Aanpassen van personeelsdiensten bij buitendienststellingen

Algoritme voor het "pricing problem" in een kolomgeneratie aanpak
NS reizigers - VU Amsterdam

Vincent Vijfvinkel

Voorwoord

Van september 2004 tot en met maart 2005 heb ik voor mijn studie Bedrijfswiskunde en Informatica aan de Vrije Universiteit te Amsterdam stage gelopen bij de Nederlandse Spoorwegen N.V. te Utrecht. Bij de afdeling logistiek van NS Reizigers heb ik onderzoek verricht naar het aanpassen van diensten van machinisten. Voor u heeft u het bijbehorende eindverslag.

Dit eindverslag is niet alleen het resultaat van mijn onderzoek maar ook van hulp en begeleiding die ik kreeg bij het onderzoek van Dennis Huisman, vanuit de NS, en van Sandjai Bhulai, vanuit de VU. Beide willen ik dan ook hierbij bedanken. Samen hebben de collegas van de NS mij naast een leuke werkplek ook advies, tips en suggesties gegeven, hiervoor wil ik ook zeggen: Koszonom!

Juli 2005
Vincent Vijfvinkel

Management uittreksel

Het onderzoek naar het aanpassen van personeelsdientsten is een vervolg op een eerder onderzoek. Het onderwerp van dit eerdere onderzoek was het aanpassen van de personeelsdiensten met behulp van het bestaande systeem TURNI. Een aanbeveling van dit onderzoek was om het probleem aan te pakken met thoerieën vanuit de wiskunde in plaats van met bestaande systemen. Dit verslag beschrijft de theorie, implementatie en resultaten van de toepassing van deze theorieën.

Opdrachtomschrijving

Modelleer en implementeer een algoritme dat consistent is in het genereren van diensten voor machinisten, waarbij voornamelijk gelet moet worden op de veranderingen aan de originele personeelsdiensten. Bovendien moet er een manier zijn waarin de kwaliteit van de gevonden oplossing te beoordelen is.

Methode van aanpak

De stage begon met het verzamelen en lezen van onderzoek naar vergelijkbare problemen. Ook de probleembeschrijving en analyse van het eerdere onderzoek werd hierin meegenomen. Het doel van deze eerste fase was om een beeld te krijgen van het huidige probleem en het bekend raken met mogelijke manieren om het probleem aan te pakken.

Na deze fase van verzamelen van informatie en classificeren van het probleem zijn keuzes gemaakt ten aanzien van de te gebruiken theorie en implementatie. Het praktische probleem is in deze fase omgeschreven zodat via de gekozen theorie een oplossing gevonden kon worden. Beslissingen ten aanzien van de implementatie zijn voornamelijk gebasseerd op de verwachte uiteindelijke snelheid van het geheel.

De fase van keuze en implementatie werd gevolgd door een fase van validatie. Het programma is hiervoor getest met 2 cases uit de praktijk. De gegenereerde diensten zijn allemaal diensten die toegestaan zijn. De uitkomsten laten aanzienlijke verbeteringen zien ten opzichte van eerder onderzoek en de echte realisaties. Deze verbeteringen komen naar voren in het aantal benodigde machinisten (een reductie van mininaal 2 procent ten opzichte van de realisatie), het aantal gewijzigde diensten (een reductie van minimaal 7 procent ten opzichte van de realisatie) en de benodigde rekentijd (tot 14

maal sneller dan de realisatie).

De belangrijkste conclusie die na het analyseren van de uitkomsten getrokken kon worden is dat er duidelijke verbeteringen behaald kunnen worden. Deze nieuwe manier van aanpak kan verbeteringen geven op alle fronten van het probleem.

Inhoudsopgave

Hoofdstuk 1

Inleiding

Voor het goed laten verlopen van het primaire proces van NS Reizigers, het leveren van een betrouwbare dienstregeling aan de reizigers, is veel planning nodig. Een kwalitatief goede planning van diensten, materieel en personeel is essentieel om de doelstellingen van het bedrijf te halen. Voor redenen als overzichtelijkheid en complexiteit worden deze onderdelen opgedeeld in kleinere, maar nog steeds complexe, planningsvraagstukken. Eén van deze vraagstukken is de personeelsplanning wanneer er sprake is van werkzaamheden aan het spoor.

Als een gedeelte van de rail infrastructuur tijdelijk niet bruikbaar is, door bijvoorbeeld periodiek onderhoud of nieuwbouw, is er sprake van een zogenaamde buitendienststelling (BDS) van dit traject deel. Zo'n buitendienststelling kan (grote) gevolgen hebben voor de normale dienstregeling. Wanneer er onderhoudswerkzaamheden plaatsvinden aan een deel van het spoor dan kan er tijdens dit onderhoud geen gebruik gemaakt worden van dit stuk spoor. Hierdoor moet de planning van alle machinisten die rijden op dit deel van het spoor, en door een sneeuwbaleffect ook sommige machinisten die hier niet direct gebruik van maken, aangepast worden. Een voorbeeld van het sneeuwbaleffect is wanneer een machinist door een buitendienststelling niet op een station komt waar hij normaal gesproken wel komt. Als volgens het jaarplan de machinist op dit station een andere trein neemt, dan moet er een vervangende machinist gevonden worden voor deze trein. Voor deze vervangende machinist verandert ook het rooster zonder dat zijn dienst direct geraakt hoeft te zijn. Geraakt wil zeggen dat de buitendienststelling het traject treft waar deze trein of machinist op rijdt. Het probleem in het plannen zit, naast het gewone plannen, in het feit dat het aantal machinisten al (ongeveer) bekend is. Het aantal machinisten dat volgens de reguliere

planning nodig was, staat al ingeroosterd voor de periode en zal dus ook uitbetaald worden. Minimaliseren van het aantal machinisten zal dus niet de hoofddoelstelling zijn bij het inplannen van de machinisten tijdens een BDS. Hier zal het minimaliseren gericht zijn op het zo laag mogelijk houden van de afwijking van het aantal reeds gecontracteerde machinisten en de afwijking van de inhoud van de diensten. De planning van machinisten op dagen dat er sprake is van een buitendienststelling is het onderwerp van dit stageverslag.

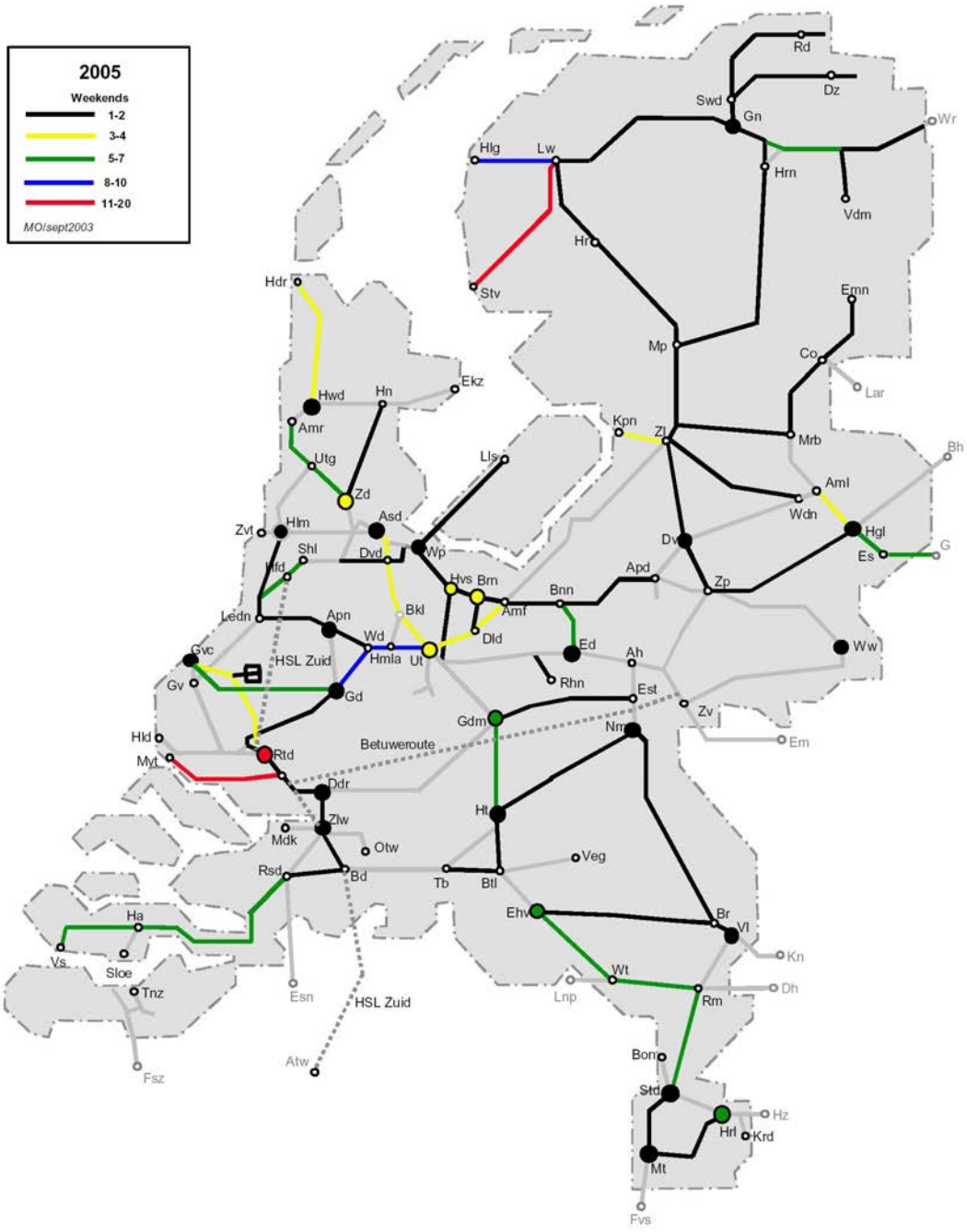
De huidige werkzaamheden rond het inplannen van de machinisten op de door buitendienststellingen veroorzaakte afwijkingen van de reguliere dienstregeling worden op dit moment grotendeels op meer ambachtelijke wijze ingepland. Dit is een tijdrovende bezigheid die gezien de complexiteit van het probleem niet een oplossing kan leveren van constante kwaliteit. Hierdoor ontstaat de behoefte aan een model dat op een efficiënte wijze een rooster voor de machinisten construeert en waarvan theoretisch is na te gaan hoeveel kwaliteit het levert.

Ter illustratie is in figuur 1.1 de planning weergegeven van het geplande onderhoud in 2005. Zoals men kan zien zijn er een behoorlijk aantal BDS'en gepland. Met deze informatie kan duidelijk gemaakt worden dat een goede planning tijdens een BDS een grote meerwaarde kan hebben voor de NS.

Er is reeds onderzoek gedaan naar twee verschillende manieren om dit probleem op te lossen, dit verslag is ook een vervolg van dit onderzoek. Ten eerste is er gekeken of het met de aanwezige modellen en software mogelijk is om de machinisten in te plannen, zie [1]. Ten tweede is er bij de NS een opzet gemaakt voor een zelf ontwikkelde tool dat diensten genereert. Deze tool genereerde reeds oplossingen. Hoewel deze oplossingen aan de eisen van de NS voldoen en zij binnen een korte tijd worden gegenereerd, is een nadeel dat zij gebruikersafhankelijk (niet consistent) zijn. De oplossing is afhankelijk van de input die door de gebruiker bepaald wordt. Verschillende input levert verschillende oplossingen waarvan wel bekend is welke oplossing het beste is maar niet bekend is hoe goed deze oplossing is. Hier begint de opdracht, die als volgt kan worden omschreven:

Modelleer en implementeer binnen de bestaande tool en de gebruikte theorie, een algoritme dat consistent is in het genereren van diensten voor machinisten, waarbij voornamelijk gelet moet worden op de veranderingen aan de originele personeelsdiensten. Bovendien moet er een manier zijn waarin de kwaliteit van de gevonden oplossing te beoordelen is.

In de volgende hoofdstukken zullen de theoretische en praktische aspecten



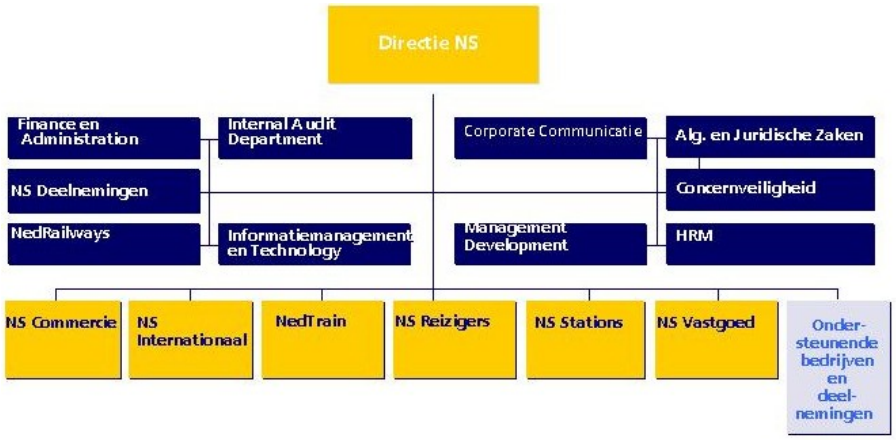
Figuur 1.1: Indeling belangrijke verkeershinder door werkzaamheden aan de infrastructuur gedurende de weekeinden in 2005

van het probleem besproken worden. In het de eerste twee hoofdstukken wordt begonnen met een beschrijving de Nederlandse Spoorwegen N.V. en het creëren van een kader waaruit het probleem voortkomt. Hierna volgen drie hoofdstukken met daarin de theoretische achtergrond van de gekozen aanpak en de implementatie hiervan. In het laatste gedeelte van dit verslag worden twee cases besproken, gevolgd door de bijbehorende de resultaten, conclusies en aanbevelingen.

Hoofdstuk 2

Nederlandse Spoorwegen N.V.

Figuur 2.1 geeft een schematische weergave van de organisatiestructuur van de Nederlandse Spoorwegen (NS). Alle onderdelen hebben hun eigen taken en verantwoordelijkheden binnen het grotere geheel. Zo worden alle stations-



Figuur 2.1: Organigram Nederlandse Spoorwegen

gebouwen beheerd door NS Stations, en vertegenwoordigt NS Commercie de klanten van de NS voor het aansluiten van het product bij de wensen van de klant. NS Reizigers (NSR) is de afdeling die de verantwoordelijkheid draagt over het binnenlands passagiersvervoer.

