11

Het samenstellen van toetsen

Bij het samenstellen van toetsen kunnen we te maken krijgen met drie soorten eisen: psychometrische, inhoudelijke en praktische eisen. De psychometrische eisen zullen veelal betrekking hebben op de gewenste meetnauwkeurigheid van de samen te stellen toetsen. Met inhoudelijke eisen worden de vakinhoudelijke en onderwijskundige eisen bedoeld: de verdeling van de vragen over de leerstofcategorieën, de gewenste moeilijkheidsgraad van de toets en dergelijke. Ook relaties op itemniveau kunnen een rol spelen bij het samenstellen van toetsen. Als bijvoorbeeld het antwoord op item 4 een aanwijzing bevat voor de antwoorden op item 16 en item 400, dan kan de toetsconstructeur eisen dat als item 4 in de toets wordt opgenomen, item 16 en item 400 niet meer worden opgenomen. Onder praktische eisen worden die aspecten van toetsconstructie verstaan die psychometrische noch inhoudelijke betekenis hebben, maar bij het samenstellen van toetsen wel degelijk een rol spelen. Een voorbeeld is de tijd die voor het afnemen van een toets beschikbaar is. Aangezien die tijd niet onbeperkt is, zal men hiermee bij het samenstellen van een toets rekening moeten houden. Een ander voorbeeld betreft het budget dat beschikbaar is om een toets te kunnen afnemen. Een bepaald budget zou kunnen betekenen dat niet meer dan drie beoordelaars ingeschakeld kunnen worden.

In dit hoofdstuk laten we zien hoe met behulp van wiskundige modellen toetsen samengesteld kunnen worden die voldoen aan de psychometrische, inhoudelijke en praktische specificaties van toetsconstructeurs. De modellen zijn ontleend aan een tak van de wiskunde, aangeduid met operationele research of mathematische programmering, die als doel heeft het ontwikkelen van modellen ter ondersteuning van besluitvorming. De eerste paragraaf van dit hoofdstuk bevat een beknopte bespreking van mathematische programmering. De drie volgende paragrafen bevatten toepassingen van mathematische programmering binnen de itemresponstheorie, de klassieke testtheorie en de generaliseerbaarheidstheorie.

11.1 Mathematisch programmeren

Stel, iemand is op expeditie in Groenland. De bagage wordt vervoerd op een hondenslede waar nog genoeg ruimte over blijft om een paar extra dingen mee te nemen om onderweg in de handelspost te verkopen. De reiziger heeft nog een doos met tien literblikken ananas, een doos met twintig literblikken hondevoer en een jerrycan met twintig liter benzine. In de handelspost is men bereid tweehonderd Groenlandse kronen te betalen voor de ananas, honderd voor het hondevoer en honderd voor de benzine. De doos ananas weegt dertig kilo, het hondevoer veertig kilo en de benzine twintig kilo. Op de hondenslede is nog plaats voor veertig liter extra bagage. De honden mogen echter niet meer trekken dan zestig kilo. Het probleem van onze reiziger is nu, te beslissen welke dingen hij moet meenemen zodat hij de meeste opbrengst in de handelspost heeft. We zullen laten zien hoe modellen voor dit soort problemen geformuleerd worden binnen de mathematische programmering en hoe deze problemen vervolgens opgelost worden.

Het besluit om een bepaald produkt mee te nemen kunnen we voorstellen door een zogenaamde beslisvariabele. Deze variabele neemt de waarde 1 aan als het desbetreffende produkt wordt meegenomen en de waarde 0 als het produkt niet wordt meegenomen. Variabelen die alleen waarden 0 en 1 kunnen aannemen, worden binaire variabelen genoemd. Noemen we de beslisvariabele die betrekking heeft op het meenemen van de benzine x_1, het meenemen van de ananas x_2 en van het hondevoer x_3, dan kunnen we de opbrengst uitdrukken als $100\,x_1 + 200\,x_2 + 100\,x_3$. Deze functie wordt de doelfunctie genoemd. Het totale volume van de mee te nemen produkten wordt uitgedrukt als $20\,x_1 + 10\,x_2 + 20\,x_3$ en het totale gewicht als $20\,x_1 + 30\,x_2 + 40\,x_3$. Het doel van de reiziger is een zo hoog mogelijke opbrengst te realiseren, terwijl de beperkingen ten aanzien van volume en gewicht niet worden overschreden. Deze beperkingen worden de restricties genoemd. De verzameling van alle beslissingen die toegelaten zijn, dat wil zeggen beantwoorden aan de restricties, heet de oplossingsruimte. Het model voor het probleem van de reiziger kunnen we nu formuleren als:

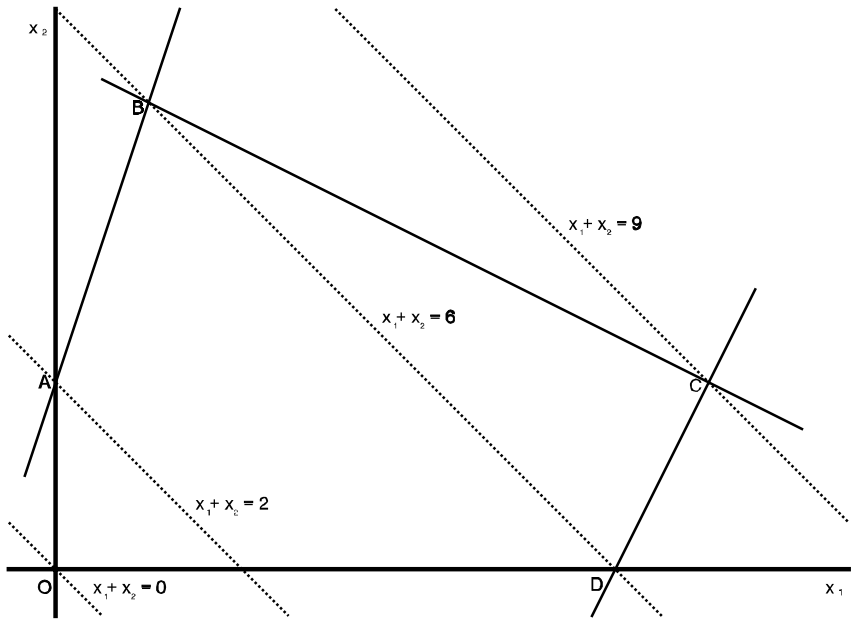
maximaliseer	$100\,x_1 + 200\,x_2 + 100\,x_3$	(opbrengst)
onder voorwaarde dat	$20\,x_1 + 10\,x_2 + 20\,x_3 \leq 40$	(volume)
	$20\,x_1 + 30\,x_2 + 40\,x_3 \leq 60$	(gewicht)
	$x_1,\ x_2,\ x_3\ \varepsilon\ \{0,1\}$.	(binaire variabelen)

Modellen waarvan de doelfunctie en alle restricties lineair zijn en alle beslisvariabelen continu, noemen we lineaire programmeringsmodellen. Wanneer de beslisvariabelen geen continue maar binaire variabelen zijn, zoals in ons reizigersprobleem, dan spreken we van binaire programmeringsmodellen.

Een populaire oplosmethode voor lineaire programmeringsmodellen is de simplexmethode. Om een grafische illustratie van de methode mogelijk te maken, nemen we een voorbeeld met twee variabelen. Het model voor het voorbeeld luidt:

maximaliseer $x_1 + x_2$ (11.1)

onder voorwaarde dat
$2x_1 - x_2 \leq 12$ (11.2)
$x_1 + 2x_2 \leq 11$ (11.3)
$-3x_1 + x_2 \leq 2$ (11.4)
$x_1, x_2 \geq 0.$ (11.5)



