40 ton per dag geslit. Een week voor levertijd moet alles klaar staan, een weekvoorraad bestaat uit 100 tot 200 ton. Het bedrijf wil graag eens in de vier weken het programma runnen.

Er wordt een nieuwe taakverdeling gemaakt. Anita zal eerst de courante offcuts in het programma implementeren en het oorspronkelijke gewicht van de moederrollen in de output zetten. Daja zal kijken hoe de gegevens uit excel naar C++ ingelezen kunnen worden. Zodra de gegevens binnen

zijn, zullen Iris, Simone en Sterre kijken hoe de berekeningen efficiënter kunnen naar aanleiding van een keuze van een moederrol. Anita zal dit dan in het programma implementeren.

13 april

Simone en Sterre hebben fase 1 aangepast, ze vertellen even wat ze hebben gedaan. Ze hadden nog wat vragen, ze vroegen zich af of je een stuk opnieuw gebruikt of verkoopt als je het overhoudt. Nu berekent het programma de minste kosten, met verkoop offcuts voor 80 cent per kilo. Anita heeft nog aan het programma gewerkt, nu berekent het programma de minste kostenverlies. Courante offcut is 2,70-0,80 verlies.

De volgende stap wordt de kwaliteit. Voor de kwaliteit zijn goede bestanden nodig om in te lezen in het programma. Rens mailt naar het bedrijf voor een ovezicht van moederrollen en orders. Meerdere voorraden moederrollen, elke voorraad heeft een andere kwaliteit. Eerste stap: meerdere orders. Daarnaast willen we een .txt naar excel-bestand kunnen omzetten en andersom. Hier zijn we nog mee bezig, maar dit lijkt opgelost te kunnen worden.

Dan is er nog de order finish. In het programma is het lastig om data te geven (feestdagen, vakantie), wij denken dat handmatig plannen makkelijker is. Dan geeft programma uit hoeveel dagen een order nodig heeft om geslit te worden en daarbij een einddatum, zodat het bedrijf zelf kan indelen wanneer voor welke order geslit wordt.

Voor nu hebben we het volgende nodig: informatie over het inkoopadvies van fase 2. Wat kunnen we inkopen? Voor deze informatie gaan we naar het bedrijf toe. Iris en Daja kijken of programma wat Anita gemaakt heeft goed werkt.

20 april

Het probleem om van een .txt over te schakelen naar een excel-bestand en weer terug is opgelost. Daarnaast is de kwaliteit ook opgelost.

Rens en Simone zijn naar het bedrijf gegaan om wat zaken door te spreken. Daarbij is de order finish aan de orde gekomen. Wij hebben verteld dat de order finish met een programma niet erg handig is, dit kan beter handmatig gedaan worden. Hier was Torsten het mee eens, dus dit hebben we zo gelaten. Daarnaast hebben we een aantal vragen gesteld over het inkoopadvies. Er is ons hierover verteld dat de bedoeling is om vanuit het model een advies te geven over de te inkopen moederrollen. Echter, Torsten wist niet welke keuze we zouden hebben in het aangeven van een bepaalde breedte moederrol. Daarom is voor het inkoopadvies vanuit het model nog niet helemaal duidelijk hoe dat aangepakt gaat worden.

Maar, eerst hebben we nog te maken met een ander probleem. Namelijk dat het programma van fase 1 er veel te lang over doet om een werkbare oplossing uit te geven. Dat betekent dat we moeten nadenken over andere manieren om naar het probleem te kijken. Daarvoor hebben we nu twee opties die we beter gaan bekijken.

26 april

Er zijn groepjes gemaakt om de twee verschillende mogelijke oplossingen eens wat beter te bekijken om daarna met z'n allen te beslissen met welke mogelijke oplossing we verder gaan werken. De twee

opties zijn het cutting stock problem en het bussenprobleem. Iris, Anita en Simone gaan verder kijken naar het bussenprobleem en Daja, Rens en Sterre gaan onderzoek doen naar het cutting stock problem. Wanneer we deze problemen wat beter hebben bekeken gaan we weer in overleg om te kijken welke nu het beste is om mee verder te werken.

17 mei

Beide groepjes hebben elk naar een mogelijke oplossing gekeken in de afgelopen week. Daar is uitgekomen dat het cutting stock problem voor ons probleem niet dusdanig van toepassing is dat het handig is om te gebruiken. Daarom gaan we vanaf nu verder werken met het bussenprobleem. Sterre en Anita gaan kijken hoe het programma aangepast moet worden om het bussenprobleem te gebruiken. Rens houdt contact met het bedrijf, hij informeert ze over de huidige stand van zaken.

25 mei

Sterre en Anita hebben naar het programma gekeken en Anita heeft het voor elkaar gekregen om de betere aanpak van het busprobleem in het programma te stoppen. Daarbij is het programma getest en het werkt. Het kwam toen wel naar voren dat er een fout zat in het oude programma, aangezien deze een slechtere oplossing uitgaf dan het nieuwe programma, wat onmogelijk is. Anita gaat proberen om deze fout nog te vinden, alhoewel aan het nieuwe programma werken nu wel de voorkeur heeft. Als we het programma testen komt er een error, dus daar moet nog naar gekeken worden.