NSR is, met ongeveer 10000 werknemers en ruim 1 miljoen klanten per

dag, verreweg de grootste aanbieder van personenvervoer per trein in Nederland en tevens het grootste onderdeel van de Nederlandse Spoorwegen. Hier komen het materieel aanbod, personeelsplanning en service graad samen tot een produktlevering in de vorm van een dienstregeling en verzorging van informatie en service bij het afwijken van deze dienstregeling. NSR is opgedeeld in divisies namelijk Financiën, Personeel & Organisatie en Productie.

Binnen de divisie Productie bevindt zich de afdeling NSR logistiek, deze houdt zich bezig met de logistieke planning voor NSR. Onder deze logistieke planning valt bijvoorbeeld het maken van dienstregelingen, de materieel- en personeelsinzet. NSR Logistiek is heeft zelf meerdere afdelingen. In figuur 2.2 staat dit aangegeven. Product ontwerp houdt zich voornamelijk bezig

Figuur 2.2: Organigram NSR Logistiek

met de lange termijn planning; het gaat hier om voorstudies op een termijn tot tien jaar. Jaarplan focust zich op het maken van een dienstregeling voor het komende jaar. Hier wordt rekening gehouden met het beschikbare materieel en het benodigde personeel. Dagplan is verantwoordelijk voor de tijdelijke aanpassingen op de dienstregeling, personeelsplanning en materieelplanning. Deze aanpassingen komen voort uit een tijdelijke verandering van het spooraanbod of uit een tijdelijke verandering in de vraag zoals bij feestdagen. Dagplan bestaat zelf ook weer uit meerdere secties namelijk Projecten, Dienstregeling en Extra Vervoer. Wanneer een verandering van spoor of perron afdoende is om een wijziging op te vangen en er dus geen verandering nodig is in de materieel- en personeelsplanning dan zorgt de sectie

Dienstregeling voor de planning. Moeten er wel aanpassingen komen dan draagt de sectie Projecten verantwoordelijkheid voor de aanpassingen. Tijdelijke veranderingen in de vervoersvraag worden afgehandeld door de sectie Extra Vervoer. Automatisering beheert de systemen die gebruikt worden. Hiernaast houdt de sectie Automatisering zich bezig met het ontwikkelen van nieuwe systemen.

Naast de normale gang van zaken is NSR Logistiek altijd op zoek naar nieuwe technieken om problemen aan te pakken. Om nieuwe aanpakken te verkennen is er onder NSR Logistiek de stafafdeling onderzoek/innovatie, hier wordt geëxperimenteerd met nieuwe ideeën en onderzocht of er wiskundige technieken zijn om het huidige proces verder te verbeteren. Op deze stafafdeling is het ook waar universitaire stages op gebied van wiskundig onderzoek uitgevoerd worden.

Hoofdstuk 3

Personeelsplanning

Kort samengevat komt de probleemstelling op het volgende neer: Modelleer en implementeer een algoritme dat consistent is in het genereren van een verzameling diensten voor machinisten ter invulling van een dienstregeling tijdens een buitendienststelling. Om een uitleg te geven van het probleem zal in de volgende paragraaf eerst de definities gegeven worden van de gebruikte termen. Hierna volgt de wijze waarop personeelsdiensten worden aangepast.

3.1 Termen

3.1.1 Taken

Een taak is een activiteit die uitgevoerd moet worden door één persoon en niet verder is op te delen in meerdere activiteiten. Een voorbeeld hiervan is het rijden van station A naar station B, zonder tussen de stations, waar een overstapmogelijkheid is (station A kan hier ook gelijk zijn aan station B), te stoppen. Hier is het niet mogelijk om op de helft van rit A naar B over te stappen op een andere trein. Deze rit is dus een taak.

Taken komen in verschillende vormen voor, maar ze hebben allemaal de volgende kenmerken:

1. begin- en eindlocatie,
2. begin- en eindtijd,
3. bijbehorende traject,
4. materieelsoort.

Er bestaan meerdere soorten taken zo zijn er rittaken, rangeertaken, passagierstaken en leeg materieeltaken. Rittaken zijn de taken waar een machinist reizigers vervoert. Rangeertaken zijn bijvoorbeeld taken waarbij treinen van en naar het opstelterrein gereden worden. Passagierstaken zijn taken waarbij een machinist meerijdt met een andere machinist om zo op het station te komen waar de volgende taak begint. Leeg materieeltaken zijn taken waarin een machinist een trein verplaatst van één station naar een ander station zonder dat er passagiers in zitten. Ook zijn er zo geheten VL en NSRDAG taken. Dit zijn taken die ingeroosterd worden om reserve capaciteit te creëren. De VL taken komen vanuit de verkeersleiding en worden ingezet bij 'vertragingen' van de machinist. De NSRDAG taken komen vanuit Dagplan en worden ingezet bij het inplannen van taken. Bij de eerste gaat het vaak over reservecapaciteit van ongeveer een uur en bij de tweede gaat het over dagdelen of hele dagen.

3.1.2 Standplaatsen

Iedere dienst begint en eindigt op een standplaats. Een standplaats is een station dat door bepaalde kenmerken is gekozen tot standplaats. Er zijn in totaal 29 stations die ook standplaats zijn. De standplaats van een machinist heeft invloed op het plannen van taken en diensten bij machinisten. De totale infrastructuur van het spoor is erg groot, hierdoor is het niet mogelijk voor de machinisten om alle trajecten te kennen. Ook zorgt de bekendheid met een traject ervoor dat een machinist weet wat hij kan verwachten en hierdoor dus een veiligere rit kan rijden. Het al dan niet kennen van een traject wordt wegbekendheid genoemd. Iedere standplaats heeft machinisten met een bepaalde wegbekendheid. Wanneer een standplaats een bepaalde wegbekendheid heeft, dan betekent dit dat er op die standplaats minimaal één machinist is die het traject behorend bij die wegbekendheid kan rijden. Naast deze restricties zijn er ook enkele regels waaraan iedere standplaats moet voldoen. Deze regels komen voort uit de discussie van enkele jaren geleden over het gelijk verdelen van de aantrekkelijke en minder aantrekkelijke taken. Dit heeft geresulteerd in een set van regels die lusten en lasten verdeling is genoemd. Deze restricties zijn niet van toepassing op het plannen door Dagplan, alleen Jaarplan heeft hier mee te maken.

3.1.3 Diensten

Een dienst is een verzameling van opeenvolgende taken die worden uitgevoerd door één persoon op één dag. Een dienst moet normaliter aan een

aantal meer en minder strenge eisen voldoen.

- Iedere dienst begint op een standplaats en eindigt ook op deze standplaats.
- De toegestane lengte van een dienst is afhankelijk van de begintijd van een dienst. Hier een overzicht van de maximale lengte:

type	*begintijd*	*eindtijd*	*maximale lengte*
1^ste vroege dienst	4-5 uur	voor 13 uur	8 uur
2^de vroege dienst	5-6 uur	voor 15 uur	9 uur
reguliere dienst	6 uur	voor 24 uur	9,5 uur
late dienst	15-16:30 uur	voor 1:30 uur	9 uur
nacht dienst	X	na 1:30 uur	8,5 uur

  Of een dienst een nacht dienst is, is alleen afhankelijk van de eindtijd. Er staat bij nacht dienst dus geen begintijd aangegeven.
- De minimale lengte van een dienst is 4 uur en de maximale lengte is 9,5 uur.
- Iedere dienst langer dan 5,5 uur moet een pauze bevatten. Bij voorkeur bevindt deze pauze zich voor het verstrijken van 5,5 uur na aanvang van de dienst en na 5,5 uur voor het einde van de dienst.
- Een pauze moet op een station gehouden worden waar faciliteiten zijn voor een pauze, denk hierbij bijvoorbeeld aan een kantine.
- Een pauze heeft een minimale lengte van 30 minuten.
- De opvolgtijd tussen twee taken, wanneer deze taken niet op dezelfde trein zijn, is minimaal 20 minuten.
- Iedere dienst heeft een voor- en een natijd. Voortijd is het aantal minuten dat de machinist aanwezig moet zijn voordat de eerste taak begint. De natijd is het aantal minuten dat de machinist nodig heeft na het uitvoeren van de laatste taak. Op deze momenten krijgt de machinist de tijd zich voor te bereiden op zijn dienst of de tijd om zijn dienst af te ronden. De voor- en natijden hebben de waarden van respectievelijk 20 en 15 minuten. Deze tijden worden ook meegeteld bij de dienstlengte.
- Alle diensten moeten voldoen aan de regels welke gelden volgens het lusten en lasten principe.

Diensten komen voor in verschillende soorten. Er zijn twee types te onderscheiden namelijk Jaarplan- en Dagplandiensten. Deze types komen ook weer voor in meerdere variaties. Jaarplan diensten zijn de diensten die worden ingepland door Jaarplan. Deze diensten zijn de diensten die als er geen bijzonderheden zijn gewoon gedraaid worden. Een speciaal geval hiervan zijn NSRDAG diensten. Dit zijn diensten die door Jaarplan ingeroosterd worden om reservecapaciteit te creëren voor Dagplan. Deze diensten hebben geen taken maar wel een ruime begin- en eindtijd waarbinnen eventueel taken gepland kunnen worden. Dagplandiensten zijn Jaarplandiensten die gewijzigd zijn als gevolg van een BDS. Het kan voorkomen dat Dagplandiensten maakt waar nog geen personeel voor ingepland is, in dat geval spreekt men van een extra dienst.

3.1.4 Buitendienststelling

Buitendienststellingen komen voor als er sprake is van een tijdelijke vermindering van een deel van de infrastructuur en/of de inzet van extra treinen waardoor er meer planning nodig is. Buitendienststellingen zijn in de regel enkele maanden van te voren bekend. Het kan bijvoorbeeld zo zijn dat Prorail, het bedrijf dat verantwoordelijk is voor het onderhoud van het spoor, een deel van een traject vrij wil hebben voor onderhoud. Ook kan er sprake zijn van de inzet van extra treinen wanneer er sprake is van een grote gebeurtenis zoals beurzen en sportevenementen. Onderhoud aan het spoor komt met grote regelmaat voor en kan enkele dagen duren. Vanuit NSR-logistiek wordt er een aangepaste personeelsplanning voor die dag(en) gemaakt.

3.1.5 Alternatief vervoer

Om personeelsdiensten kloppend te maken moet er soms gebruik worden gemaakt van andere soorten vervoer dan de trein. Het kan bijvoorbeeld zo zijn dat het rooster enkele diensten bevat waarbij een machinist niet eindigt in zijn eigen standplaats en er voor hem geen mogelijkheid meer is om een trein te nemen. Wanneer zulke gevallen zich voordoen is het nodig dat er vervoer geregeld wordt voor de machinist. Dit vervoer kan een taxi, een lijnbus of een NS-bus zijn, die wordt ingelast vanwege een BDS. Voornamelijk voor de eerste taak of na de laatste worden deze mogelijkheden ingezet.

De verschillende vormen van vervoer hebben zo hun voor- en nadelen. Zo weet je van een taxi dat deze een redelijk constante duur heeft om van plaats

A naar plaats B te gaan, maar ook is bekend dat de kosten erg hoog zijn. Een bus daarentegen heeft in vergelijking zeer lage kosten, maar is minder stipt. Omdat stiptheid voor de NS zeer belangrijk is, worden busritten bij voorkeur alleen gebruikt aan het begin en eind van een dienst. Op deze manier is er in geval van vertraging van de bus een redelijke buffer in de vorm van de voortijd.

3.1.6 Optimaliteit

In de meest gangbare vorm van personeelsplanning is het doel om de omvang van het benodigd personeel, en daarmee de kosten binnen alle randvoorwaarden, zo laag mogelijk te houden. Hoewel de randvoorwaarden het probleem zeer lastig kunnen maken is het doel duidelijk. Bij het probleem waar de personeelsdiensten aangepast moeten worden is het doel minder eenduidig. Logistiek gezien is het wenselijk als de verandering ten opzichte van de oude diensten minimaal is. Dit zorgt ervoor dat de tijd die nodig is om de veranderingen te verwerken ook minimaal is. Zo is er dus ook weinig inspanning nodig binnen de geraakte standplaatsen. Financieel gezien is het wenselijk wanneer er een minimaal aantal aan extra personeel nodig is en wanneer het aantal taxiritten beperkt blijft. Vanuit het oogpunt van de machinisten is een dienst goed wanneer enerzijds de veranderingen klein zijn en anderzijds weinig impopulaire diensten gemaakt worden. Zo ontstaat er een spanningsveld tussen al dan niet subjectieve invalshoeken. Vanwege deze verschillende invalshoeken zal er in de modellen geen gebruik worden gemaakt van kosten uitgedrukt in geldbedragen, maar kosten die een indicator zijn van de wenselijkheid van verschillende opties. Er kan dus beter gesproken worden over parameters dan kosten.

3.2 Personeelsplanning bij Dagplan

3.2.1 Regels en veronderstellingen

Voor diensten tijdens een buitendienststelling, diensten van een tijdelijk karakter, zijn enkele eisen minder scherp gesteld of zelfs opgeheven. De meest complicerende regels komen vanuit de lusten en lasten verdeling. Bij deze verdeling hoeft bij een BDS geen rekening gehouden te worden omdat de wijzigingen van tijdelijke aard zijn. De eis met betrekking tot de plaats van de pauze is minder scherp gesteld. De grenzen van 5,5 uur mogen worden losgelaten maar zijn net als de lusten en lasten verdeling nog wel wenselijk. Wat overblijft, na het vervallen van deze eisen, is een overzichtelijke verza-

meling van regels en eisen. Naast de eisen die betrekking hebben op een dienst zijn er ook eisen met betrekking tot veranderingen in diensten, hiervoor gelden de volgende regels:

- Zolang de verschuiving niet in conflict raakt met andere regels mag een dienst zonder overleg een half uur verschoven worden, zowel naar voren als naar achteren.
- De opvolgtijd tussen twee taken die niet op dezelfde trein zijn is verandert van minimaal 20 minuten naar minimaal 15 minuten.
- De opvolgtijd tussen twee taken is 10 minuten wanneer 1 van de 2 taken een taxi- of bustaak is.
- Wanneer de eerste taak in een dienst een bus- of taxi taak is dan wordt de voortijd 10.
- Een late dienst mag niet omgezet worden naar een nachtdienst.
- Een vroege dienst mag niet omgezet worden naar een late dienst beginnend na 18 uur.
- Een nachtdienst eindigend voor of om 7 uur mag niet omgezet worden naar een dienst eindigend na 7 uur.

Bij het aanpassen van personeelsdiensten hoort dus rekening gehouden te worden met zowel de eisen ten aanzien van een dienst als met de eisen op wijzigingen van een dienst.