Figuur 11.1
Voorbeeld van de simplexmethode

De oplossingsruimte wordt hier gegeven door ongelijkheden (11.2)-(11.5) en wordt voorgesteld door de veelhoek OABCD in figuur 11.1. Zo wordt restrictie (11.2) weergegeven door het gebied links van de lijn CD, restrictie (11.3) door het gebied onder de lijn BC, restrictie (11.4) door het gebied rechts van de lijn AB en restrictie (11.5) door het gebied rechtsboven het assenkruis in figuur 11.1. De hoekpunten van

de oplossingsruimte (hier O, A, B, C en D) worden ook wel extreme punten genoemd. De oplossing die correspondeert met een extreem punt wordt een basisoplossing genoemd. Lineaire programmeringsproblemen hebben de eigenschap dat er altijd een optimale oplossing kan worden gevonden in de groep van basisoplossingen. Van deze eigenschap wordt door de simplexmethode gebruik gemaakt door op een systematische manier de groep van basisoplossingen af te zoeken. In ieder extreem punt zijn slechts twee restricties actief, dat wil zeggen dat aan twee ongelijkheden met een strikte gelijkheid wordt voldaan. Uitgaande van een extreem punt zoekt de simplexmethode steeds een naburig extreem punt op met een hogere doelfunctiewaarde. Dit gebeurt in de grafiek in figuur 11.1 door de doelfunctie (11.1) evenwijdig aan zichzelf naar rechts te verschuiven. In figuur 11.1 start de simplex in punt O. Hier zijn de beide niet-negativiteitseisen (11.5) actief. Punt O heeft twee naburige extreme punten: A en D. Zij hebben beide een hogere doelfunctiewaarde dan punt O. De simplex kiest het gunstigste naburige extreme punt, en verwisselt daarmee steeds één actieve restrictie door een andere. De simplex gaat steeds door naar een naburig punt totdat punt C, het optimum, bereikt wordt. De simplexmethode stopt zodra een extreem punt is gevonden met alleen naburige extreme punten die een lagere doelfunctiewaarde hebben. Zolang de simplexmethode van ieder extreem punt alleen naar een hoger gelegen extreem punt kan gaan, zorgt het feit dat een oplossingsruimte slechts een eindig aantal extreme punten heeft ervoor dat het optimum ook daadwerkelijk wordt bereikt.

Problemen in de praktijk zijn vaak complexer dan het probleem in dit voorbeeld, maar de simplexmethode zoekt nog steeds op ongeveer dezelfde manier de basisoplossingen af. Uitbreiding naar meer dan twee beslisvariabelen en daarmee samenhangend uitbreiding naar meer dan twee dimensies is niet eenvoudig in te zien. Er zijn nu geen twee actieve restricties, maar evenveel als er dimensies zijn. Tevens neemt het aantal basisoplossingen sterk toe bij toenemende dimensionaliteit. In bijvoorbeeld Dirickx, Baas en Dorhout (1987) vindt men een uitgebreide beschrijving van de simplex voor problemen met meer variabelen, alsmede de andere technieken die in dit hoofdstuk aan de orde komen.

Branch-and-bound methode

De oplosmethode voor binaire programmeringsmodellen is eveneens gebaseerd op de simplexmethode. De geheeltalligheidseisen ($x_j \in \{0,1\}$) worden gerelaxeerd, dat wil zeggen dat ze vervangen worden door de restricties $0 \leq x_j \leq 1$. Het zo ontstane continue probleem wordt vervolgens opgelost met behulp van de simplexmethode. Is

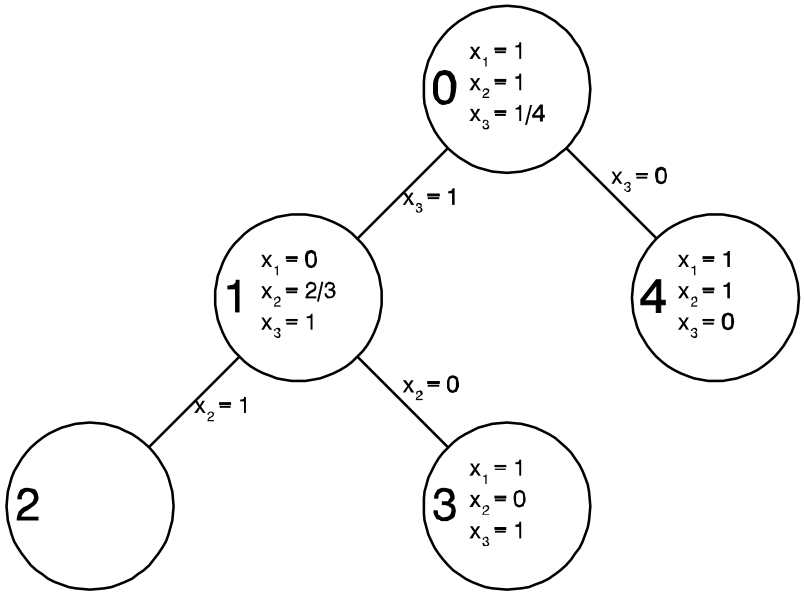
de optimale oplossing geheeltallig, dan is de optimale oplossing voor het continue probleem tevens de oplossing voor het binaire probleem. In het algemeen is de gevonden oplossing niet geheeltallig. De optimale oplossing van het continue probleem is nu niet meer een toegelaten oplossing voor het binaire probleem, maar het geeft wel een bovengrens voor de optimale doelfunctiewaarde voor het geheeltallige probleem. De extra geheeltalligheidseis legt een beperking op waardoor geen enkele geheeltallige oplossing een doelfunctiewaarde kan hebben die beter is dan de reeds gevonden oplossing. Van dit gegeven wordt handig gebruik gemaakt in de zogenaamde branch-and-boundmethode. Wanneer geen geheeltallige oplossing gevonden wordt, wordt het probleem gesplitst in twee subproblemen (branching). Er wordt een beslisvariabele gekozen die in de continue oplossing een niet-gehele waarde heeft. Vervolgens worden aan de hand van deze variabele twee kleinere problemen opgelost. Eén waarbij de beslisvariabele verplicht de waarde 1 krijgt en één waarbij de beslisvariabele de waarde 0 krijgt. Deze problemen worden subproblemen of knopen genoemd. Voor beide subproblemen wordt de procedure herhaald. Is er binnen een subprobleem nog geen geheeltallige oplossing gevonden, dan wordt er weer een variabele gekozen waarop de knoop wordt vertakt. Men gaat net zo lang door met vertakken tot er of een geheeltallige oplossing gevonden is of dat de gerelaxeerde oplossing van het beschouwde subprobleem een lagere doelfunctiewaarde heeft dan een eerder gevonden geheeltallige oplossing (bound). Wordt er een geheeltallige oplossing gevonden die beter is dan de tot dan toe beste oplossing, dan wordt deze oplossing vastgehouden als kandidaat voor de optimale oplossing. Is de optimale doelfunctiewaarde van het beschouwde subprobleem lager dan de kandidaatoplossing, dan heeft verder vertakken geen zin meer. De gevonden oplossing is immers een bovengrens voor de oplossing van alle subproblemen van het beschouwde probleem. Hiermee kan worden bewezen dat het subprobleem geen oplossingen kan geven die beter zijn dan de kandidaatoplossing. Ook kan het zijn dat de oplossingsruimte voor het subprobleem leeg is. Aangezien verdere subproblemen ook geen toegelaten oplossingen meer kunnen bevatten, heeft vertakken geen zin meer. De branch-and-boundmethode stopt als alle knopen beschouwd zijn. De gevonden kandidaat blijkt daadwerkelijk de optimale oplossing voor het oorspronkelijke probleem. De volgorde van vertakken is niet van wezenlijk belang voor de werking van de branch-and-boundmethode. In de praktijk wordt eerst de knoop waaraan men werkt verder vertakt, en pas als alle subproblemen van deze knoop zijn onderzocht wordt de tweede knoop onderzocht. De branch-and-boundmethode lijkt weliswaar omslachtig, maar als er een oplossing bestaat voor een probleem dan vindt de branch-and-bound altijd de optimale oplossing.

De branch-and-boundmethode zullen we toelichten aan de hand van het model voor het reizigersprobleem:

maximaliseer $100\,x_1 + 200\,x_2 + 100\,x_3$,

onder voorwaarde dat $20\,x_1 + 10\,x_2 + 20\,x_3 \leq 40$,
$20\,x_1 + 30\,x_2 + 40\,x_3 \leq 60$,
$x_1, x_2, x_3 \in \{0,1\}$. (geheeltalligheidseis)

De branch-and-boundmethode begint met de geheeltalligheidseis te vervangen door $0 \leq x_1, x_2, x_3 \leq 1$. Dit probleem duiden we aan met **0**. De simplex geeft voor **0** als optimum $x_1 = 1$, $x_2 = 1$, $x_3 = ¼$, met als opbrengst een bedrag van 325 kronen. Dit is geen geheeltallige oplossing en dus moet er worden gesplitst. In figuur 11.2 wordt in een zogenaamde beslisboom weergegeven hoe de problemen worden gesplitst en welke oplossing zij hebben.