We hebben een datum geprikt om een presentatie te geven aan het bedrijf over de vorderingen van het project. Vervolgens hebben we een afspraak gemaakt, 1 juni gaan we naar het bedrijf toe. Sterre en Rens geven de presentatie, dus die moet ook nog voorbereid worden. Daarnaast schrijft Iris een stukje over hoe we het bussenprobleem toepassen in onze situatie, Daja verwerkt in het verslag waarom het cutting stock problem niet werkt en Simone houdt het logboek bij.

31 mei

Sterre en Rens hebben de presentatie in elkaar gezet, die we vandaag gaan bespreken. Hier en daar wordt het een en ander aangepast en het voorbeeld wordt veranderd. Anita heeft het nieuwe programma werkend gekregen. De volgende stappen voor het programma zijn: tijdsindicatie weergeven in de output, maximaal een order in een moederrol, oplossing uitgeven in een excel-bestand. We hebben besloten om de tijdsindicatie slechts weer te geven in de output als de tijd die een order nodig heeft om geslit te worden. We weten al hoe we de oplossing uit moeten geven in een excel-bestand, dit moet alleen nog aan het programma toegevoegd worden. Naast deze aanpassingen aan het nieuwe programma wordt gekeken of de fout uit het oude programma gevonden kan worden.

1 juni

Vandaag geven Sterre en Rens een presentatie bij het bedrijf over de vorderingen. Daarbij vertellen ze eerst iets over het oude programma, waarom dat niet meer werkt enzovoorts. Vervolgens leggen ze het bussenprobleem uit en daarna de betere aanpak van het bussenprobleem welke wij gaan toepassen in het programma. Dit doen ze aan de hand van een voorbeeld. Daarna leggen ze uit welke methode we nu eigenlijk in het programma toepassen. Ook wordt verteld wat ons plan is de komende week. Fase drie gaan we niet meer halen, fase twee is nog de vraag. Daar willen we ook nog wat informatie over krijgen. Wel gaan we nog de tijdsindicatie weergeven in de output van het programma en we zorgen ervoor dat maximaal één order in een moederrol wordt gestopt. Na de

presentatie is er de mogelijkheid om vragen te stellen.

Mark Jansen was zeer tevreden over het werk wat we tot nu toe hebben geleverd. We gaan het programma verbeteren op een paar punten en we gaan ons bezig houden met het inkoopadvies. Het idee hiervan is voor ons duidelijker geworden na dit bezoek. Het is de bedoeling dat we gegeven een aantal orders de geschikte moederrollen inkopen.

De planning voor deze week is dat iedereen het verslag leest en controleert. Daarnaast gaat Anita werken aan het programma en de rest denkt alvast na over een oplossing voor het inkoopadvies.

8 juni

We hebben afgesproken om ieder voor zichzelf voor vandaag het verslag wat we tot nu toe hebben gemaakt door door te lezen. Vandaag hebben we alle opmerkingen die bij het verslag zijn gezet besproken. Vervolgens is bekeken wat nog verbeterd moet worden in het verslag. Dit is onderling verdeeld, om de komende week aan te passen of toe te voegen.

13 juni

We hebben vandaag afgesproken om na te denken over het inkoopadvies van fase 2. Onze begeleider Wieb Bosma is hierbij om mee te denken over dit probleem en daarnaast hebben we hem geinformeerd over de stand van zaken.

Daarnaast hebben we nog een stuk verslag doorgekeken en Anita heeft laten zien wat het programma uitgeeft na het runnen van het programma. Daar hebben we nog over gesproken, wat daarbij beter moet enzovoorts. We hebben ook een plan voor het inkoopadvis van fase 2. We bekijken namelijk de orderbreedtes en sorteren de orderbreedtes groter dan 515 mm van klein naar groot en sorteren de orderbreedtes kleiner dan 515 mm van groot naar klein en zetten deze onder elkaar in bovengenoemde volgorde. Vervolgens passen we op het lijstje orderbreedtes het busprobleem toe. Als dit gedaan is nemen we per bus de best bijpassende moederrol.

15 juni

Tijdens deze afspraak gaan we het verslag doorlopen, met name de toegevoegde stukken. Vervolgens doen we nodige aanpassingen in het verslag.

23 juni

Vandaag is het tijd voor de eindpresentatie. Anita en Daja vertellen wat er verwerkt is in het programma sinds de presentatie van 1 juni. Ook geven ze aan dat er nagedacht is over fase twee van het project, maar dat dit niet geïmplementeerd is. Ze lichten dit toe met een voorbeeld. Na de presentatie is er weer de mogelijkheid om vragen te stellen.

Mark Jansen en zijn collega's zijn tevreden over het werk wat tot nu toe geleverd is. Ze vinden dat de excel sheet er overzichtelijk uitziet en zijn blij dat fase 1 geheel afgerond is. Er wordt afgesproken om op donderdag 30 juni om 12:00 uur langs te komen bij het bedrijf voor de overdracht van het programma.

29 juni

Voor vandaag heeft ieder voor zich aan de stukjes voor het verslag gewerkt. We komen bijeen om het hele verslag door te spreken. We lezen de nieuw toegevoegde stukken door en doen eventuele aanpassingen. Wanneer dit af is, kunnen we het verslag inleveren.

30 juni

Met het bedrijf is afgesproken dat we het programma om 12:00 over gaan dragen. Hierbij wordt CodeBlocks bij Royal Smit geïnstalleerd. Daarnaast dragen we het programma en de bijbehorende code over en leggen uit hoe het programma moet worden gebruikt. Daarnaast overhandigen we het bijbehorende verslag.