Naast deze eisen zijn er ook enkele veronderstellingen gemaakt over de machinisten en standplaatsen. Zo wordt aangenomen dat op een standplaats iedere machinist gelijke weg- en materieelbekendheid heeft. Deze veronderstelling maakt het plannen op landelijk niveau een stuk minder complex. Op deze manier kunnen alle machinisten van een standplaats samengenomen worden en wordt dus voorkomen dat de landelijke planning op bijna individueel niveau plaatsvindt. Ook is het efficiënter om het toewijzen van machinisten aan diensten op lokaal niveau te doen omdat hier de kennis over de groep machinisten specifieker is.

3.2.2 Aanpak

Dagplan plant bij een BDS op basis van dagen. Wanneer een BDS een week duurt dan wordt de planning voor deze week dag voor dag veranderd. Voor een dag worden alle diensten bekeken door de planners en wordt er

handmatig met taken geschoven en ook worden nieuwe taken ingeschoven. Dit is een tijdrovend proces maar ervaring en kennis van de dienstregeling en het spoorwegnet zorgen er voor dat alle cases binnen twee weken behandeld kunnen worden. De resultaten van Dagplan zijn op het eerste gezicht niet onaantrekkelijk maar Dagplan kan niet alles overzien. Zo is het moeilijk voor de planners om alle diensten mee te nemen. Wanneer zij plannen, is er dus een beperking in het aantal diensten en het aantal verschuivingen in het rooster waar zij rekening mee kunnen houden.

Hoofdstuk 4

Wiskundige technieken

Voor het maken van een tool dat een goede en consistente oplossing levert voor het aanpassen van personeelsdiensten, is een solide basis nodig van waaruit het probleem aangepakt wordt. Om de uitvoering van deze aanpak te bespreken, zal dan ook eerst deze basis beschreven moeten worden. In dit hoofdstuk komen alle wiskundige aspecten van de oplossingsmethode aan bod. In de beschrijvingen van de gebruikte technieken is geprobeerd de uitleg niet te technisch te maken wat betekent dat bewijzen van de technieken in dit hoofdstuk achterwege gelaten zijn. Voor een meer technische beschrijving van de methoden wordt er verwezen naar [4] en [3].

4.1 Relaxatie

Het principe van het relaxeren van een model is het verminderen of afzwakken van voorwaarden die gelden op het model. Hierdoor is het zoeken naar een oplossing makkelijker en kan de benodigde rekentijd aanzienlijk verminderd worden. Men kan ook wel spreken over een optimistische versie van het model, omdat bij een minimalisatie probleem de relaxatie sneller een oplossing zal vinden die kleiner dan of gelijk is aan het minimum van het origineel. Er zijn twee redenen voor het gebruik van een relaxatie. Ten eerste levert de relaxatie een ondergrens voor de oplossing van het originele probleem. Ten tweede kan de oplossing van de relaxatie, hoewel in de meeste gevallen niet toegelaten, vaak gebruikt worden als startpunt van een specifieke heuristiek.

4.1.1 Lagrange Relaxatie

Bij de Lagrange Relaxatie (LR) is er sprake van afzwakken van voorwaarden die het oplossen van het originele probleem bemoeilijken. Neem het volgende standaard minimalisatie probleem.

$$\min\{cx\} \tag{4.1}$$

$$\text{s.t. } Ax \leq b \tag{4.2}$$

$$Cx \leq d \tag{4.3}$$

Veronderstel dat als de voorwaarden (4.2) weggelaten zouden worden er een relatief makkelijk probleem ontstaat. Deze voorwaarden worden dan gerelaxeerd; de andere voorwaarden blijven in het model (let wel dat de voorwaarden (4.3) ook leeg kunnen zijn). Laat nu λ een niet negatieve vector zijn, genaamd de Lagrange multiplier. Dan wordt een ondergrens $\Theta(\lambda)$ voor iedere $\lambda \geq 0$:

$$\Theta(\lambda) = \min\{cx - \lambda(Ax - b)\} \tag{4.4}$$

$$\text{s.t. } Cx \leq d \tag{4.5}$$

In deze relaxatie zijn de slack-variabelen van de complicerende voorwaarden (4.2) opgenomen met gewicht λ in de doelfunctie en zijn de moeilijke bijvoorwaarden verwijderd. Er wordt hier ook wel gesproken over het dualizeren van de voorwaarden (4.2). Om van het originele probleem een zo scherp mogelijke ondergrens te krijgen moet het volgende model opgelost worden:

$$\max_{\lambda} \Theta(\lambda) \tag{4.6}$$

$$\text{s.t. } \lambda \geq 0 \tag{4.7}$$

Dit wordt het Lagrangian dual van het origineel genoemd naar de bijvoorwaarde $Ax \leq b$. Voor het vinden van de λ die de relaxatie maximaliseert zijn meerdere methodes. De meest gebruikte en redelijk duidelijke methode is de subgradiënt methode. Deze methode wordt ook gebruikt in het programma om de optimale Lagrange multipliers te vinden. Voor een bespreking van deze methode wordt er verwezen naar [4].

4.2 Kolomgeneratie

Eén van de kenmerken en ook moeilijkheden van het probleem is de grootte van de oplossingsruimte. Er is een groot aantal mogelijke vervangende diensten per originele dienst mogelijk. Het is waarschijnlijk dat bijvoorbeeld diensten van maar twee taken of diensten, waarbij vaak gebruik gemaakt wordt van een taxi, nooit geselecteerd zullen worden voor de uiteindelijke oplossing. Wanneer er bijvoorbeeld sprake is van duizend originele diensten, dan is het maximaal aantal diensten dat geselecteerd zal worden voor de oplossing ongeveer gelijk aan duizend. Het aantal diensten dat niet geselecteerd wordt is dus vele malen groter dan het aantal diensten dat wel geselecteerd zal worden. Ook is de verzameling van alle mogelijke diensten niet bekend. Het maken van alle mogelijke vervangers is, als gevolg van de grote oplossingsruimte, ook een ondoenlijk karwei.

In het geval van een probleem als hierboven beschreven, waarin de oplossing maar een klein deel van de oplossingsruimte is, wordt er vaak gebruik gemaakt van kolomgeneratie. Het principe van kolomgeneratie is het beginnen met een deelverzameling van de oplossingsruimte. Hierna wordt deze verzameling stap voor stap uitgebreid met elementen waarvan berekend kan worden dat deze de optimale oplossing van het probleem met de deelverzameling het meest verbeteren. De waardering van de elementen die toegevoegd worden wordt op gelijke wijze gedaan als bij de herziene simplex methode, namelijk op basis van de gereduceerde kosten. Echter, het verschil tussen kolomgeneratie en de herziene simplex methode is dat bij kolomgeneratie het niet nodig is om ieder element van de oplossingsruimte expliciet te kennen. De waardering van de toe te voegen deeloplossing gebeurt via een optimalisatieprobleem. De oplossing van dit probleem, dat afhankelijk is van de huidige beste oplossing van het originele probleem, zoekt een deelverzameling die de optimale oplossing het meest verbetert. Zo zal bij de herziene simplex methode toegepast op een minimalisatieprobleem de kolom gekozen worden met de maximale negatieve waarde en gestopt worden als deze maximale waarde een ander teken heeft dan de vorige gevonden waarden. Bij kolomgeneratie wordt bekeken wat de minimale waarde is van de gereduceerde kosten, als de minimale waarde negatief is dan wordt de gevonden deeloplossing toegevoegd aan de deelverzameling van de oplossingsruimte. Is de waarde positief dan is het bekend dat er geen deeloplossing meer is die de tot nu toe gevonden oplossing verbetert. Dit geldt ook voor een maximalisatieprobleem omdat het negatieve minimum van de negatieve criteriumfunctie gelijk is aan het maximum van de criteriumfunctie, $-min\{-Z\} = max\{Z\}$. Op deze manier kan ieder maxi-

malisatieprobleem geschreven worden als een minimalisatieprobleem en dus ook op dezelfde manier behandeld worden.

Bij het genereren van kolommen is het zo dat tijdens het proces de gereduceerde kosten van de al geselecteerde kolommen ook verandert met iedere iteratie. Deze verandering kan zo groot zijn dat tijdens de iteratie deze kosten een positieve waarde krijgen. In de optimale oplossing bestaan er ook geen kolommen meer met negatief gereduceerde kosten. Om deze redenen wordt er na het genereren van een nieuwe kolom gekeken naar de nu verkregen verzameling kolommen en worden de kolommen met positief gereduceerd kosten verwijderd uit de verzameling diensten.

Vaak heeft het optimaliseringsprobleem voor het vinden van de gereduceerde kosten een speciale vorm. Hierdoor kan dan met een specifiek algoritme een optimale oplossing gevonden worden. Termen die vaak gebruikt worden met betrekking tot kolomgeneratie zijn master problem (MP), restricted master problem (RMP) en pricing problem. Het master problem is het originele probleem waarvoor een oplossing gevonden moet worden. Het restricted master problem is het probleem waarbij niet de gehele oplossingsruimte wordt bekeken voor een oplossing maar waar een deel van de oplossingsruimte wordt bekeken. Het pricing problem is tenslotte het probleem waar de kolommen met minimale/maximale gereduceerde kosten gezocht worden. Men kan ook wel zeggen dat de 'price' van de deeloplossing hier wordt vastgesteld.

4.3 Branch-and-Price

Branch-and-price is een combinatie twee methoden, namelijk: kolomgeneratie en branch-and-bound. Er wordt gebruik gemaakt van deze methode wanneer er sprake is van een geheeltallig optimalisatieprobleem met de kenmerken van een probleem waar kolomgeneratie ook toegepast kan worden. Als eerste wordt er gerelaxeerd door de geheeltalligheidseis te laten vallen. Voor het ontstane probleem wordt een (triviaal) RMP gekozen. Nadat deze is opgelost wordt kolomgeneratie hierop toegepast. Als een bepaald aantal iteraties is gedaan of de criteriumwaarde aan bepaalde eisen voldoet, wordt er door middel van branching een toegelaten oplossing gevonden. Een oplossing van kolomgeneratie kan een geheeltallige oplossing zijn, maar deze oplossing hoeft nog niet toegelaten te zijn. Dit komt door de relaxatie die toegepast is. De interactie tussen kolomgeneratie en branching kan meerdere keren voorkomen bij het oplossen van een probleem. Het kan zelfs zo zijn dat er in iedere vertakking een kolomgeneratiealgoritme wordt aangeroepen.

4.4 Labeling methode

De labeling methode is een belangrijk onderdeel in de meeste algoritmes om het kortste pad te zoeken in een netwerk. De output van deze methode is een uitgaande boom vanaf het beginpunt. De opbouw van deze boom gaat iteratief, deze opbouw wordt afgebroken wanneer het kortste pad gevonden is.

Bij iedere knoop worden in deze methode 3 gegevens onthouden. Dit zijn de afstand, de 'parent' knoop en de status van de knoop. De afstand is de tot dan toe kortst gevonden afstand vanaf de begin knoop. De afstand moet hier eigenlijk gezien worden als kosten. Men spreekt van afstand omdat dit makkelijk verbonden kan worden aan een kortste pad maar in de praktijk kunnen de kosten veel verschillende interpretaties hebben. Een voorwaarde op de kosten is wel dat ze vergeleken kunnen worden om het 'goedkoopste' pad te vinden. De 'parent' knoop is de knoop die de directe voorganger is van de knoop. De status heeft in de methode 3 mogelijke waarden, dit zijn: onbereikt, tijdelijk gelabeled en permanent gelabeled. Wanneer een knoop (nog) niet geëvalueerd is dan heeft deze de status 'onbereikt'. Wanneer zeker is dat de afstand het minimum is dan is de knoop permanent gelabeled. Als deze 2 statussen niet bereikt zijn, met andere woorden er is mogelijk nog een korter pad te vinden, dan heeft de knoop de status 'tijdelijk gelabeled'. Als de doel knoop permanent gelabeled wordt dan voldoet de methode aan de stopconditie.

Hoofdstuk 5

Model en Oplossingsmethode

In dit hoofdstuk wordt het algoritme behandeld dat gebruikt is voor het aanpassen van de personeelsdiensten. In de eerste paragraaf wordt het model besproken dat al gemaakt was. Hierna wordt, met de in hoofdstuk 4 besproken technieken, de uitbreiding op dit model doorgenomen.

5.1 Model

5.1.1 Huidige model

Voor het aanpassen van de personeelsdienst is een model opgesteld en een tool ontwikkeld, zie [5]. Het onderwerp van dit stageverslag is een uitbreiding die hierop gedaan is. Voordat er op deze uitbreiding ingegaan kan worden zal eerst het huidige model in redelijk detail besproken worden. Dit is voornamelijk om inzicht te verschaffen in de herkomst van de probleemstelling en om een context te scheppen binnen het geheel van dienstaanpassingen als gevolg van een buitendienststelling. Aan dit model zijn voor verdere uitbreidingen geen aanpassingen nodig. Het zal dus deze vorm houden.