Figuur 11.2
De beslisboom van de branch-and-bound procedure voor het reizigersprobleem

Eerst wordt subprobleem **1**, met als substitutie $x_3 = 1$, opgelost:

maximaliseer $100\,x_1 + 200\,x_2 + 100$,

onder voorwaarde dat $20\,x_1 + 10\,x_2 \leq 20,$
$20\,x_1 + 30\,x_2 \leq 20,$
$0 \leq x_1, x_2 \leq 1.$

Voor dit subprobleem wordt het optimum bereikt bij $x_1 = 0, x_2 = \frac{2}{3}, x_3 = 1$, met als opbrengst een bedrag van 233 kronen. Aangezien er weer geen geheeltallig optimum is gevonden, wordt er weer gesplitst. Let wel dat de subproblemen van **1** opgelost worden voordat er een nog openstaand probleem, namelijk probleem **4**, opgelost wordt. Het nieuwe subprobleem, probleem **2** genoemd, en ontstaan na substitutie van $x_2 = 1$, luidt:

maximaliseer $100\,x_1 + 300,$

onder voorwaarde dat $20\,x_1 \leq 10,$
$20\,x_1 \leq -10,$
$0 \leq x_1 \leq 1.$

Dit probleem heeft echter geen toegelaten oplossingen. Er wordt nu niet verder gegaan met splitsen, maar wordt het eerstvolgende nog openstaande probleem beschouwd. Dit is het subprobleem van **1**, probleem **3** genoemd, ontstaan na substitutie van $x_2 = 0$ en dit probleem luidt:

maximaliseer $100\,x_1 + 100,$

onder voorwaarde dat $20\,x_1 \leq 20,$
$20\,x_1 \leq 20,$
$0 \leq x_1 \leq 1..$

Nu wordt er wel een geheeltallig optimum bereikt bij $x_1 = 1, x_2 = 0, x_3 = 1$, met als opbrengst 200 kronen. Dit is de opbrengst die de reiziger krijgt als hij benzine en hondevoer meeneemt. We noemen deze oplossing nu de kandidaatoplossing, gaan niet verder met splitsen maar beschouwen het eerstvolgende nog openstaande probleem **4**. Merk op dat voor ieder volgend subprobleem de optimale doelfunctiewaarde is gedaald. Het nu nog openstaande probleem is het subprobleem van **0**, probleem **4**, ontstaan door substitutie van $x_3 = 0$, dat luidt:

maximaliseer $100\,x_1 + 200\,x_2,$

onder voorwaarde dat $20\,x_1 + 10\,x_2 \leq 40,$
$20\,x_1 + 30\,x_2 \leq 60,$
$0 \leq x_1, x_2 \leq 1.$

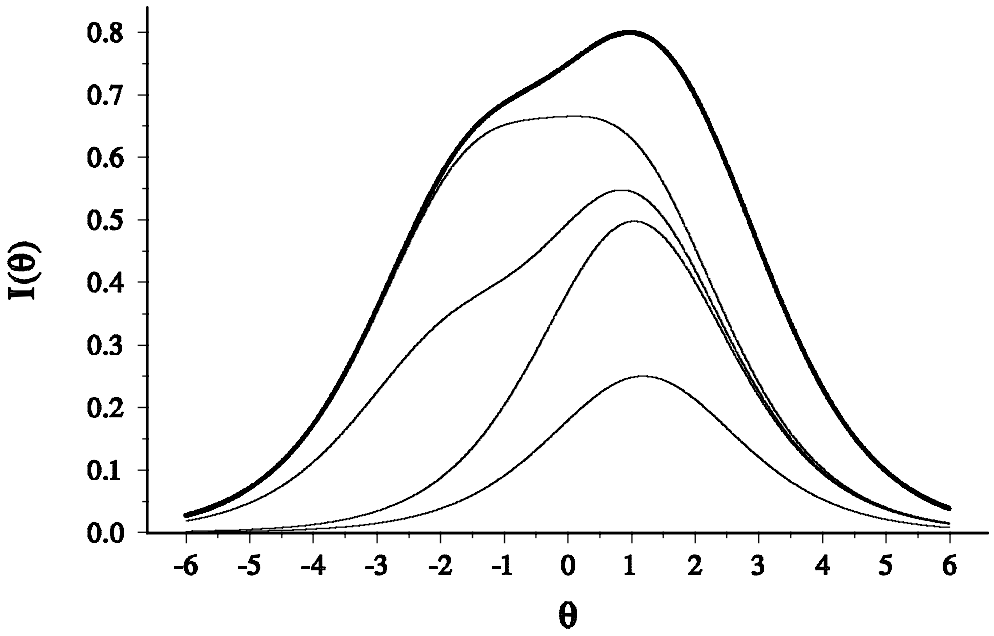
Hier wordt het optimum bereikt bij $x_1 = 1, x_2 = 1, x_3 = 0$, wat betekent dat de reiziger benzine en ananas moet meenemen, met als opbrengst een bedrag van 300 kronen. Er wordt dus weer een geheeltallig optimum gevonden. De opbrengst is nu echter hoger dan de opbrengst van de kandidaatoplossing, zodat de kandidaatoplossing wordt vervangen door de nu gevonden oplossing. Aangezien er geen openstaande subproblemen meer zijn is dit tevens de optimale oplossing voor het oorspronkelijke probleem.

11.2 Het samenstellen van toetsen in de itemresponstheorie

In de inleiding merkten we op dat psychometrische eisen betrekking hebben op de meetnauwkeurigheid van de toets. Binnen de itemresponstheorie worden voor het specificeren van de meetnauwkeurigheid continue functies gebruikt. De belangrijkste zijn de iteminformatie- en toetsinformatiefunctie. Zo is de standaarddeviatie van de grootste aannemelijkheidsschatter van de vaardigheid θ een functie van θ en gelijk aan $\mathrm{SE}(\hat{\theta}) = I(\theta)^{-½}$, waarbij $I(\theta)$ de toetsinformatiefunctie in het punt θ is (zie paragraaf 4.5.1). De informatie van een toets met lengte k is gelijk aan de som van de iteminformaties en wordt gegeven door

$$I(\theta) = \sum_{i=1}^{k} I_i(\theta).$$

Voor het Raschmodel is de iteminformatie gegeven door $I_i(\theta) = e^{(\theta-\beta_i)} \big/ \{1+e^{(\theta-\beta_i)}\}^2$, en deze functie is maximaal als $\theta = \beta_i$. Belangrijk voor toetsconstructie is het feit dat deze functies lokale meetnauwkeurigheid aangeven, dat wil zeggen dat de informatie afhankelijk is van het vaardigheidsniveau. Items die niet te moeilijk en niet te gemakkelijk zijn geven een hogere meetnauwkeurigheid dan zeer moeilijke en zeer gemakkelijke items. Figuur 11.3 laat zien hoe de toetsinformatie toeneemt wanneer we items aan een toets toevoegen. Telkens stijgt de toetsinformatiefunctie met de iteminformatiefunctie van het toegevoegde item.



Figuur 11.3 Toetsinformatiefunctie bij toenemende toetslengte

In Birnbaum (1968) en Lord (1980) vindt men een beschrijving van een trial-and-error heuristiek die van deze eigenschap gebruik maakt om toetsen met een bepaalde doelfunctie te construeren: de gewenste toetsinformatiefunctie, die afhankelijk is van het toetsdoel, wordt één voor één opgebouwd met de informatiefunctie van de gekozen items. Een belangrijke overweging is dat men vaak slechts in een beperkt gebied van de vaardigheidsschaal geïnteresseerd is, bijvoorbeeld in het cesuurgebied bij zakslaagbeslissingen. Men kan dan eisen stellen aan de meetnauwkeurigheid in het cesuurpunt en op twee punten daar net iets onder en boven. Men legt dan de toetsinformatiefunctie vast op een aantal punten maar blijft toch gebruik maken van het gegeven dat de toetsinformatie in ieder punt op de vaardigheidsschaal de som is van de iteminformaties. In het algemeen is het zo, dat continue functies voor bepaalde doeleinden goed gerepresenteerd kunnen worden door functies die uitsluitend zijn gedefinieerd op een aantal met zorg gekozen discrete punten.

Samenvattend: zeer belangrijk voor het probleem van het samenstellen van toetsen binnen de itemresponstheorie zijn de noties dat we voor een aantal punten op de vaardigheidsschaal de toetsinformatie specificeren en dat in elk punt de iteminformaties gesommeerd kunnen worden tot toetsinformatie. Op deze overwegingen is het gebruik van mathematische programmering bij toetsconstructie binnen itemresponstheorie gebaseerd. Al naar gelang de omstandigheden kan men eisen voor de toets met betrekking tot toetsinformatie formuleren als doel of als restrictie. Van beide zullen

later voorbeelden gegeven worden, zie ook Theunissen (1985, 1986), Van der Linden en Boekkooi-Timminga (1989).

11.2.1 Lineaire programmeringsproblemen

Voor de psychometrische en praktische eisen geldt dat ze als doelfunctie of als restrictie geformuleerd kunnen worden. Voor de inhoudelijke eisen geldt dat zij normaliter als restrictie geformuleerd worden. In de doelfunctie formuleert de toetsconstructeur wat er moet worden geoptimaliseerd, waarbij zowel minimaliserings- als maximaliseringsproblemen voor kunnen komen. Zoals we hebben gezien in paragraaf 11.1 zijn zowel doelfuncties als restricties te formuleren als eenvoudige lineaire expressies, waarbij men zich moet blijven realiseren dat de items in de expressies gerepresenteerd worden door binaire beslisvariabelen. Lineaire programmeringsmodellen worden algemeen geformuleerd als:

maximaliseer $$\sum_{i=1}^{K} c_i x_i,$$

onder voorwaarde dat $$\sum_{i=1}^{K} A_{ji} x_i \leq b_j, \quad j = 1,...,M$$

$$x_i \geq 0, \qquad i = 1,...,K.$$

Hierin zijn A_{ji}, b_j en c_i constanten, K het totaal aantal items in de itembank en M het aantal restricties.

We concentreren ons voorlopig op de doelfunctie. De variabelen x_i kunnen de waarden 1 en 0 aannemen. Ongeacht de betekenis van c_i is het duidelijk dat als $x_i = 0$, de daarbij behorende waarde van c_i niet zal bijdragen aan de waarde van de doelfunctie. De doelfunctie betreft een maximalisering: dat wil zeggen dat we proberen zoveel mogelijk van 'iets' te krijgen en dat 'iets' moet gunstig zijn in de ogen van de toetsconstructeur. Stel nu dat c_i de iteminformatie van item i is op een bepaald vaardigheidspunt, dan zegt bovenstaande doelfunctie niets anders als: 'maak een toets met een zo hoog mogelijke toetsinformatie (som van iteminformaties)'. Uiteraard dienen restricties toegevoegd te worden aan deze doelfunctie omdat anders alle beschikbare items in de toets zouden worden opgenomen. Stel nu dat de doelfunctie als volgt geformuleerd was:

minimaliseer $\sum_{i=1}^{K} c_i x_i$.

Ook hier nemen de x_i de waarden 1 en 0 aan, aangevend of item *i* al dan niet in de toets komt. Stel, dat de constructeur een bepaald doel voor ogen staat en we geven in deze doelfunctie aan alle c_i de waarde 1, dan houdt bovengenoemde doelfunctie niets meer in dan 'probeer aan alle (nog verderop te formuleren) voorwaarden te voldoen met een zo klein mogelijk aantal items', ofwel maak een toets van minimale omvang die nog aan eventuele andere voorwaarden beantwoordt. Een ander voorbeeld: stel dat -om herkenning te voor-komen- de toetsconstructeur vooral items in de toets op wil nemen die nog niet vaak gebruikt zijn en dat de gebruiksfrequentie voor alle items bekend is. We noemen de gebruiksfrequentie over een bepaalde periode voor item *i* hier dan c_i. Dus als item *i* bijvoorbeeld de afgelopen vier jaar twaalf maal gebruikt is, dan geldt $c_i = 12$. Omdat de doelfunctie als een minimalisering geformuleerd is, zullen items met een hoge bijbehorende waarde van c_i alleen in de toets worden opgenomen als er geen items in de bank beschikbaar zijn met een lagere waarde van c_i. Ook hier geldt uiteraard weer dat de gebruiksfrequentie van een item meetelt in de doelfunctie als de beslisvariabele x_i voor item *i* de waarde 1 aanneemt.

Behalve een doelfunctie zijn er ook randvoorwaarden in het probleem. Deze restricties zouden kunnen luiden:

$$\sum_{i=1}^{K} A_i x_i = b, \qquad \text{ofwel} \tag{11.6}$$

$$\sum_{i=1}^{K} A_i x_i \leq b, \qquad \text{ofwel} \tag{11.7}$$

$$\sum_{i=1}^{K} A_i x_i \geq b. \tag{11.8}$$

In de *b*'s in (11.6) - (11.8) kunnen de *b*'s van probleem tot probleem telkens iets anders betekenen en hoeven niet in dezelfde eenheden te zijn uitgedrukt. Hetzelfde geldt voor de A_i's. De flexibiliteit van deze eenvoudige modellen blijkt uit de zeer uiteenlopende interpretaties die men aan (11.6) - (11.8) kan toekennen. Zo kan men de eis dat de te maken toets van een specifieke lengte moet zijn formuleren als restrictie (11.6). Vaak is een vaste lengte de gewoonte, zoals bijvoorbeeld enkele meerkeuze examens van het voortgezet onderwijs die altijd vijftig items bevatten. Een restrictie als (11.6) wordt dan ingevuld door $A_i = 1$ te stellen voor alle items en uiteraard geldt $b = 50$. De restrictie zegt dan dat de te maken toets uit precies vijftig items moet bestaan, ongeacht doelfunctie of eventuele andere voorwaarden. Zou aan de eis dat er van alle items die

betrekking hebben op een bepaald hoofdstuk uit een leerboek precies twintig in de toets voorkomen moeten worden voldaan, dan geldt b = 20. Verder geldt dat de A_i's van alle items die horen bij dit hoofdstuk de waarde 1 krijgen, terwijl de A_i's voor de andere items de waarde 0 krijgen. Het is duidelijk, dat het geven van een waarde 1 of 0 aan de A_i's aanduidt of een item al dan niet 'meedoet' in de restrictie. Verderop zullen we zien dat aan de A_i's wel degelijk ook fractionele waarden toegekend kunnen worden of waarden groter dan 1.

Restricties als in (11.7) komen voor wanneer men in de toets bepaalde aspecten van die toets aan een grens wil verbinden die niet overschreden mag worden. Stel dat voor b de maximale afnametijd voor de gehele toets (zeg 50 minuten) wordt gekozen en voor A_i de benodigde tijd voor item i. Dan geeft restrictie (11.7) de eis weer dat de maximale toetsafnametijd vijftig minuten is. Het moge duidelijk zijn dat restricties als in (11.8) voorkomen als bepaalde zaken in een toets aan een ondergrens verbonden worden. Stel dat de toetsconstructeur eist dat op één bepaald vaardigheidspunt de toetsinformatie minimaal gelijk moet zijn aan 12.5. De waarde voor b wordt nu 12.5. Vervolgens berekent men voor alle items de iteminformatie voor dat specifieke vaardigheidspunt. Voor het Raschmodel zullen deze waarden liggen tussen 0 en het maximum 0.25, aannemend dat genormeerd is op een logistische schaal met gemiddelde 0 en discriminatieparameter gelijk aan 1. Dit zijn dan de waarden die aan de A_i's in restrictie (11.8) worden toegekend en in het optimaliserings-model worden opgenomen.

Het is niet mogelijk om een continue toetsinformatiefunctie te specificeren. Wel is het mogelijk om niet één vaardigheidspunt te definiëren maar meer. Zo worden de continue informatiefuncties gediscretiseerd. In alle zogenaamde discretisatiepunten worden de iteminformatiefuncties berekend en wordt een gewenste toetsinformatie opgegeven.

Hier ziet men trouwens hoe een zo belangrijke zaak als toetsinformatie in het optimale toetsconstructieproces kan verschijnen in ofwel de doelfunctie, ofwel in een restrictie. In het algemeen geldt dat dit voor verschillende aspecten van het toetsconstructieproces het geval kan zijn, zie het andere voorbeeld hierboven betreffende de toets van minimale lengte (doelfunctie) of vaste lengte (restrictie). Een combinatie van (11.7) en (11.8) zou kunnen zijn een voorwaarde waarin de onder- en bovengrenzen van aantallen items uit de onderscheiden leerstofcategorieën worden gespecificeerd:

$$A^{\ell} \leq \sum_{i=1}^{K} A_i x_i \leq A^{u}. \qquad (11.9)$$

Stel dat het aantal kennisvragen in de toets tussen een bepaald minimum en maximum moet liggen, zeg tussen vijftien en twintig. In dat geval wordt $A^{\ell} = 15$ en $A^{u} = 20$.

Definiëren we $A_i = 1$ voor alle kennisitems en $A_i = 0$ voor alle andere items, dan geeft (11.9) de eis weer dat er tussen vijftien en twintig kennisitems in de toets moeten worden opgenomen.