In woorden beschreven is het overkoepelende probleem als volgt: gegeven de taken die niet uitgevoerd kunnen worden en extra taken als gevolg van een buitendienststelling, genereer er een verzameling van personeelsdiensten waarbij het aantal extra mensen minimaal is en waar er weinig tot geen gebruik wordt gemaakt van bus- en taxiritten om personeel te verplaatsen. Anders gezegd, er is een verzameling personeelsdiensten waarbij iedere uit te voeren taak in een personeelsdienst zit en waarbij veranderingen aan de originele diensten niet groot zijn. Met de laatste formulering wordt duidelijk waarom er voor de gebruikte modellering is gekozen. Het probleem dat een taak in ieder geval in één dienst moet zitten wordt ook wel een set-covering

probleem genoemd. Dit is een uitgebreid beschreven aanpak (zie [4] en [3]) dat in veel problemen naar voren komt, bijvoorbeeld bij het maken van roosters. Het probleem heeft gemodelleerd de volgende vorm:

$$\min p \sum_{\delta \epsilon \Delta} y^{\delta} + \sum_{\delta \epsilon \Delta} \sum_{k \epsilon K^{\delta}} c_k^{\delta} x_k^{\delta} \tag{5.1}$$

$$\text{s.t.} \quad \sum_{\delta \epsilon \Delta} \sum_{k \epsilon K^{\delta}} a_{ik}^{\delta} x_k^{\delta} \geq 1 \qquad \forall i \epsilon N \tag{5.2}$$

$$\sum_{k \epsilon K^{\delta}} x_k^{\delta} + y^{\delta} \geq 1 \qquad \forall \delta \epsilon \Delta \tag{5.3}$$

$$x_k^{\delta}, y^{\delta} \epsilon \{0, 1\} \tag{5.4}$$

$$x_k^{\delta} = \begin{cases} 1, \text{ als dienst } \delta \text{ wordt vervangen met dienst } k \text{ uit } K^{\delta} \\ 0, \text{ anders} \end{cases}$$

$$y^{\delta} = \begin{cases} 1, \text{ als dienst } \delta \text{ vervalt} \\ 0, \text{ anders} \end{cases}$$

A = matrix met als kolommen diensten en als rijen taken.
a_{ik}^{δ} = element van matrix A.

$$a_{ik}^{\delta} = \begin{cases} 1, \text{ als taak } i \text{ aanwezig is in dienst } k^{\delta} \\ 0, \text{ anders} \end{cases}$$

Δ = verzameling van te vervangen diensten
K^{δ} = verzameling diensten die dienst δ kunnen vervangen
N = verzameling van taken dat minimaal 1 keer geselecteerd moet worden
c_k^{δ} = kosten voor het vervangen van dienst δ door dienst k
p = kosten voor het niet vervangen van een jaarplan dienst

In dit model bestaan twee type beslissingsvariabelen, y^{δ} en x_k^{δ}. De eerste staat voor het al dan niet selecteren van een vervangende dienst voor dienst δ. De tweede geeft aan dat wanneer er een vervanger voor dienst δ gekozen wordt, welke dienst uit de verzameling K^{δ} dit betreft. In geval dat dienst δ niet geraakt is door de BDS, dan is δ ook een element van K^{δ}. De standaard modellering van het set-covering probleem is puur gericht op het minimaliseren van de kosten en wordt dus eigenlijk toegepast

op het aantal diensten. Om dit aan te passen naar een model dat minimaliseert naar de verandering in een rooster is y^δ toegevoegd. Door aan het niet selecteren van een vervanger voor een originele dienst een penalty p te hangen, wordt er 'gedwongen' het aantal x_k^δ dat de waarde 1 krijgt ongeveer gelijk te laten aan het aantal diensten in de originele planning. Bijvoorwaarde (5.3) representeert deze relatie tussen de variabelen x_k^δ en y^δ. De bijvoorwaarde (5.2) verzekert dat iedere taak minimaal één keer voorkomt in de uiteindelijke oplossing.

Het is niet strict noodzakelijk om y^δ op te nemen in het model, er zijn andere manieren om dit te modelleren. Zo kan de y^δ weggelaten worden uit het model door aan K^δ lege diensten toe te voegen. De kosten c_k^δ van deze lege diensten kunnen dan gelijk genomen worden aan p waardoor een gelijkwaardig model ontstaat. In de volgende paragrafen wordt om redenen van overzichtelijkheid van deze situatie, het equivalente model zonder beslissingsvariabele y^δ, uitgegaan.

5.1.2 Huidige heuristiek

Het direct oplossen van dit probleem kost door de omvang van de oplossingsruimte te veel tijd. De mogelijke oplossingen zijn niet allemaal bekend en erg groot in aantal. Daarom wordt er in het zoeken naar een oplossing gezocht in een deelverzameling van de oplossingsruimte. De elementen van deze deelverzameling worden geselecteerd op overeenkomsten met de te vervangen diensten. Hier volgt een voorbeeld van hoe deze vervangende diensten verkregen worden:
Een dienst bevat een een taak van 14:00 tot 15:00 die, door de buitendienststelling, is komen te vervallen. Er onstaat dan in deze dienst een 'gat' van een uur. Dit 'gat' wordt nu opgevuld met taken die binnen dit tijdslot passen. Doordat dit tijdslot met verschillende taken gevuld kan worden, ontstaat er een verzameling van diensten die de originele dienst kunnen vervangen.

Binnen deze deelverzameling die weliswaar te overzien is, maar nog steeds aanzienlijk van formaat is kan binnen een klein tijdsbestek een goede oplossing gevonden worden. Door deze aanname in het bovenstaand model verandert nu de verzameling K^δ van de gehele oplossingsruimte naar een deelverzameling van alle mogelijke vervangers van dienst δ. Hoewel op deze manier een goede oplossing tot stand komt zijn er twee kanttekeningen te plaatsen bij deze aanpak. Ten eerste is de oplossing zeer afhankelijk van de gekozen deelverzameling van de vervangende oplossingen en van de grootte van deze verzameling. Ten tweede is de uiteindelijke kwaliteit van de oplossing niet vast te stellen, er kunnen hooguit conclusies getrokken worden als

'beter dan handmatig plannen' of 'redelijk snel een toegelaten oplossing vinden'. Om deze twee redenen is er voor gekozen een aanpak te implementeren waarbij er wel uitspraken gedaan kunnen worden over het bovenstaande.

5.2 Algoritme

De gewenste uitbreiding op het model uit de vorige paragraaf, ligt in het niet verkleinen van de K^{δ} maar het behouden van de gehele oplossingsruimte. Door de grootte van deze ruimte is het dus noodzakelijk om 'slim' te zoeken in deze verzameling.

5.2.1 Input

De input van het algoritme bestaat uit de volgende punten:

- Lijst van alle (relevante) stations en standplaatsen, inclusief of er een mogelijkheid is voor een pauze.
- Alle taken (begin- en eindtijd, begin- en eindlocatie, wegbekendheid), inclusief passagierstaken, rangeertaken, etc.
- Diensten uit het jaarplan en de NSRDAG diensten.
- Vervallen en nieuwe taken als gevolg van de BDS.
- Een selectie van mogelijke taxiritten (duur, begin- en eindlocatie).
- Lijst met parameters met betrekking tot regels en kosten.

Deze informatie bevat alles dat nodig is voor het maken van een toegestane personeelsdienst. Met de lijsten van parameters, stations en standplaatsen kan een oplossingsruimte gemaakt worden. Hierbinnen kan gezocht worden naar vervangende diensten voor de diensten uit het jaarplan, die alle uit te voeren taken en geen vervallen taken bevatten.

5.2.2 Stappenplan

Aan de hand van het gegeven het model zonder de y^δ variabele

$$\min \sum_{\delta \epsilon \Delta} \sum_{k \epsilon K^\delta} c_k^\delta x_k^\delta \tag{5.5}$$

$$\text{s.t.} \quad \sum_{\delta \epsilon \Delta} \sum_{k \epsilon K^\delta} a_{ik}^\delta x_k^\delta \geq 1 \qquad \forall i \epsilon N \tag{5.6}$$

$$\sum_{k \epsilon K^\delta} x_k^\delta \geq 1 \qquad \forall \delta \epsilon \Delta \tag{5.7}$$

$$x_k^\delta \epsilon \{0,1\} \tag{5.8}$$

en de besproken methode zal er stap voor stap uitgelegd worden welke aanpassingen en uitbreidingen er op het model gedaan zijn.

stap 1 Als eerste wordt van het set-covering probleem de Lagrange Relaxatie, zoals besproken in paragraaf 4.1.1, naar bijvoorwaarde (5.6) en (5.7) genomen. Dit geeft het volgende model.

$$\min \sum_{\delta \epsilon \Delta} \sum_{k \epsilon K^\delta} c_k^\delta x_k^\delta + \sum_{i=1}^{N} (\lambda_i (1 - \sum_{\delta \epsilon \Delta} \sum_{k \epsilon K^\delta} a_{ik}^\delta x_k^\delta) + \mu_i (1 - \sum_{k \epsilon K^\delta} x_k^\delta)) \tag{5.9}$$

$$\text{s.t.} \quad x_k^\delta \epsilon \{0,1\} \tag{5.10}$$

$$\mu_i, \ \lambda_i \geq 0 \qquad \forall i \epsilon N \tag{5.11}$$

De gerelaxeerde constraint (5.7) is nu eigenlijk een penalty geworden wanneer x_k^δ de waarde nul krijgt. Voor de verder analyse kan deze buiten beschouwing gelaten worden. Met behulp van de oplossing van de originele methode kunnen er door gebruik te maken van de subgradient methode de waarden van de Lagrange multiplier berekend worden.

stap 2 Als een dienst k als vervanger van originele dienst δ geselecteerd wordt, zal de criteriumwaarde veranderen met:

$$-c_\delta^\delta + (c_k^\delta - \sum_{i \epsilon N} \lambda_i a_{ik}^\delta x_k^\delta) \tag{5.12}$$

De kosten gegeven door:

$$c_k^\delta - \sum_{i \epsilon N} \lambda_i a_{ik}^\delta x_k^\delta \tag{5.13}$$

zijn de gereduceerde kosten van het model. Voor een lagere waarde van de oplossing zal er nu gezocht moeten worden naar een x_j^δ met gereduceerde kosten die in ieder geval kleiner zijn dan de kosten van de dienst die het zal vervangen. Wanneer je de oplossing zo snel mogelijk naar de optimale oplossing wilt laten bewegen dan moet er gezocht worden naar de diensten met zo laag mogelijke gereduceerde kosten. Aangezien de verschillende waarden van de kolommen niet expliciet bekend zijn, zal hier kolomgeneratie toegepast worden. Het minimalisatieprobleem van de kolomgeneratie komt er dan voor iedere originele dienst δ als volgt uit te zien.

$$\min c_k - \sum_{i \epsilon N} \lambda_i a_{ik} x_k \tag{5.14}$$

$$\text{s.t.} \quad \sum_{k \epsilon K} x_k = 1 \tag{5.15}$$

$$x_k \epsilon \{0, 1\} \tag{5.16}$$

stap 3 De vorm van het kolomgeneratieprobleem is dat van een kortste pad probleem waar de restricties op de paden verwerkt zijn in het feit dat iedere kolom van matrix A een toegestane dienst is. Dit kortste pad probleem wordt voor iedere originele dienst uitgevoerd en zolang er kolommen bestaan die de oplossing van het gerelaxeerde probleem zullen verbeteren. Met de uitkomsten van de kolomgeneratie kunnen de waarden van de Lagrange Relaxatie aangepast worden aan de nieuwe situatie en kan hierna opnieuw een kortste pad gezocht worden met de nieuwe gereduceerde kosten.

Deze aanpak vertoont veel overeenkomsten met de branch-and-price aanpak. Hier is eigenlijk ook sprake van alleen de eerste stap. Het branchen wordt dus achterwege gelaten.

5.2.3 Netwerk

Zoals eerder genoemd is, komt het vinden van individuele diensten ter vervanging van jaarplandiensten er op neer dat binnen een netwerk een kortste pad gevonden moet worden. Dit kortste pad representeert een toegestane

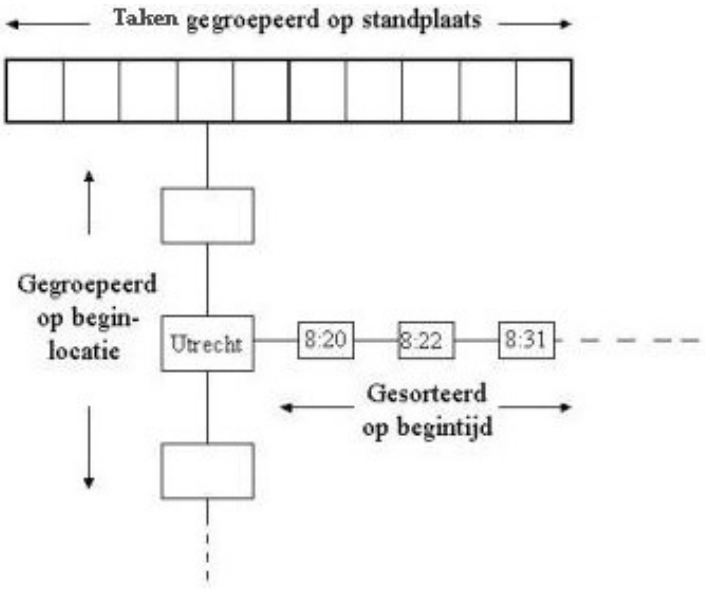
dienst die de tot nu beste oplossing het meest zal verbeteren. Voor het vinden van een pad zal eerst een netwerk geconstrueerd moeten worden. Dit netwerk moet zo gemaakt worden dat het zoeken hierbinnen naar een pad niet te veel tijd in beslag neemt en er moeten ook zoveel mogelijk bijvoorwaarden van het model in opgenomen worden. Dit laatste om het aantal overbodige zoekacties zo laag mogelijk te houden.

Knopen

Taken worden voor het maken van het netwerk gesorteerd. Dit sorteren gebeurt in drie dimensies, namelijk wegbekendheid, beginlocatie en begintijd. Allereerst worden de taken gesorteerd op de wegbekendheid. Van iedere taak is bekend welke standplaats de machinisten hebben die deze taak kunnen uitvoeren. In hoofdstuk 6 zal blijken dat we het hier meer hebben over clusters óf pieces van taken. Wanneer taken een identieke verzameling hebben van standplaatsen dan worden deze gegroepeerd. Vervolgens wordt er gekeken naar de beginlocatie van iedere taak binnen de wegbekendheid groepen. Gelijke beginlocaties worden ook weer gegroepeerd. Tot slot wordt er gekeken binnen de groepen van beginlocaties naar de begintijd van iedere taak en wordt hierop gesorteerd, zie figuur 5.1. Dit zijn tijdrovende handeling- en maar de netwerkconstructie hoeft maar één keer uitgevoerd te worden. Daarom zal dit weinig effect hebben op de uiteindelijke snelheid van het algoritme en de winst in het zoeken binnen het netwerk zal groter zijn dan de tijd die de constructie kost.

Pijlen

Na het sorteren van de knopen worden de pijlen gemaakt. Iedere pijl zal een mogelijkheid tot opvolging voorstellen van de twee verbonden knopen in de richting van de pijl. Met andere woorden als knoop i opgevolgd kan worden door knoop j dan zal er een pijl gaan van i naar j. De eerste voorwaarde wanneer er een pijl komt tussen twee knopen is ook de meest voor de hand liggende: de locatie waar de eerste taak van de machinist eindigt moet gelijk zijn aan de locatie van het beginpunt van de tweede taak. Ook moet de eerste taak zijn afgelopen voordat de tweede taak begint. Een andere technische voorwaarde is die van de wegbekendheid. Wanneer knopen geen overeenkomstig standplaats hebben ten aanzien van de wegbekendheid, bestaat er tussen deze knopen geen pijl. Andere regels ten aanzien van het opvolgen van twee taken heeft betrekking op de tijd tussen deze taken. Wanneer de taken op twee verschillen treinen zijn dan is er een verplichte



Figuur 5.1: Sortering van de taken

tussentijd van 15 minuten. Als de taken op dezelfde trein zijn dan is er geen sprake van een tussentijd. Er is geen maximale duur vastgesteld voor de tijd tussen twee taken. In het programma wordt een tussentijd van 120 minuten aangehouden.