11.2.2 Praktijkvoorbeelden

Hoewel uit enkele combinaties van doelfuncties met restricties eenvoudige voorbeelden van toetsconstructie kunnen worden geformuleerd, zal er in de praktijk meestal sprake zijn van één doelfunctie en nagenoeg altijd van verschillende restricties. Het moge duidelijk zijn dat het probleem van het construeren van een toets van minimale lengte met een gespecificeerde ondergrens voor toetsinformatie op één discretisatiepunt zonder verdere restricties triviaal is vanuit zowel psychometrisch standpunt als optimaliseringsstandpunt. Daar in nagenoeg alle gevallen van toetsconstructie in het kader van itemresponstheorie gebruik wordt gemaakt van specificaties van toetsinformatie, zal eerst een aantal gevallen worden behandeld die in de toetspraktijk zullen voorkomen, waarbij we ons concentreren op deze toetsinformatie. Uitgewerkte voorbeelden worden om praktische redenen tot beperkte omvang gehouden. Bij de voorbeelden hierna zal voor de vaardigheidsschaal de logistische θ-schaal gebruikt worden in het praktische bereik van $\theta = -3$ tot $\theta = 3$.

Bij de specificatie van de toetsinformatie wordt de toetsconstructeur gedwongen goed voor ogen te houden wat het gebruiksdoel van de toets is. Daar er in de praktijk altijd met een eindig aantal items gewerkt wordt, is het mogelijk dat er geen enkele toets is te vinden die aan alle te bedenken gebruiksdoelen op gelijkwaardige wijze voldoet. Stel dat een toets-constructeur vooral geïnteresseerd is in zak-slaagbeslissingen. Een eis die aan de te maken toets gesteld moet worden is dat deze het meest nauwkeurig meet op het zak-slaagpunt, aangezien er voor kandidaten met een geschatte vaardigheid in dit gebied belangrijke beslissingsfouten gemaakt kunnen worden. Kandidaten met hoge of lage vaardigheid zullen door meetonnauwkeurigheid in het cesuurgebied niet benadeeld of bevoordeeld worden. Stel dat het cesuurpunt ligt op die vaardigheid, zodat vijftig procent van de groep studenten zakt en vijftig procent slaagt. De gewenste ondergrens voor de toetsinformatie in dit vaardigheids-punt wordt gesteld op 10. Voor het 25e en 75e percentiel wordt een toetsinformatie van 8 geëist. Dit heeft als gevolg dat het verloop rondom de piek van de toetsinformatie iets vlakker wordt. Het volgende schema kan dan gepresenteerd worden (zie tabel 11.1).

Tabel 11.1

Het eerste programmeringsprobleem

Specificatie θ-niveau bij percentiel	Iteminformatie voor item 1, item 2,..., item K	Ondergrens toetsinformatie bij θ-niveau
25	$I_1(\theta_1),...,I_K(\theta_1)$	$I(\theta) = 8$
50	$I_1(\theta_2),...,I_K(\theta_2)$	$I(\theta) = 10$
75	$I_1(\theta_3),...,I_K(\theta_3)$	$I(\theta) = 8$

Stel dat het de wens van de toetsconstructeur is deze specificatie met een zo gering mogelijk aantal items te bereiken, dan zal voor bovengenoemd voorbeeld de mathematische formulering van het optimaliseringsprobleem als volgt luiden:

minimaliseer $x_1 + x_2 + \cdots + x_K$

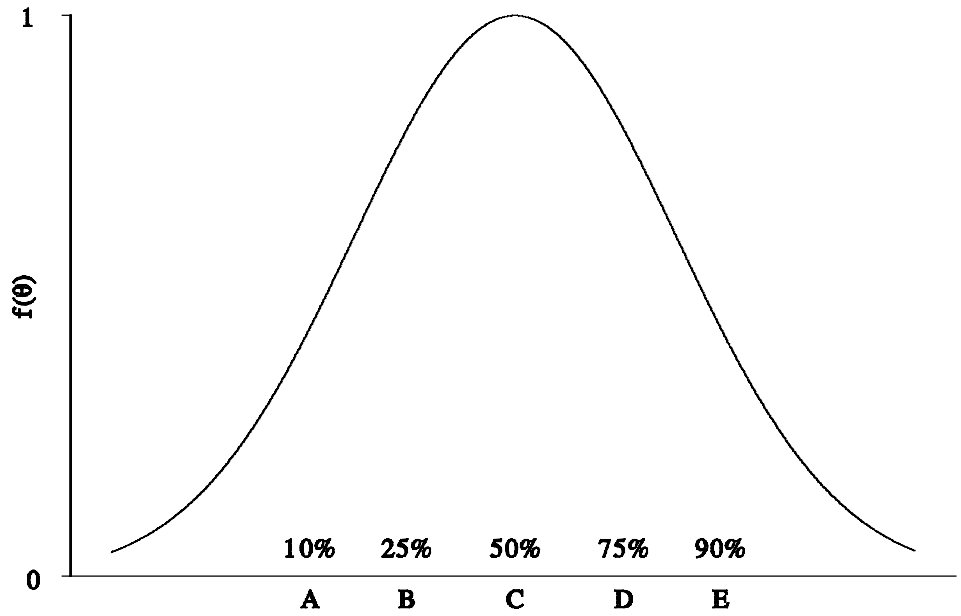
onder voorwaarde dat $I_1(\theta_1)x_1 + I_2(\theta_1)x_2 + \cdots + I_K(\theta_1)x_K \geq 8,$

$I_1(\theta_2)x_1 + I_2(\theta_2)x_2 + \cdots + I_K(\theta_2)x_K \geq 10,$

$I_1(\theta_3)x_1 + I_2(\theta_3)x_2 + \cdots + I_K(\theta_3)x_K \geq 8,$

$x_i = \varepsilon \{0,1\}, \quad i = 1,...,K.$

Uitgaande van een itembank van vijfhonderd gecalibreerde rekenitems kunnen we de praktijk van toetsconstructie verduidelijken. Als discretisatiepunten kiezen we hier de vaardigheden die overeenkomen met het 25e, 50e en 75e percentiel in de doelgroep. Alleen op deze discretisatiepunten worden de iteminformatie-functies, de te bereiken toetsinformatiefunctie en de bereikte toetsinformatie beschouwd. Deze vaardigheidsniveaus zijn in figuur 11.4 aangegeven met B, C en D.

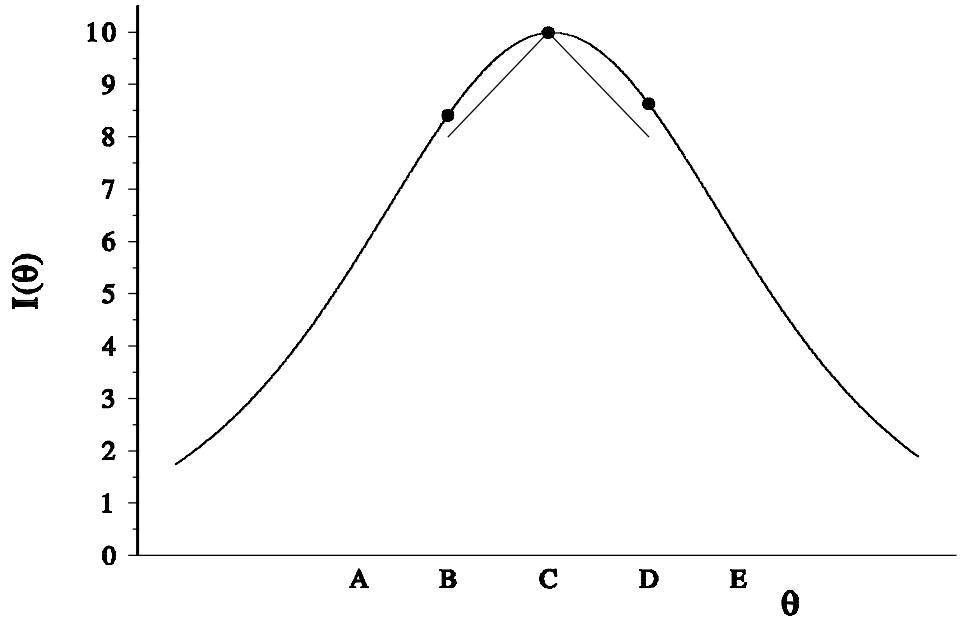


Figuur 11.4 Discretisatiepunten voor de toetsconstructie

Het voert te ver om in detail te laten zien hoe de branch-and-bound een oplossing vindt voor dit probleem. Het resultaat van de oplosmethode kunnen we echter wel laten zien. De geëiste toetsinformatie en de bereikte toetsinformatie met 40 items staan weergegeven in figuur 11.5

Punten A en E zijn in figuur 11.4 en figuur 11.5 toegevoegd om een vergelijking met het probleem van figuur 11.6 te vereenvoudigen.

Stel echter dat de constructeur een geheel ander doel voor ogen staat, namelijk een toets voor zeer algemeen gebruik voor het meten van vaardigheid en hij of zij vindt, dat - uit hoofde van sociale rechtvaardigheid - iedere leerling er recht op heeft met ongeveer dezelfde nauwkeurigheid gemeten te worden. Dit impliceert dat de gewenste toetsinformatie over het relevante gedeelte van de vaardigheidsschaal zoveel mogelijk uniform moet zijn.



Figuur 11.5 Geëiste en bereikte toetsinformatie voor het eerste probleem

Als de toetslengte niet onbeperkt toe mag nemen, impliceert dit tevens dat de gespecificeerde (uniforme) toetsinformatie beduidend lager moet zijn dan in het eerste voorbeeld. Stel de toetsspecificatie is het maken van een toets van minimale omvang en met toetsinformatie 6.0 op de θ-niveaus die behoren bij het 10e, 25e, 50e, 75e en 90e percentiel. Een schema van de formulering van dit probleem wordt weergegeven in tabel 11.2.

Tabel 11.2
Schema van het tweede probleem

Specificatie toetsinformatie θ-niveau bij percentiel	Iteminformatie voor item 1, item 2, ..., item K	Ondergrens bij θ-niveau
10	$I_1(\theta_1),...,I_K(\theta_1)$	$I(\theta_1) = 6$
25	$I_1(\theta_2),...,I_K(\theta_2)$	$I(\theta_2) = 6$
50	$I_1(\theta_3),...,I_K(\theta_3)$	$I(\theta_3) = 6$
75	$I_1(\theta_4),...,I_K(\theta_4)$	$I(\theta_4) = 6$
90	$I_1(\theta_5),...,I_K(\theta_5)$	$I(\theta_5) = 6$

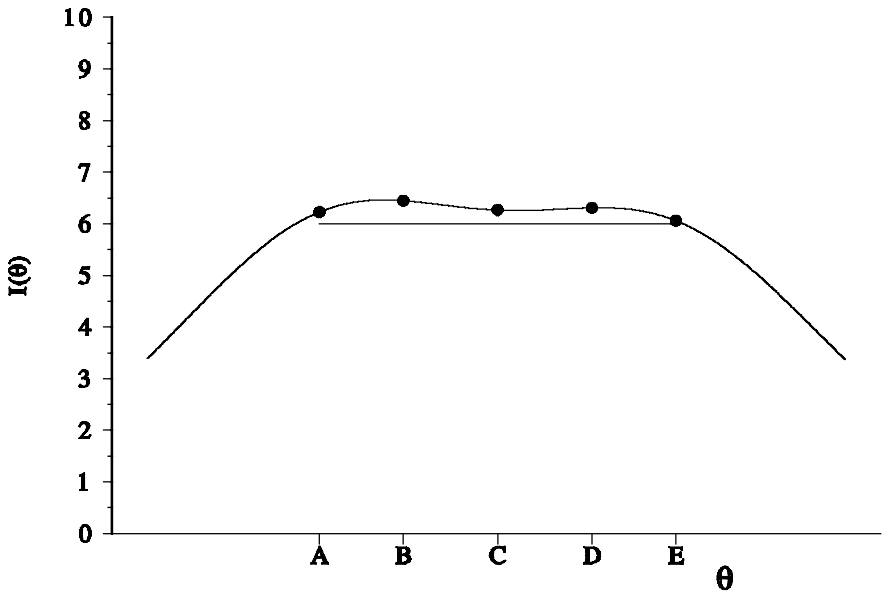
Als ook hier de toets uit een zo gering aantal mogelijk aantal items moet bestaan, dan zal de mathematische formulering luiden:

minimaliseer $\sum_{i=1}^{K} x_i$

onder voorwaarde dat $\sum_{i=1}^{K} I_i(\theta_m)\, x_i \geq I(\theta_m), \qquad m = 1,...,5$

$x_i \; \varepsilon \; \{0,1\}, \qquad i = 1,...,K.$

Figuur 11.6 laat de informatiefunctie van de nu geconstrueerde toets zien. Deze toets bestaat uit 40 items, net als de toets die geconstrueerd is voor het eerste probleem. Merk op dat om een meer gelijkmatige meetnauwkeurigheid te bereiken de toetsinformatie in het punt C lager is dan in het eerste voorbeeld.



Figuur 11.6
De toetsinformatie voor het tweede probleem

In voorgaande voorbeelden werd de gewenste toetsinformatie geformuleerd als een restrictie in het optimaliseringsprobleem. We geven nu een voorbeeld van toetsinformatie in de doelfunctie, waarbij een gewenste vorm van de toetsinformatiefunctie wordt gespecificeerd in plaats van de hoogte. Dit is nuttig als de toetsconstructeur slechts globaal kan aangeven hoe de verhouding van de toetsinformatie voor de verschillende vaardigheidsgebieden moet zijn. Deze situatie zal

bij voorbeeld ontstaan als de toetsconstructeur wel weet waarvoor de informatiefunctie dient, maar geen ervaring heeft in het omgaan met deze functie of met de getalsmatige aspecten ervan. De constructeur zou dan op de gewenste M specificatiepunten op de vaardigheidsschaal fiches kunnen plaatsen, zodanig dat de aantallen r_m (m = 1,...,M) de gewenste verhouding weerspiegelen. Vervolgens moeten de items zo gekozen worden dat de toetsinformatie gemaximaliseerd wordt met behoud van de vorm. Dit houdt in dat de toetsinformatie voor het θ_m-punt waarvoor de verhouding tussen toetsinformatie en r_m het laagst is, wordt gemaximaliseerd. Dit wordt in de volgende doelfunctie geformuleerd:

$$\text{maximaliseer} \left\{ \text{minimum} \quad \frac{I(\theta_m)}{r_m} \right\} = \left\{ \text{minimum} \quad \frac{\sum_{i=1}^{K} I_i(\theta_m)\, x_i}{r_m} \right\}.$$

Hierbij geldt $x_i \varepsilon \{0,1\}$. Daar de simplexmethode lineariteit van de doelfunctie vereist, dus geen 'knik' in het functieverloop of discontinuïteit toestaat, moet er een extra maatregel genomen worden. Dit is de introductie van een dummyvariabele y die de doelfunctie lineair maakt. Dummyvariabelen worden gebruikt om een probleem te kunnen formuleren maar spelen zelf geen rol in de oplossing van het eigenlijke probleem. Dit leidt dan tot het volgende optimaliseringsprobleem:

maximaliseer $\quad y$

onder voorwaarde dat $\quad y \leq \dfrac{\sum_{i=1}^{K} I_i(\theta_m)\, x_i}{r_m}, \qquad m = 1,...,M$

ofwel, na herschrijving, $\quad \sum_{i=1}^{K} I_i(\theta_m)\, x_i - r_m y \geq 0 \qquad m = 1,...,M.$

In deze restrictie worden ondergrenzen $r_m\, y$ aan de toetsinformatie geformuleerd voor elk van de θ_m-punten. De maximalisatie van y, en daarmee van de grootheden $r_m\, y$, 'duwt' de toetsinformatie omhoog. Zoals eerder vermeld leidt deze formulering tot opname van alle beschikbare items. Dus wordt de volgende restrictie toegevoegd:

$$\sum_{i=1}^{K} x_i = k,$$

waar k de gewenste lengte van de toets is. Voorts uiteraard weer $x_i \in \{0,1\}$ en y niet-negatief (waarom?). Deze modellen staan bekend onder de naam maximinmodellen, vanwege het feit dat het minimum over een aantal functies wordt gemaximaliseerd.

Ook hier geven we een voorbeeld uit de eerder genoemde itembank van vijfhonderd rekenitems. Naast calibratiegegevens zijn echter ook vakinhoudelijke gegevens beschikbaar: ieder item is gecategoriseerd als een optelling, een aftrekking, een vermenigvuldiging of een deling. Deze categorieën zijn hieronder vermeld als categorie 10, 11, 12 en 13. Stel dat de toetsconstructeur een toets wil samenstellen van veertig items, met tien optellingen, tien aftrekkingen, tien vermenigvuldigingen en tien delingen. Deze eis kan worden geformuleerd zoals in (11.9). Voor m kiezen we 10, 11, 12 en 13. Verder definiëren we $A_{10,i} = 1$ voor alle optellingen, en $A_{10,i} = 0$ voor de andere items. De andere A_{mi}'s worden op dezelfde wijze gedefinieerd. Nu geldt:

$$\sum_{i=1}^{K} A_{mi} x_i = 10, \qquad m = 10,...,13.$$

Daarnaast moet de toets nauwkeuriger meten in het vaardigheidsgebied van de iets zwakkere leerlingen: de toetsinformatie voor het 10e tot het 50e percentiel moet twee keer zo hoog zijn als de toetsinformatie voor het 90e percentiel. Dit komt tot uitdrukking in figuur 11.7. Hier geldt $r_1 = 10$, $r_2 = 10$, $r_3 = 5$.