Een andere categorie pijlen is die van de taxiritten. Wanneer de mogelijkheid van een taxirit tussen twee stations bestaat dan wordt dit ook gerepresenteerd door een pijl. De tijd tussen de taken moet dan wel minimaal gelijk zijn aan de duur van de taxirit plus twee maal de overstaptijd. Voor de overstaptijd staat standaard 10 minuten. Binnen het netwerk bestaan nu alleen pijlen op plekken waar een taakopvolging mogelijk is. Bijna alle eisen die gelden voor de diensten zijn door de representatie opgenomen in het netwerk. De eisen die zijn opgenomen zijn eisen die voor alle diensten in algemeenheid gelden.

5.2.4 Kortste pad algoritme

Bij het vaststellen van de zoekmethode is afgestapt van de oorspronkelijke probleemstelling van de stage. De eis dat iedere dienst langer dan 4,5 uur een pauze van minimaal een half uur moet bevatten, is in de modellering niet volledig opgenomen. Deze is niet opgenomen vanwege de variabiliteit van de pauze. Van de pauze staat namelijk niet vast hoeveel minuten na het begin van de dienst deze begint. Er is dus sprake van een tijdsinterval waarbinnen een pauze toegestaan is; dit is voor iedere specifieke dienst ook weer anders. Het modelleren van deze voorwaarden in het netwerk zou betekenen dat in plaats van 1 netwerk voor alle diensten er voor iedere dienst een hoeveelheid grafen gemaakt zou moeten.

Er is geen literatuur gevonden dat een algoritme beschrijft voor het inplannen van een pauze. Heuristieken zijn wel aanwezig maar deze leveren niet een optimale oplossing. Het is ook bekend dat dit een moeilijk probleem is en dat het voor grote instanties niet binnen redelijke tijd op te lossen is. Door het speciale geval van de pauzes tijdens een BDS heeft dit probleem ook een vorm die weinig beschreven is. Het is namelijk zo dat de pauze in de gehele dienst mag zitten, zolang deze maar niet aan het begin of het einde van een dienst zit. In de meest beschreven gevallen zijn er eisen met betrekking tot de plaats van de pauze in het rooster. Bijvoorbeeld de eis dat een pauze binnen 5,5 uur van het begin en binnen 5,5 uur van het eind van een dienst moet vallen.

Dit probleem is op de volgende manier aangepakt: binnen de graaf wordt alleen gezocht naar het kortste pad, al dan niet met een pauze. Wanneer de afstand tot alle punten gevonden is, worden alle mogelijke eindpunten

bekeken. Wanneer deze punten een laatste taak van een dienst kunnen zijn, wordt gekeken of het pad dat naar dit punt leidt een pauze bevat. Alleen de paden met pauze worden toegevoegd aan de oplossingsruimte. Op deze manier lijkt het alsof de gegenereerde diensten ver kunnen afwijken van de optimale diensten. Dit is om meerdere redenen niet het geval. Ten eerste zijn alle gegenereerde paden de kortste paden tussen de begin- en eindknoop. Ten tweede zal het geval dat het kortste pad dat geen pauze bevat niet vaak voorkomen. Wanneer dit het geval zou zijn dan zouden de taken met de hoogste gereduceerde kosten alleen bereikt kunnen worden via pijlen die geen pauze bevatten. Het is niet onwaarschijnlijk dat dit voorkomt maar door het iteratieve karakter van de aanpak is het ook niet onwaarschijnlijk dat een toegelaten pad tussen deze punten in latere iteraties gevonden wordt. Ten derde is gebleken uit test runs dat bij een maximale tussentijd van 120 minuten gemiddeld 80 procent van de pijlen een pauze bevat.

De methode die gebruikt is, komt neer op een labelingsalgoritme zoals besproken in paragraaf 4.4. De kosten zijn hier uitgebreid met een variabele die bijhoudt of het pad een pauze bevat. Hierover is vastgelegd dat bij het vergelijken van kosten de gereduceerde kosten zwaarder wegen dan het wel of niet aanwezig zijn van een pauze. Dit wil zeggen dat een pad van kosten 10 zonder pauze de voorkeur krijgt boven een pad met kosten 15 met pauze. In het geval dat de kosten gelijk zijn zal het wel of niet aanwezig zijn van een pauze in de dienst wel de doorslag geven.

5.2.5 Aannames en beperkingen

Voor en tijdens het proces van modelleren en implementeren zijn enkele keuzes gemaakt ten aanzien van de kwaliteit van de oplossing. Op bepaalde punten is afgeweken van het zoeken naar de optimale oplossing voor het gehele probleem. Deze afwijkingen hebben verschillende redenen. Ten eerste is er afgeweken om de snelheid van het algoritme te verhogen. Ten tweede zijn er keuzes gemaakt voor het implementeren om bepaalde aspecten van het probleem niet mee te nemen in de implementatie. Hieronder worden deze keuzes uitgelegd en gemotiveerd.

Alternatief vervoer

Bij het alternatief vervoer is geen rekening gehouden met de mogelijkheid van busvervoer. Het geheel uitbreiden met deze mogelijkheid is een puur technisch probleem. Het model is eerst opgebouwd zonder alternatief vervoer en later is gekozen om dit uit te breiden met de ‘taximogelijkheid’.

Deze keuze is om drie redenen gemaakt. De 'taximogelijkheid' is interessanter omdat dit vervoer op ieder tijdstip van de dag mogelijk is. Hierdoor wordt de oplossingsruimte waarin gezocht wordt veel groter, omdat er een soort flexibiliteit ingebouwd wordt. Busritten zijn relatief makkelijk te implementeren. Dit kan dus, wanneer gewenst, snel ingebouwd worden. Dit kan snel omdat bustaken vergelijkbaar zijn met andere taken, met andere woorden ze hebben een vaste tijd, plaats en traject. Binnen diensten is het meestal niet wenselijk om bustaken te hebben. De betrouwbaarheid is hiervoor te klein. Om deze reden worden bustaken voornamelijk aan het begin en einde van een dienst gepland. Wanneer er diensten gemaakt worden met taxitaken dan kan achteraf door de planners nog altijd bekeken worden of het wenselijk is om een taxirit te vervangen door een busrit.

Reserve diensten

Omdat het belangrijk is niet te veel veranderingen aan te brengen in het originele rooster, ligt de focus van het algoritme op het vinden van vervangende diensten. Aangezien reserve diensten geen vervangende dienst hoeven te hebben, worden deze niet meegenomen in het model. Wanneer het noodzakelijk is om reserve diensten te gebruiken kan dit aangepast worden op twee verschillende manieren. Er kan vooraf bepaald worden of bepaalde taken wel of niet opgenomen kunnen worden in nieuwe diensten. Met deze informatie kunnen er handmatig diensten worden toegevoegd aan de verzameling gegenereerde kolommen. Op deze manier kan een goede toegelaten oplossing gevonden worden. Ook kunnen er aan de input van oude diensten reserve diensten worden toegevoegd waarbij de extra taken binnen de werktijden van deze diensten vallen. Dit kan door van 1 of meerdere standplaatsen een reserve dienst mee te geven met de input.

5.3 Oplossen van het set-covering probleem

Voor het uiteindelijk selecteren van een dienst voor een machinist wordt een set-covering heuristiek gebruikt. Dit algoritme is een onderdeel van een al bestaand planningssysteem TURNI. Er bestaat al een interface tussen het gedeelte waar diensten worden gegenereerd en deze set-covering heuristiek uit TURNI. Op wat voor manier deze zoekt naar een goede oplossing is niet bekend.

Hoofdstuk 6

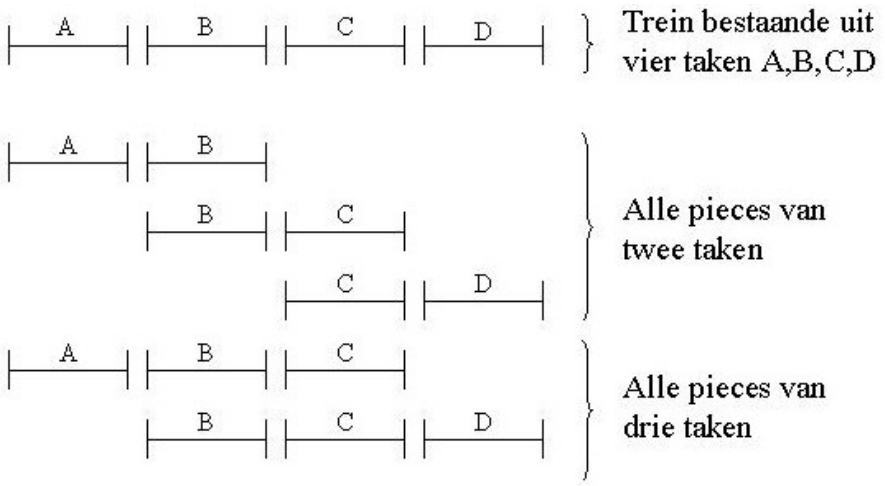
Implementatie

6.1 Netwerk en knopen

Uit alle taken worden voor het maken van het netwerk zogenaamde *pieces* gemaakt. Dit zijn samenstellingen van 1 of meer taken op dezelfde trein. Zo wordt uit een trein waarop vier taken tien pieces gemaakt. Let op dat hier de link tussen machinist en taak los gelaten wordt, en taken verbonden worden aan treinen. Vier samenstellingen van één taak, drie samenstellingen van twee taken etc. (zie figuur 6.1). Pieces hebben beginkenmerken van de eerste taak en eindkenmerken van de laatste taak.

Zoals in figuur 6.1 te zien is, zijn de opvolgende taken in een trein ook de opvolgende taken in de piece. In dit voorbeeld is het dus niet mogelijk een piece te hebben van taken A en D omdat deze elkaar in de dienst niet direct opvolgen. Deze pieces zullen in het netwerk gerepresenteerd worden als knopen. Dit komt doordat de toegestane paden op deze manier uit minder knopen zullen bestaan en dit heeft een positieve invloed op de rekentijd, zie [6]. Tijdens de constructie van de pieces worden deze ook topologisch gesorteerd op tijd. Nadat de pieces gemaakt zijn, worden deze op een specifieke manier gesorteerd.

Wanneer gezocht wordt naar een kortste pad in een netwerk dan vergt dit veel tijd. Wil men dit programmeren dan is het noodzakelijk om structuren te gebruiken waar zo min mogelijk in gezocht hoeft te worden. De structuur moet zoveel mogelijk 'direct access' zijn. Structuren als bomen of lijsten zullen dus niet goed functioneren gezien het feit dat voor iedere access er een search uitgevoerd moet worden. Dit leidt dan tot het gebruik van array's om het netwerk te representeren. Onderzoek heeft aangetoond dat de 'forward star' representatie het meest efficiënt is wanneer het gaat om



Figuur 6.1: Constructie van pieces van 2 en 3 taken uit 1 trein

netwerk representatie, zie [2]. De pijlen en knopen worden apart opgeslagen in een structuur. Een structuur om alle data van de pijlen in op te slaan en een structuur om alle knopen in op te slaan. Pijlen vanuit knopen A, B, C worden sequentieel opgeslagen. Pijlen uit een gelijke knoop worden willekeurig opgeslagen. De knopen staan in willekeurige volgorde waarbij knoop i een verwijzing bevat naar de $i - de$ sequentie van de structuur met pijlen. Wanneer knoop j geen uitgaande pijlen heeft dan bevat deze een gelijke verwijzing naar de pijlen als de $(j + 1) - ste$ knoop. Ook wordt er standaard een extra knoop toegevoegd die een verwijzing bevat naar een extra verzameling pijlen (er is dus een dummyknoop en een dummypijl). In de implementatie is deze structuur grotendeels aangehouden, maar omdat er hier sprake is van sorteerbare knopen is het anders op het punt van de willekeurigheid van de verzameling knopen.

Het abstracte netwerk wordt gesorteerd in drie dimensies, zie figuur 5.1. In de implementatie is er voor gekozen om dit in plaats van één dimensie ook in drie dimensies te programmeren. Deze keuze is gemaakt om voornamelijk twee redenen. Ten eerste is het intuïtief duidelijker wanneer de code het model 'namaakt'. Dit houdt het later aanpassen door derden makkelijker. Ten tweede zorgden de data ervoor dat voordat een 1-dimensionaal array gemaakt kon worden er toch eerst gebruik gemaakt moest worden van een object met drie dimensies; met andere woorden het was in ieder geval

een onvermijdelijke tussenstap naar gebruik van ieder type object. Een verschil met de modellering is dat de input, de verzameling van alle pieces, al gesorteerd is op tijd. Dit zorgt naast de sortering in de dimensies, voor een extra sortering die bij het uiteindelijke kortste-pad algoritme zorgt voor een snellere afhandeling.

Een nadeel van het gebruik van meer dimensionale array's voor de netwerk representatie is het feit dat er dan sprake kan zijn van een groot aantal lege entries. Om deze reden is er bij de implementatie gekozen voor het gebruik van twee objecten die beide een array bevatten. Om in deze objecten geen lege entries te krijgen wordt voordat de array's gevuld worden eerst het aantal elementen geteld, zodat de grootte hieraan aangepast kan worden. Het resultaat is dus een driedimensionaal array zonder null-waarden.

6.2 Pijlen

Als alle knopen gesorteerd en opgeslagen zijn, worden deze stuk voor stuk doorlopen om te zien of een knoop uitgaande pijlen heeft. Er wordt begonnen met het selecteren van een knoop. Hierna wordt er gekeken naar welke machinisten deze piece uit kunnen voeren en wat de eindlocatie is van deze piece. Op basis van de wegbekendheid kan iedere verzameling die geen standplaats overlap heeft met de piece buiten beschouwing gelaten worden. Binnen deze verzameling wordt alleen gezocht bij pieces waarvan de beginlocatie gelijk is aan de eindlocatie. Nu is er een verzameling pieces waarin enkel op tijd gezocht hoeft te worden voor mogelijke opvolgers van de knoop. Wanneer er een opvolger gevonden wordt, wordt er voor deze relatie een pijl gemaakt. Pijlen worden opgeslagen in een array waarbij pijlen die vertrekken uit dezelfde knoop gegroepeerd zijn. Wanneer een pijl tussen twee knopen bestaat omdat er een taxirit mogelijk is tussen deze knopen dan wordt in de pijl opgeslagen dat het een taxirit is.

6.3 Kortse pad algoritme

Na de constructie van het netwerk kan er gezocht worden naar diensten (paden) die de originele personeelsdiensten kunnen vervangen. Het volgende wordt voor iedere dienst een keer uitgevoerd: allereerst worden alle taken geselecteerd op basis van wegbekendheid. Er wordt gekeken vanaf welke standplaats de dienst begint. Aan de hand hiervan kunnen taken die niet uitgevoerd kunnen worden door machinisten van deze standplaats genegeerd worden tijdens het zoeken. Hierna worden alle mogelijke begintaken gese-

lecteerd die als eerste knoop kunnen dienen in het te vinden pad. Stuk voor stuk worden deze nu langsgelopen. Deze knopen worden zo gekozen dat ze binnen de toegelaten speling van 30 minuten vallen. Aan de hand van de begintijd van de knoop wordt bekeken hoe lang een dienst mag duren. Is deze tijd langer dan de duur van de oude dienst plus 30 minuten dan wordt de maximale duur van de te vinden dienst gelijk gesteld aan de duur van de oude dienst plus 30 minuten. Anders wordt de maximale duur vastgesteld op de maximale duur vanuit de gestelde eisen aan een dienst. Vanuit het beginpunt worden alle opvolgers bekeken en de kosten om deze opvolgers te bereiken worden onthouden. Wanneer een pijl om een van de opvolgers een pauze bevat dan wordt dit ook aangegeven bij dit punt. Deze handelingen worden nu herhaald voor iedere opvolger. Wanneer een knoop al een keer bekeken is, omdat een knoop een opvolger kan zijn van meerdere knopen, wordt er geselecteerd op kosten. Er is bekend hoeveel het kostte om deze knoop te bereiken vanuit het andere punt en deze kosten worden vergeleken met kosten vanuit het nieuwe punt. Wanneer de nieuwe kosten lager zijn, zullen de kosten om het punt te bereiken worden veranderd. Als de kosten gelijk zijn dan wordt er gekeken of één van de twee paden een pauze bevat en wordt er gekozen voor dat pad dat een pauze bevat. Dit proces wordt herhaald totdat er geen taken meer bereikt kunnen worden die vallen binnen de eindtijd van de oude dienst. In de volgende paragraaf wordt een voorbeeld gegeven van enkele stappen in dit proces.

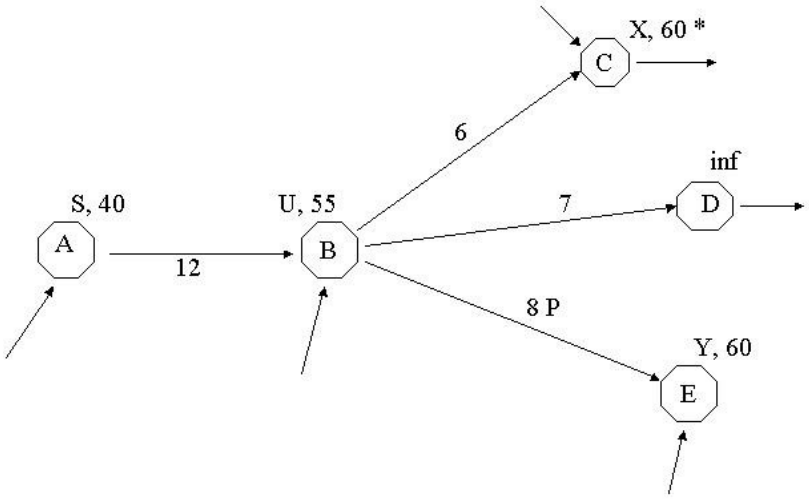
Nu zijn voor alle te bereiken knopen de kosten bekend van het kortste pad daar naar toe. Alle taken die eindigen op hetzelfde station als de oude te vervangen dienst representeren nu korte paden en één pad zal het kortste pad zijn.

Wanneer het zoeken afgelopen is, zijn de kosten van de paden naar ieder bereikbaar punt gevonden. Nu wordt er gekeken naar alle punten waar het laatste station gelijk is aan het laatste station van de oude dienst. Ook moet er gelden dat de paden tot deze punten een toegestane pauze bevatten. De selectie van deze paden is de uiteindelijke verzameling van kortste paden.

6.3.1 Voorbeeld van het kortste pad algoritme

In figuur 6.2 staat een gedeelte van een graaf waar het kortste pad in gezocht wordt. In iedere knoop staat de naam van de knoop. Naast een pijl staat een P als deze pijl een pauze bevat en de kosten van de knoop. Bij iedere knoop staat een getal of Inf. (infinite) dit zijn de kosten van het goedkoopste pad naar deze knoop tot nu toe. Wanneer er Inf. staat betekent dit dat de knoop nog niet bereikt is. Staat hier een letter en een getal, dan staat dat

voor de kosten van het kortste pad tot dit punt met als voorganger de knoop letter. Een ster naast dit getal houdt in dat het pad naar deze knoop een pauze bevat. De volgende uitleg wordt gezien vanuit het meest linker punt. Veranderingen worden in het rood aangegeven.



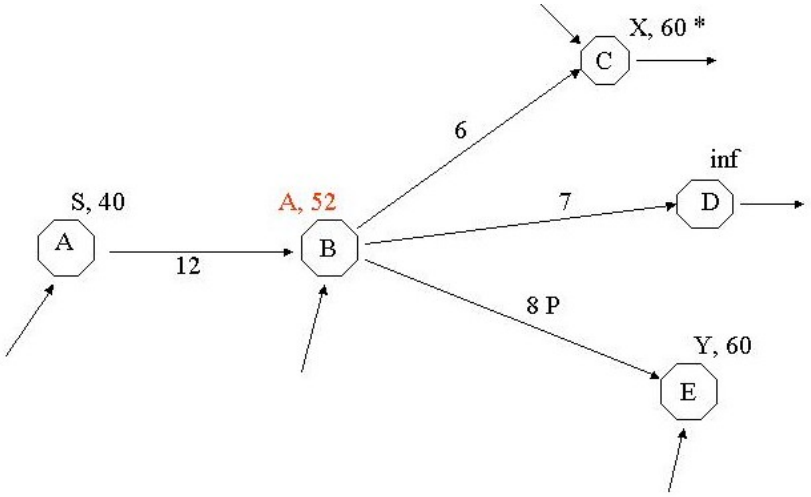
Figuur 6.2: Beginsituatie

Vanuit het meest linker punt kan alleen via de enige uitgaande pijl naar een andere knoop worden gegaan. Van deze knoop verandert de afstand tot die knoop van 55 naar 52. Deze 52 is de som van hoeveel het kost om de voorganger te bereiken (40) en de kosten van de knoop zelf (12). De voorganger van deze knoop wordt nu knoop A. Het pad hier naar toe bevat geen pauze, zie figuur 6.3. Vanuit het knoop B kunnen de knopen C, D en E bereikt worden. Deze knopen worden hier beneden behandeld, zie hiervoor ook figuur 6.4.

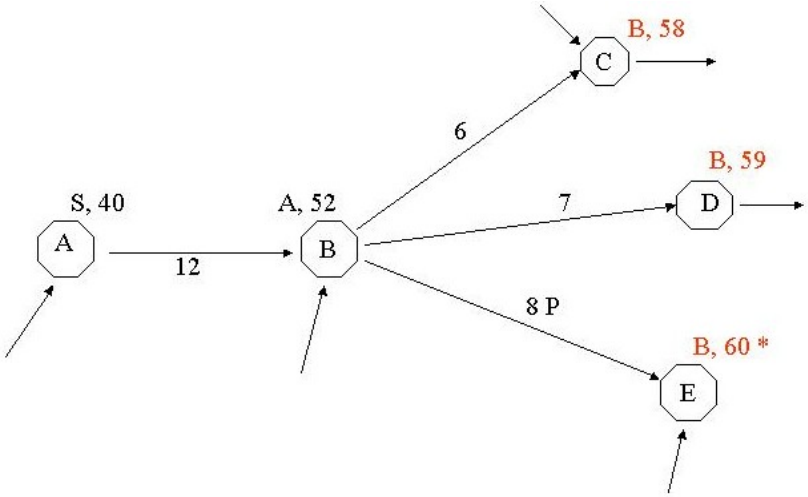
- Knoop C: De kosten om deze knoop te bereiken zijn 60 en dit pad bevat een pauze. Vanuit de huidige knoop zou het bereiken van deze knoop 58 kosten en dit pad bevat geen pauze. De kosten en voorganger worden nu vervangen

- Knoop D: Hier veranderen de kosten om deze knoop te bereiken van Inf. naar 59. De voorganger wordt ook knoop B.

- Knoop E: De kosten om deze knoop te bereiken zijn 60. Vanuit punt B



Figuur 6.3: Na de 1^e iteratie



Figuur 6.4: Na de 2^e iteratie

zouden de kosten ook 60 zijn. Omdat de pijl van B naar E een pauze wordt gekozen voor knoop B

6.4 Beperkingen

Bij de implementatie van het algoritme is een probleem met het aantal diensten dat gegenereerd kan worden. Het set-covering algoritme kan een beperkt aantal kolommen aan. In het hele proces van kolomgeneratie is ook een stap waarin kolommen verwijderd worden uit de verzameling gevonden kolommen. Om dit tot stand te brengen moest er te veel veranderd worden. De keuze van de opslag van de diensten was al gemaakt. Hierdoor kunnen er niet makkelijk diensten verwijderd worden. Het proces van verwijderen komt neer op het sorteren en verschuiven van veel data, deze acties zouden veel tijd kosten. Omdat tijd al de bottleneck is, is ervoor gekozen het verwijderen van kolommen niet te implementeren.

Een andere keuze die gemaakt is om het proces te versnellen is het aantal diensten waarvoor een vervanger gekozen wordt voordat er opnieuw gereduceerde kosten berekend worden. Idealiter zou na iedere generatie van een dienst nieuwe gereduceerde kosten berekend moeten worden. Omdat dit ook enige tijd kost is er voor gekozen om voor alle diensten op de dag een verzameling vervangers te zoeken en hierna de gereduceerde kosten opnieuw te berekenen. Dit betekent wel dat wanneer er in iedere iteratie in een vaste volgorde van oude diensten, diensten gegenereerd worden, de chronologisch eerste dienst altijd de kwalitatief beste vervangende diensten krijgt. Om er voor te zorgen dat in iedere iteratie het niet dezelfde dienst is waarvoor als eerste en laatste vervangers gezocht worden, worden de diensten willekeurig gepakt.

Hoofdstuk 7

Cases

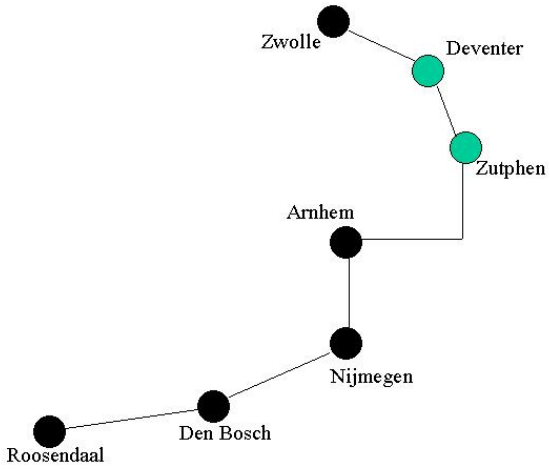
Om de werking van het programma te testen zijn er twee cases uit de praktijk uitgevoerd. De eerste eenvoudige case wordt in dit hoofdstuk uitgebreid beschreven. De ander zal in een iets beknoptere vorm behandeld worden.

7.1 Case Snippeling Aansluiting en Zutphen

Deze case betreft een buitendienststelling tussen Snippeling Aansluiting (Spa) en Zutphen (Zp). Snippeling Aansluiting is een locatie net onder Deventer waar de sporen uit Zutphen en Hengelo samenkomen.

7.1.1 Normale dienstregeling

Tussen stations Zutphen en Deventer rijdt volgens het jaarplan één treinserie, namelijk de sneltrein van Zwolle naar Roosendaal. In figuur 7.1 staat een schematische weergave van de route die de trein volgens het normale plan rijdt. In dit figuur zijn alleen de stations opgenomen waar een machinist de mogelijkheid heeft om te wisselen van trein.



Figuur 7.1: Schematische weergave route van sneltrein tussen Roosendaal en Zwolle

7.1.2 Buitendienststelling

De werkzaamheden vonden plaats van dinsdag 14 oktober 2003 01.32 uur tot vrijdag 17 oktober 2003 01.32 uur. Als gevolg van deze werkzaamheden was er geen treinverkeer mogelijk tussen Zutphen en Deventer. De sneltrein tussen Deventer en Zutphen werd door de buitendienststelling opgeheven. Ook de leeg materieelritten die over dit traject rijden werden opgeheven. Twee leeg materieelritten tussen Nijmegen en Zwolle worden omgeleid via Amersfoort en Utrecht. Daarnaast wordt Zutphen gebruikt als opstelterrein voor vroege starters en late eindigers in Deventer. Deze treinen worden tijdens de BDS omgelegd naar Zutphen. Tussen Deventer en Zwolle en tussen Zutphen en Roosendaal werd de normale dienstregeling aangehouden alleen werd er gekeerd op station Deventer en station Zwolle. In totaal waren er 42 diensten die direct geraakt werden door de BDS.

7.1.3 Invoer

De eisen aan de invoer zijn na het uitbreiden van het programma niet veranderd. De invoer is om deze reden al compleet, alleen de keuze welke NSRDAG diensten er meegenomen moeten worden staat nog open. Er is

Case	Spa - Zp
geraakte diensten	42
meegenomen diensten	203
aantal extra taken	172
totaal aantal taken	1980
aantal taxi mogelijkheden	4

Tabel 7.1: Samenvatting case Spa - Zp

voor gekozen om NSRDAG diensten mee te nemen van standplaatsen die in de buurt liggen van het traject uit figuur 7.1.

Er is hier sprake van een 'kleine' case waar weinig diensten geraakt worden. Omdat de BDS alleen gevolgen heeft voor enkele leeg materieel treinen en 1 treintraject, traject Roosendaal Zwolle is er voor gekozen om niet alle diensten van de dag mee te nemen. Er wordt bij deze case gewerkt met 203 diensten wat neer komt op ongeveer 20 procent van het totaal aantal diensten op een dag. Naast de grootte van de case zijn er nog twee redenen om niet alle diensten mee te nemen. Ten eerste leidt de reductie van het aantal diensten tot een snellere rekentijd. Ten tweede is er op basis van dit aantal diensten onderzoek gedaan naar personeelsplanning. Op deze manier kan er een betere vergelijking gemaakt worden tussen de twee methoden.