Het gehele model wordt geformuleerd als:

maximaliseer y

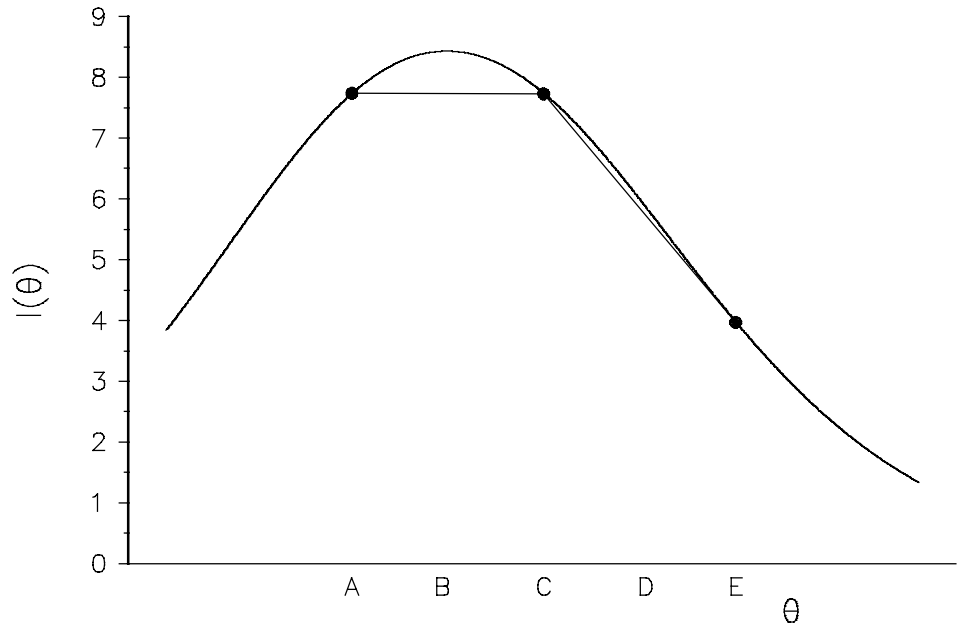
onder voorwaarde dat
$$\sum_{i=1}^{K} I_i(\theta_m) x_i - r_m y \geq 0 \qquad m = 1,...,M$$

$$\sum_{i=1}^{K} x_i = 40$$

$$\sum_{i=1}^{K} A_{mi} x_i = 10 \qquad m = 10,...,13$$

$$x_i \in \{0,1\} \qquad i = 1,...,K.$$

Merk op dat in figuur 11.7 de variabele y de waarde 0.77 heeft in de optimale oplossing. In figuur 11.7 zijn zowel de bereikte toetsinformatie gegeven als de grootheden $r_m y$.



Figuur 11.7
De toetsinformatie behorend bij het derde probleem

In de praktijk ontstaan vaak situaties waarbinnen behoefte is aan toetsen die dezelfde lokale meetnauwkeurigheid hebben. In het kader van de itemresponstheorie worden toetsen zwak parallel genoemd als ze identieke toetsinformatiefuncties hebben (Samejima, 1977). Behoefte aan parallelle toetsen ontstaat in die situaties waarin het gewenst is dat toetsen uitwisselbaar zijn, bijvoorbeeld bij kort opeenvolgende herhaalde metingen van dezelfde personen. Parallelle toetsconstructie kan zowel sequentieel als simultaan plaatsvinden. Voor een uit-gebreid overzicht zie Boekkooi-Timminga (1990). Bij sequentiële constructie is er sprake van een opeenvolging van toetsconstructies, waarbij men steeds rekening moet houden met hetgeen voorafging. Bij simultane constructie probeert men gelijktijdig een verzameling items te verdelen over een aantal toetsen. Voor een itembank die goed gevuld is met items die relevant zijn voor het toetsconstructieprobleem dat aan de orde is, blijkt het in de praktijk vaak al voldoende om eerst één toets volgens de specificatie te laten maken. Vervolgens geeft men bij de aanmaak van de tweede toets die items die reeds in de eerste toets zijn opgenomen een gewicht van 2 in plaats van 1 in de doelfunctie. Hierdoor is het vrijwel uitgesloten dat

deze items in een volgende toets worden opgenomen. Dit geldt uiteraard alleen als de doel-functie de minimalisering van het aantal items betreft.

Gebruikt men het eerder beschreven maximinmodel dan kan door het toevoegen van de drie volgende restricties een tweede parallelle toets geconstrueerd worden:

$$\sum_{i=1}^{K} I_i(\theta_m)\, x_{it} \geq (1 - p)\, I(\theta_m)$$

$$\sum_{i=1}^{K} I_i\,(\theta_m)\, x_{it} \leq (1 + p) I(\theta_m)$$

$$\sum_{t=1}^{T} x_{it} \leq 1 \qquad i = 1,...,K.$$

De eerste twee restricties geven een ondergrens en een bovengrens voor afwijking tussen de gewenste toetsinformatie en de bereikte informatie. De derde restrictie stipuleert dat geen enkel item in meer dan een toets aanwezig mag zijn. In het model is $I(\theta_m)$ de reeds bestaande toetsinformatie op punt θ_m, en p is het maximaal toegestane verschil in toetsinformatie tussen de reeds bestaande en nog te realiseren toets. Ook hier geldt dat het raadzaam is goed gevulde banken te gebruiken.

11.2.3 Specificeren van restricties en relaties

In het voorgaande lag de nadruk op specificaties van aantallen items en toetsinformatie in zowel doelfuncties als restricties. Vele andere specificaties kunnen eveneens gemodelleerd worden als restrictie of doelfunctie (Van der Linden & Boekkooi-Timminga, 1989). Zo werd reeds gewezen op de mogelijkheid de verdeling van items over inhoudelijke categorieën te modelleren. Hetzelfde geldt voor de afnametijd van de toets, door ofwel een bovengrens als restrictie op te nemen, ofwel door opname in de doelfunctie, ofwel door het zogenaamde multi-stage programming, waarin specificaties afwisselend in doelfunctie en restrictie terecht komen. Voorts blijkt het mogelijk om te werken met clusters van items, een situatie die zich voordoet bij tekstbegriptoetsen (Theunissen, 1987). Hier horen bij één tekst in de regel verschillende items en kunnen de teksten alleen met alle bijbehorende items tegelijk geselecteerd worden.

Nu zullen we zien hoe het constructieproces desnoods zeer gedetailleerd op verschillende niveaus en zeer specifiek gestuurd kan worden, zodat bijvoorbeeld ook aan detailwensen gehoor kan worden gegeven. We sluiten daarbij weer aan bij het volgende basismodel:

minimaliseer $\sum_{i=1}^{K} c_i x_i$

onder voorwaarde dat $\sum_{i=1}^{K} A_i x_i \geq b,$

$x_i \in \{0,1\}.$

Eén mogelijkheid behelst het introduceren van dummyvariabelen. Dit kan bijvoorbeeld nodig zijn voor bijsturing van het toetsconstructieproces op het niveau van de restricties. Stel we hebben in de specificatie van het constructieproces opgenomen de restrictie:

$$\sum_{i=1}^{K} A_i x_i \leq b. \tag{11.10}$$

Laten we nu aannemen dat deze restrictie niet altijd van kracht hoeft te zijn, maar pas geldt als een bepaald item, of een bepaalde groep van items, in de toets wordt opgenomen. Bovenstaande restrictie kan bijvoorbeeld betrekking hebben op de gemiddelde tijd die nodig is voor het maken van een item, waarbij de coëfficiënten A_i voor de antwoordtijd per item staan en b de maximale toetstijd is. De restrictie wordt geacht mee te gaan spelen bij opname van items met een lange antwoordtijd. Als dit gebeurt wordt een dummyvariabele δ gelijk gesteld aan 1 en vervolgens geldt

$$\delta = 1 \rightarrow \sum_{i=1}^{K} A_i x_i \leq b, \tag{11.11}$$

waar $\rightarrow$ betekent 'impliceert'. We stellen nu het getal G als een bovengrens voor de uitdrukking $\sum A_i x_i - b$. Als $\delta = 1$ (ofwel, als $1 - \delta = 0$), wensen we dat $\sum A_i x_i - b \leq 0$, hetgeen volgt uit (11.11). Als G voldoende groot wordt gekozen, zal dit het geval zijn als $\sum A_i x_i - b \leq G(1 - \delta)$. Na enige herordening krijgen we dan uit de conditie (11.11) de volgende restrictie:

$$\sum_{i=1}^{K} A_i x_i + G\delta \leq G + b. \tag{11.12}$$

Uit (11.12) volgt, dat als $\delta = 0$ er geen sprake is van een restrictie, terwijl bij $\delta = 1$ de restrictie (11.10) van kracht is. Het verband tussen het 'optreden' van item *i* en de dummyvariabele δ wordt gelegd door de volgende restrictie te introduceren: $x_j - \delta \leq 0$. Dit houdt in dat δ de waarde 1 aanneemt als x_i groter is dan 0, dat wil zeggen, gelijk is aan 1.