7.1.4 Case Spa - Zp samengevat

In tabel 7.1 staat een samenvatting van de BDS.

Hierin staat het aantal geraakte diensten voor de diensten die direct beïnvloed worden door de BDS. Een traject bevat zelf één of meerdere taken en een traject wordt één of meerdere keren per dag gereden. Het totaal aantal taken is het aantal taken dat ook echt uitgevoerd moet worden. Dit is dus het origineel aantal taken met hierbij opgeteld de extra taken en hiervan afgetrokken de vervallen taken.

7.2 25 september 2004

Naast de Spa - Zp case is er nog een andere case uitgevoerd. Dit is de case waar er onder andere BDS'en waren rond Amsterdam en Den Haag. Deze case is een stuk groter dan de eerste case. Alle diensten van de dag zijn dan ook meegenomen in deze case. De case (25 sept.) heeft echt te maken met grote BDS'en. Deze case is dus een grote instantie waarbij veel

Case	25 sept.
geraakte diensten	X
meegenomen diensten	801
aantal extra taken	1258
totaal aantal taken	8844
aantal taxi mogelijkheden	10

Tabel 7.2: Case 25 september 2004

veranderingen nodig zullen zijn. In tabel 7.2 staat een samenvatting van deze cases. Het symbool X geeft aan dat de waarde onbekend is.

In deze case is er sprake van veel BDS'en op 1 dag. Deze dag betrof zaterdag 25 september 2004. Op deze dag waren er BDS'en tussen Amsterdam en Utrecht en rond Den Haag Centraal. Ook waren er werkzaamheden tussen stations in Assen en Groningen, Den Bosch en Oss, Anna Palowna en Den Helder. Hier is dus de situatie dat er niet van een BDS op 1 plek maar op meerdere plekken verspreidt door het land sprake is.

Hoofdstuk 8

Resultaten

In dit hoofdstuk worden de resultaten van het programma gepresenteerd. In de eerste paragraaf worden de oplossingen voor de cases uit hoofdstuk 7 besproken. In de paragraaf die hier op volgt wordt de invloed van de verhouding van het aantal iteraties ten opzichte van het aantal gegenereerde kolommen behandeld.

8.1 Uitkomst standaard runs

In deze paragraaf worden de gevonden oplossingen besproken waarbij de parameters bij iedere case hetzelfde zijn. De kosten van een Jaarplan dienst zijn hier gelijk gesteld aan 1800. Om de output te sturen naar een oplossing waarin een redelijk aantal gegenereerde oplossingen worden gebruikt, zijn de kosten voor deze diensten niet veel hoger gekozen, namelijk 2000. Het is in de twee cases zeker nodig om extra diensten in te voegen. Om hier een compensatie tegenover te stellen hebben lege diensten lage kosten van 800 mee gekregen. Deze kosten worden gebruikt in het setcovering probleem dat opgelost wordt met een onderdeel van TURNI.

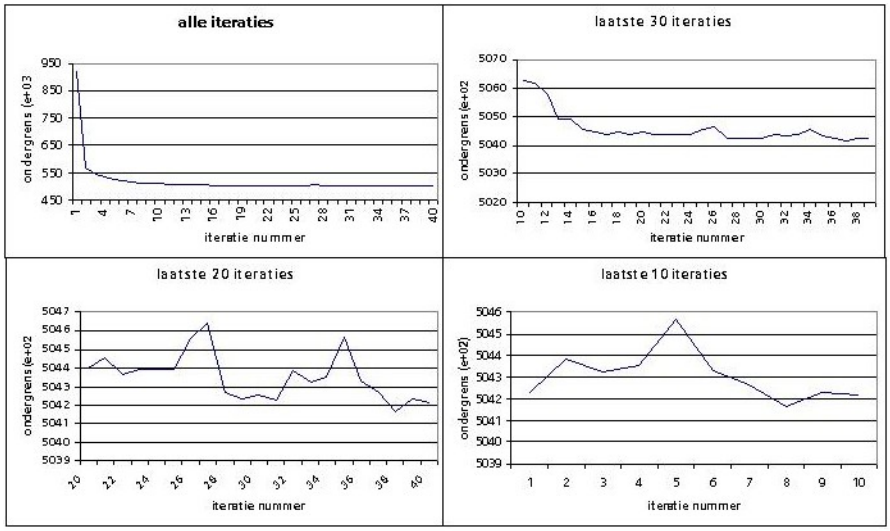
Wanneer de kolomgeneratie goed verloopt verwacht men bij iedere iteratie een scherpere ondergrens te krijgen. Met de initiële verzameling van diensten is een goede oplossing niet te verkrijgen, de berekende ondergrens zal dan ook hoog zijn. Naarmate er meer kolommen worden toegevoegd zal de ondergrens dalen en uiteindelijk naderen tot het optimum van de Lagrange Relaxatie. Dit optimum is dan de scherpste ondergrens voor het originele probleem. In de volgende bespreking van de cases zal de focus ook liggen op deze ondergrens. Hoewel het belangrijk is wat de uitkomst van de cases is, is de focus op deze verloop van de ondergrens belangrijk. De

wiskundige technieken leveren, binnen de gestelde grenzen, zeker een optimum voor de relaxatie en het is daarom belangrijk hoe het proces richting het optimum verloopt.

De berekening van deze ondergrens gebeurt wel op een snelle heuristische manier. Hierdoor zullen de grafieken van de ondergrenzen een grilliger verloop tonen dan wanneer de ondergrens heel precies berekend zou worden.

8.1.1 Case Spa - Zp

Bij deze case is er een run gedraaid van 40 iteraties, waarbij in iedere iteratie 15 kolommen werden gegenereerd. Hierna werd de output als input gegeven aan het set-covering algoritme. Deze heeft 6 uur gedraaid. In figuur 8.1 staat het verloop van de ondergrens gedurende de 40 runs.



Figuur 8.1: Verloop ondergrens

Duidelijk te zien in de bovenste twee grafieken is dat er sprake is van convergentie.

Oplossing

De uiteindelijke oplossing van deze case is als volgt. In het totaal zijn er 207 machinisten nodig. Van de machinisten kunnen 131 machinisten de gewone Jaarplan diensten uitvoeren. In totaal zijn er dus 76 diensten gewijzigd

Spa - Zp	TURNI*	NS*	Kolomgeneratie
aantal diensten	209	215	207
aantal gewijzigde diensten	95	79	74
aantal extra diensten	7	5	4
aantal NSRDAG diensten	7	5	2
aantal taxi- en busritten	11	6	11
rekentijd in uren	15	28	2

Tabel 8.1: Resultaten case Spa-Zp. *Gegevens afkomstig uit [1].

waarvan er 71 zijn gegenereerd met het algoritme voor het pricing probleem. Van de overige diensten zijn er 3 waarbij de verandering zeer klein is, zoals een verschuiving van enkele minuten binnen de Jaarplan dienst, en 2 diensten zijn NSRDAG diensten. Deze reserve diensten zijn handmatig toegevoegd aan de totale pool van diensten.

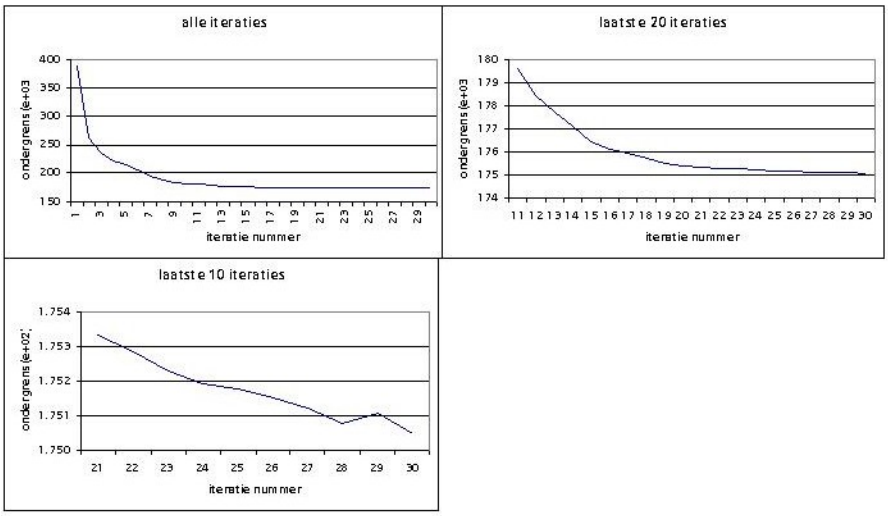
De resultaten van deze case staan samen met eerder verkregen resultaten in tabel 8.1. De resultaten die vermeld staan onder TURNI zijn de resultaten verkregen uit het onderzoek naar de toepasbaarheid van TURNI op dit probleem, zie [1]. Onder NS staat het gerealiseerde resultaat door de planners van Dagplan.

In de tabel is duidelijk te zien dat binnen de parameters van het programma, de uitkomst van de kolomgeneratie de beste is. Op alle punten is deze oplossing beter dan of gelijk aan de gerealiseerde oplossing van de planners. Ook de resultaten van de met TURNI gemaakte instantie zijn minder goed dan de resultaten van kolomgeneratie. Bij TURNI moet wel gezegd worden dat dit de run is met het minst aantal wijzigingen op het Jaarplanrooster. Er waren ook personeelsdiensten gemaakt waarbij het aantal diensten gelijk was aan 207 maar het aantal gewijzigde diensten lag dan zeker boven de 130. Qua rekentijd en bus- en taxiritten is deze output representatief voor de andere runs die met TURNI zijn uitgevoerd.

8.1.2 Case 25 september 2004

Bij deze case is er een run gedraaid van 30 iteraties, waarbij in iedere iteratie 15 kolommen werden gegenereerd. Hierna werd de output als input gegeven aan het set-covering algoritme. Deze heeft 6 uur gedraaid. In figuur 8.2 staat het verloop van de ondergrens gedurende de 30 runs.

In de bovenste twee grafieken is er sprake is van convergentie. Dit is minder duidelijk dan in de kleinere Case Spa-Zp maar nog wel zichtbaar.



Figuur 8.2: Verloop ondergrens

Worden de laatste iteraties nader bekeken dan zien we dat de tussen de 20ste en 30ste iteratie nog duidelijk sprake is van een dalende trend. Dit wijst er op de afstand tot het optimum nog niet zo klein is dat er geen echte verbeteringen meer mogelijk zijn. Omdat bij de iteraties het maximum aantal gegenereerde diensten is vastgesteld op een half miljoen is er niet verder gegaan met itereren. In het onderstaande tabel staan de 30 berekende ondergrenzen. Wanneer wij de waarden bekijken van deze ondergrenzen kan dezelfde conclusie getrokken worden. Zonder de vertekening van schaling van de grafiek lijken deze waarden te laten zien dat convergentie bijna bereikt is. In deze tabel is ook duidelijk te zien dat na 18 iteraties er convergentie optreedt.

Oplossing

De uiteindelijke oplossing van deze case is als volgt. In totaal zijn er 806 machinisten nodig. Van de machinisten kunnen 248 machinisten de gewone Jaarplan diensten uitvoeren. In totaal zijn er dus 558 diensten gewijzigd waarvan er 496 zijn gegenereerd met het algoritme voor het pricing probleem. Van de overige diensten zijn er 62 waarbij de verandering zeer klein is, zoals een verschuiving van enkele minuten binnen het gewone rooster.

De resultaten van deze case staan samen met eerder verkregen resultaten

1 - 10	11 - 20	21 - 30
388.388	179.606	175.333
261.775	178.465	175.283
236.431	177.714	175.231
223.252	177.064	175.191
213.559	176.434	175.179
205.626	176.128	175.149
196.304	175.922	175.122
189.014	175.712	175.077
183.857	175.487	175.107
181.451	175.387	175.047

Tabel 8.2: Waarden ondergrens

25 sept.	NS	Kolomgeneratie
aantal diensten	824	806
aantal gewijzigde diensten	630	558
aantal extra diensten	23	5
aantal NSRDAG diensten	18	0
aantal taxi- en busritten	X	34
reken tijd in uren	X	32

Tabel 8.3: Resultaten case 25 september 2004

25 sept.	beginoplossing
aantal diensten	820
aantal gewijzigde diensten	484
aantal extra diensten	16
aantal NSRDAG diensten	2
aantal taxi- en busritten	34
aantal lege diensten	19

Tabel 8.4: Uitkomsten van de beginoplossing

in tabel 8.3. Onder NS staat het gerealiseerde resultaat door de planners van Dagplan. De waarden van de velden waar een X staat zijn onbekend.

8.1.3 Uitkomst standaard run op basis van bestaande tool

Vergelijking

De oplossing besproken in de vorige paragrafen zijn alle gegenereerd vanuit een triviale oplossing. Voor de ontwikkeling van de tool bestond er al een tool die via het kleiner maken van de oplossingsruimte een goede oplossing vond.

De volgende run van case 25 sept. is gestart met de oplossing die al bestond. In tabel 8.4 staan de resultaten van deze oplossing wanneer er verder geen diensten gegenereerd worden. In totaal zijn er ongeveer 20 duizend meer diensten in de begin verzameling dan bij de triviale start. Er zijn bij deze run per iteratie 27 diensten gegenereerd voor één originale dienst en in totaal zijn er 20 iteraties uitgevoerd. De vergelijking met de eerste run van deze case zal dan ook uitgevoerd worden met de laatste 20 iteraties van de eerste run.

De diensten die zijn toegevoegd voldoen niet allemaal aan de eisen van een veranderde dienst. Enkele zijn niet toegelaten. Om deze reden is te verwachten dat de oplossing beter zal zijn dan de eerder gevonden oplossing waarbij alleen puur toegelaten diensten gegenereerd werden. Om deze reden gaat het bij deze run, wanneer het vergeleken wordt met de run uit de vorige paragraaf, voornamelijk om de snelheid waarmee het optimum bereikt wordt.

In figuur 8.3 staat het verloop van de gevonden ondergrenzen. Het eerste dat opvalt is de beginwaarde van de ondergrens. Zoals gezegd is deze al een stuk scherper dan bij de run zonder goede begin oplossing. Wanneer we de

eerste grafiek van het figuur bekijken zien we dat deze een sterkere convergentie vertoond dan de tweede grafiek van figuur 8.2. Dit is te verklaren doordat de er in de nieuwe run nog kolommen toegevoegd worden die een sterke invloed hebben op de uitkomst. In de eerste run, zonder een toegelaten oplossing, kunnen deze kolommen al eerder toegelaten zijn.

Om een betere vergelijking te kunnen maken tussen deze twee runs staan in figuur 8.4 van beide grafieken het verschil tussen de opvolgende waarden. Hier zien we ook dat de tweede run sneller convergeert. Verder kan men afleiden dat de convergentie in beide gevallen na iteratie 3 ongeveer gelijk verloopt. Om deze reden kan geconcludeerd worden dat beide runs een vergelijkbare kwaliteit hebben tot opzichte van hun optimum. Wel lijkt de eerste run dichter bij het optimum te liggen. De daling in de tweede run in de laatste 10 iteraties is net iets significanter.

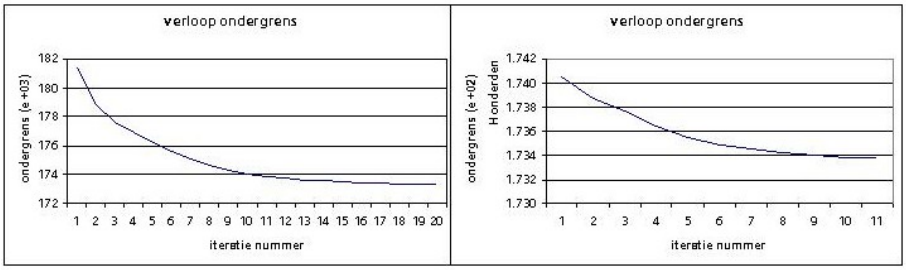
Wanneer we naar de uitkomsten kijken, tabel 8.5, is het niet moeilijk te zien dat de run met een goede startoplossing een beter resultaat geeft. Op alle fronten is deze oplossing beter. De rekentijd staat niet vermeld in de tabel omdat deze niet helemaal bekend is. De rekentijd voor het vinden van een goede start oplossing is ongeveer 10 uur. Dat de oplossing beter is, is op zich geen verrassing maar een verschil van drie diensten is zeker wel opvallend. Het is moeilijk dit verschil te verklaren omdat het als geheel samenhangt.

Op grond van deze resultaten kan geconcludeerd worden dat het verschil van aanpak in kwaliteit niet erg groot is. De tijd die nodig was met de originele tool om de verzameling diensten te maken is ongeveer gelijk aan de tijd die het kost om de 10, hier weggelaten, runs uit te voeren. De oplossing van de run waarbij de grotere verzameling diensten wordt gebruikt is wel beter. Dit om redenen die al besproken zijn. Omdat deze uitkomst een betere waarde heeft, geniet het de voorkeur boven de aanpak waarbij de kolommen gegenereerd worden vanuit een triviale oplossing. Naast deze reden is er nog een reden om voorkeur te geven aan de oplossingsmethode waarbij er wordt begonnen met een goede start oplossing. Bij het generatie proces worden alleen toegelaten diensten gegenereerd. In praktijk zal binnen de oplossing altijd een beetje speling zijn ten opzichte van de regels. Ook eigenlijk niet toegestane diensten zullen in de praktijk mogelijk zijn. Wanneer er voor een beginoplossing ook niet toegelaten diensten worden toegevoegd kan de oplossing verbeterdworden. In het geval van case 25 sept. zagen wij dat de toevoeging van niet toegelaten diensten in de oplossing zorgde voor 3 machinisten minder dan wanneer deze diensten niet toegevoegd waren. Op dit punt kan in de planning dus gevarieerd worden met niet toegelaten diensten. Op deze manier kunnen er meer afwegingen gemaakt worden voor

25 sept.	triviale startoplossing	goede startoplossing
aantal diensten	806	803
aantal gewijzigde diensten	496	476
aantal extra diensten	5	2
aantal NSRDAG diensten	0	1
aantal taxi- en busritten	34	33
aantal lege diensten	47	43

Tabel 8.5: Uitkomsten runs met en zonder goede beginoplossing

de uiteindelijke oplossing.

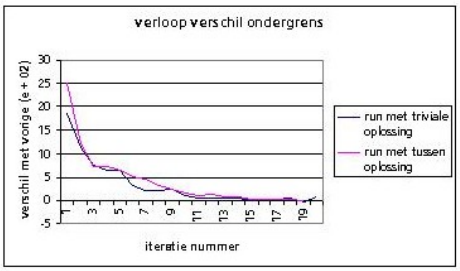


Figuur 8.3: Verloop ondergrens

8.2 Aantal iteraties versus aantal kolommen

Binnen de programmatuur is er een maximaal aantal diensten dat gegenereerd kan worden. De set-covering heuristiek uit TURNI heeft een grens van maximaal 1 miljoen kolommen. De snelheid van deze heuristiek is ook afhankelijk van het aantal kolommen. Om een redelijke rekentijd te houden is er voor gekozen het totaal aantal kolommen niet groter te maken dan een half miljoen. Voor de uitvoering is te berekenen hoeveel kolommen er per iteratie gegenereerd kunnen worden om binnen dit aantal van een half miljoen te blijven. In theorie zijn er ook mogelijkheden om kolommen te verwijderen uit de verzameling diensten maar door de bestaande programmatuur zal dit de rekentijd negatief beïnvloeden of moeten er grote aanpassingen gedaan worden aan de code.

Om nu een goede afweging te kunnen maken in de verhouding van het



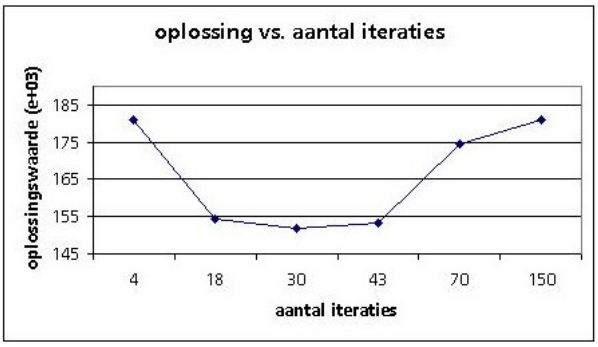
Figuur 8.4: Verloop verschil ondergrens

aantal iteraties ten opzichte van het aantal te genereren kolommen worden in deze paragraaf de uitkomsten vergeleken met verschillende verhoudingen. Deze vergelijking is uitgevoerd met de case 25 sept.

Er is in deze case voor gekozen om zes runs uit te voeren met in iedere run een andere verhouding tussen het aantal kolommen en het aantal iteraties. In figuur 8.5 staan de criteriumwaarden uitgezet tegen het aantal iteraties. Wat als eerst opvalt is het slechte resultaat wanneer er veel iteraties uitgevoerd worden. Verder is er een middenstuk waarin de kwaliteit niet veel veranderd. Het verloop aan het begin van het figuur is niet verrassend. Doordat de aanpak een iteratieve aanpak is, is er een minimum aan aantal iteraties dat uitgevoerd moet worden wil de optimale waarde bereikt worden. Bij een te klein aantal iteraties wordt dit minimum aantal niet bereikt waardoor de oplossing minder goed zal zijn.

Het verloop aan het einde van het figuur is te verklaren uit het feit dat bij een groot aantal iteraties maar weinig diensten per iteratie gegenereerd kunnen worden. Kolommen die aan het begin van de run een grote meerwaarde hebben kunnen aan het einde helemaal geen waarde meer hebben voor de oplossing. Wanneer er minder iteraties uitgevoerd worden, worden er in iedere iteratie per dienst meer diensten gegenereerd. Deze extra diensten hebben op het moment van genereren een lage toegevoegde waarde maar in het verloop van het proces kunnen juist deze diensten de oplossing doen verbeteren. Wanneer aan het einde van iedere iteratie ook een verwijderingsstap zou zijn, dan kon dit tegengegaan worden. Een ander probleem dat voortkomt uit dit gegeven is het feit dat niet na iedere dienstgeneratie de gereduceerde kosten opnieuw worden berekend. De diensten die als laatst gegenereerd worden in een iteratie zijn wel diensten met lage gereduceerde kosten maar deze kosten zijn niet gebaseerd op de meest recente situatie.

Uit dit resultaat kan geconcludeerd worden dat het aantal iteraties niet erg groot of erg klein gekozen moet worden. We kunnen zien uit de verkregen resultaten uit de vorige paragrafen dat een aantal iteraties van 35 zeker voldoende is voor een grote case. Dit aantal iteraties is, ook gezien figuur 8.5, een goed aantal om mee te beginnen wanneer er een nieuwe case uitgevoerd wordt. Is de case erg klein zoals het geval in de case Spa - Zp dan kunnen er minder runs gedraaid worden, 20 is hier zeker een voldoende groot aantal.



Figuur 8.5: Oplossingswaarde bij verschillend aantal iteraties

Hoofdstuk 9

Conclusie

In hoofdstuk 1 is beschreven wat de aanleiding is van dit onderzoek. Hierin werd de probleemstelling als volgt beschreven:

Modelleer en implementeer binnen de bestaande tool en de gebruikte theorie, een algoritme dat consistent is in het genereren van diensten voor machinisten, waarbij voornamelijk gelet moet worden op de veranderingen aan de originele personeelsdiensten. Bovendien moet er een manier zijn waarin de kwaliteit van de gevonden oplossing te beoordelen is.

In dit verslag is een algoritme beschreven dat een oplossing genereert voor het probleem. Dit algoritme geeft binnen de gestelde beperkingen hierop een scherpe ondergrens voor het probleem. De implementatie van het algoritme heeft enkele beperkingen maar geeft als uitvoer nog een goede oplossing.

Er bestond al een tool die een goede oplossing kon genereren alleen was deze oplossing afhankelijk van de gebruiker. Het gebruikte model heeft de invloed van de gebruiker tot een minimum gereduceerd. Met verschillende startsituaties worden er vergelijkbare oplossingen gegenereerd. Verschillen die optreden zijn allen te verklaren vanuit praktische verschillen in de startsituatie. Om deze reden kan gezegd worden dat de tool consistent is geworden in het genereren van diensten. Beter gezegd: vanuit iedere startsituatie wordt een bijna gelijke oplossing gevonden.

Voor het verkrijgen van inzicht in de prestaties van het geïmplementeerde model zijn twee testcases uitgevoerd. Het betrof hier een dag met één kleine BDS tussen Snippeling Aansluiting en Zutphen en een dag met meerdere grote BDS'en. Voor beide cases geeft het programma binnen redelijke tijd oplossingen. Deze oplossingen hebben betere prestatie indicatoren dan oplos-

singen uit vorig onderzoek en gerealiseerde oplossingen. Van deze prestatie indicatoren is er één die het aantal veranderingen in de originele dienst weergeeft. Deze indicator had in alle gevallen tot nu toe de laagste waarde; deze indicator is gezien de probleemstelling het belangrijkst.

Op de opgeleverde programmatuur bestond enkel de eis dat dit binnen de bestaande tool moest komen. Verder zijn hier geen eisen over gesteld. De prestatie van de programmatuur kan uitgedrukt worden in tijd. De oplossing van de 'kleine' case is binnen 2 uur gevonden wat een goede tijdsduur is. Voor de tweede en grotere case was meer tijd nodig. Na 24 uur gaf het algoritme een oplossing die tot nu toe het beste is.

Het laatste punt is de beoordeling van de kwaliteit van de oplossing. Door enkele aannames zal het algoritme geen optimale oplossing geven. Het plaatsen van een pauze in een dienst is een moeilijk probleem en om de rekentijd binnen de perken te houden is er voor gekozen om het plaatsen van een pauze in een dienst niet uit te voeren. Binnen deze beperking van het model bestaat wel een mogelijkheid om de oplossing te beoordelen, namelijk via de ondergrens. De ondergrens van het probleem zal, na voldoende iteraties, de optimale oplossing (binnen de gestelde grenzen) benaderen. De snelheid waarmee de ondergrens de optimale oplossing benadert zal met iedere iteratie afnemen. De ondergrens, uitgezet tegen het aantal iteraties, zal dus convergeren naar een bijna optimale oplossing. Door dit verloop van de ondergrens te bepalen is het mogelijk uitspraken te doen over de mate van convergentie en hiermee uitspraken te doen over de kwaliteit van de oplossing.

Hoofdstuk 10

Aanbevelingen

Na veel onderzoek naar dit probleem lijkt hier een methode gevonden te zijn die geschikt is om op dit probleem toe te passen. De eerste testcases laten goede oplossingen zien, maar het onderzoek houdt hier niet op. Er kan in drie richtingen nog onderzoek gedaan worden.

Technisch onderzoek is mogelijk. Er zijn in deze case enkele beperkingen gelegd op het algoritme. Er is nu sprake van een maximaal aantal diensten dat gegenereerd kan worden. Een goede uitbreiding op de gehele tool zou een methode zijn waarbij eerder geselecteerde diensten verwijderd worden uit de oplossing. Diensten die in eerste instantie een meerwaarde hebben voor de oplossing kunnen deze waarde gaandeweg in het proces verliezen. Door deze diensten weer te verwijderen wordt de grens van het maximaal aantal diensten niet of later bereikt.

Theoretisch is er onderzoek mogelijk naar de invloed van het eerder aanpassen van de gereduceerde kosten. Hier is gekeken naar runs waarbij in iedere iteratie voor iedere dienst een verzameling vervangers gezocht wordt. In het geval van duizend diensten is het de moeite waard om te kijken wat er verandert als in één iteratie maar voor 100 of 200 diensten een verzameling vervangers wordt gezocht. Ieder 5 of 10 iteraties zullen dan voor alle diensten een verzameling vervangers gezocht zijn. Dit punt is interessant omdat de laatst gegenereerde dienst in een iteratie de minst recente informatie heeft.

Praktisch onderzoek kan gedaan worden naar de invloed van verschillende factoren. Het onderzoek in dit verslag is toegespitst op de methode. Nu kan er door variatie in gewichten gekeken worden welke waarde van de parameters de beste oplossingen leveren.

Bibliografie

[1] F. Breed. *Personeelsplanning.* afstudeerscriptie, Vrije Universiteit Amsterdam, 2004.

[2] J.B. Orlin R.E. Tarjan B.V. Cherkassky, K. Mehlhorn. Shortest paths algorithms: Theory and experimental evaluation. Technical report, Computer Science Department, Stanford University, 1993.

[3] G.L. Nemhauser M.W.P. Savelsbergh P.H. Vance C. Barnhrt, E.L. Johnson. Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316 – 329, 1998.

[4] G.L. Nemhauser en L.A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, New York, New York, US, 1988.

[5] D. Huisman. Crew scheduling for a particular day, modifying duties when some tracks are outside service. Presentatie Dagstuhl conferentie, juni 2004.

[6] D. Huisman. Integrated and dynamic vehicle and crew scheduling. Proefschrift, Erasmus Universiteit Rotterdam, 2004.