Na formuleringen besproken te hebben die betrekking hebben op het niveau van de restricties van het toetsconstructieprobleem, zijn we nu aangekomen op het punt waar formuleringen worden gebruikt op het niveau van de items en hun onderlinge relaties. De variabelen zijn hier weer de beslisvariabelen x_i, die aangeven of de desbetreffende items gekozen worden. Uitspraken over een item of over de relaties tussen items worden geformuleerd via de volgende verzameling van operatoren:

- $\vee$ betekent of x of y of allebei,
- $\wedge$ betekent x en y tegelijk,
- $\neg$ betekent niet x,
- $\rightarrow$ betekent als...dan (implicatie),
- $\leftrightarrow$ betekent dan en slechts dan.

We kunnen bovenstaande operatoren met enige eenvoudige voorbeelden demonstreren. We stellen ons voor dat uit een itembank toetsen samengesteld moeten worden waarbij steeds de items 1 en 2 een rol spelen. Door verschillen in de toetsspecificatie kunnen onder andere de volgende verschillende eisen aan items 1 en 2 gesteld worden.
De eis, dat ofwel item 1 ofwel item 2 ofwel beide moet worden opgenomen, wordt geformuleerd als $x_1 \vee x_2$ en in de vorm van restrictie in het optimaliseringsprobleem als $x_1 + x_2 \geq 1$. De eis, dat zowel item 1 als item 2 moeten worden opgenomen, wordt geformuleerd als $x_1 \wedge x_2$ en in de vorm van restrictie als $x_1 + x_2 \geq 2$. De eis dat item 1 niet op-genomen mag worden wordt uitgedrukt als $\neg x_1$ en in de vorm van restrictie als $x_1 = 0$. De eis dat als item 1 wordt opgenomen ook item 2 moet worden opgenomen, wordt $x_1 \rightarrow x_2$ en in de vorm van restrictie $x_1 - x_2 \leq 0$. De eis dat item 1 en item 2 alleen tezamen mogen worden opgenomen, wordt geformuleerd als $x_1 \leftrightarrow x_2$ en in de vorm van restrictie als $x_1 - x_2 = 0$. Het verschil tussen beide laatste formuleringen ligt in het feit dat in het laatste geval item 2 alleen kan optreden samen met item 1, terwijl in het voorlaatste geval item 2 ook los van item 1 kan optreden, vandaar in het voorlaatste geval het '≤' teken. Vanuit deze elementaire uitdrukkingen kunnen verdere expressies geformuleerd worden van iedere noodzakelijke graad van complexiteit.

Tot besluit een voorbeeld: stel we formuleren als eis dat, als item 1 of item 2 of beide worden opgenomen, dan minstens één van de items 3, 4 of 5 moet worden opgenomen. Dit wordt geformuleerd als:

$$(x_1 \vee x_2) \rightarrow (x_3 \vee x_4 \vee x_5). \qquad (11.13)$$

Het linker lid van (11.13) wordt als restrictie $x_1 + x_2 \geq 1$, en het rechter lid $x_3 + x_4 + x_5 \geq 1$. Vervolgens introduceren we een nieuwe indicatorvariabele δ en stellen dat moet gelden $x_1 + x_2 \geq 1 \rightarrow \delta = 1$, en tevens dat $\delta = 1 \rightarrow x_3 + x_4 + x_5 \geq 1$.
Eis (11.13) wordt dan geformuleerd als de volgende twee restricties: $x_1 + x_2 - 2\delta \leq 0$ en $-x_3 - x_4 - x_5 + \delta \leq 0$. Met gebruik van dit soort formuleringen kan het proces van samen-stellen van toetsen minutieus gestuurd worden. Er kan echter ook een nadeel aan kleven. Als er teveel restricties toegevoegd worden aan het optimaliseringsprobleem, kan er een situatie ontstaan waarbij de algoritmen die gebruikt worden om de oplossing te vinden minder effectief worden.

Binnen het korte bestek van deze paragraaf kon niet alles wat er te zeggen valt over de optimale samenstelling van toetsen binnen de itemresponstheorie aan de orde komen. Zo werd niet ingegaan op de mogelijkheid om verscheidene doelfuncties te samen te optimaliseren, het zogenaamde 'multi-objective' programmeren. Ook is grotendeels onbesproken gelaten de ont-wikkeling van heuristische methoden die gebruikt kunnen worden als exacte algoritmen voor de oplossing van optimaliseringsproblemen teveel computertijd zouden vergen. Ook is weinig aandacht besteed aan de beschikbaarheid van computerprogrammatuur voor de optimale samenstelling van toetsen. Voor dit laatste verwijzen we naar de handleiding van het computerprogramma Optimal Test Design (Verschoor, 1991).

11.3 Het samenstellen van toetsen in de klassieke testtheorie

In zijn boek over klassieke testtheorie opent Gulliksen (1950) het hoofdstuk over itemselectie als volgt: 'Basically, item analysis is concerned with the problem of selecting items for a test, so that the resulting test will have certain specified characteristics' (p. 363). In hoofdstuk 3 zagen we dat in de klassieke testtheorie de betrouwbaarheid een belangrijk kenmerk van een toets is. Gulliksen beschrijft een grafische procedure voor de selectie van items die de betrouwbaarheid van de toets maximaliseert wanneer de samen te stellen toets uit een vooraf bepaald aantal items bestaat. Welke items de betrouwbaarheid meer doen toenemen dan andere items, kan toegelicht worden aan de hand van Cronbachs coëfficiënt alpha, die gedefinieerd is als

$$\alpha = k\,(k-1)^{-1}\left[1 - \left(\sum_{i=1}^{k}\sigma_i^2\right) \Big/ \left(\sum_{i=1}^{k}\sigma_i\,\rho_{it}\right)^2\right], \tag{11.14}$$

waarbij k het aantal items in de toets, σ_i^2 de variantie van item i, en ρ_{it} de correlatie tussen de score op item i en de score op de toets is. Uit formule (11.14) kan afgeleid worden dat wanneer het aantal items in de toets gefixeerd is, coëfficiënt alpha gemaximaliseerd wordt door het minimaliseren van de ratio

$$\left(\sum_{i=1}^{k}\sigma_i^2\right) \Big/ \left(\sum_{i=1}^{k}\sigma_i\,\rho_{it}\right)^2. \tag{11.15}$$

De ratio (11.15) laat zien dat minimalisatie kan worden bereikt door verkleining van de teller, de som van de varianties van de items, of door vergroting van de noemer, de gekwadrateerde som van de betrouwbaarheidsindices van de items. Merk op dat de variantie van de items zowel in de teller als in de noemer van de ratio voorkomt. In hoofdstuk 3 zagen we dat aanzienlijke verschillen in moeilijkheidsgraad slechts aanleiding geven tot kleine verschillen in itemvarianties. Het onderzoek van Ebel (1967) laat dan ook zien dat de betrouwbaarheid minder afhangt van de teller dan van de noemer van (11.15). Dit betekent dat voor het maximaliseren van de betrouwbaarheid met name items met een hoge item-testcorrelatie geselecteerd moeten worden. Het laatste gegeven betekent dat we de niet-lineaire doelfunctie (11.15) kunnen vervangen door een lineaire doelfunctie. Het oplossen van problemen met lineaire doelfuncties veel eenvoudiger is dan het oplossen van problemen met niet-lineaire doelfuncties. Adema en Van der Linden (1989) formuleerden het volgende lineaire programmeringsmodel voor het samenstellen van toetsen:

maximaliseer
$$\sum_{i=1}^{K}\rho_{it}\,x_i \tag{11.16}$$

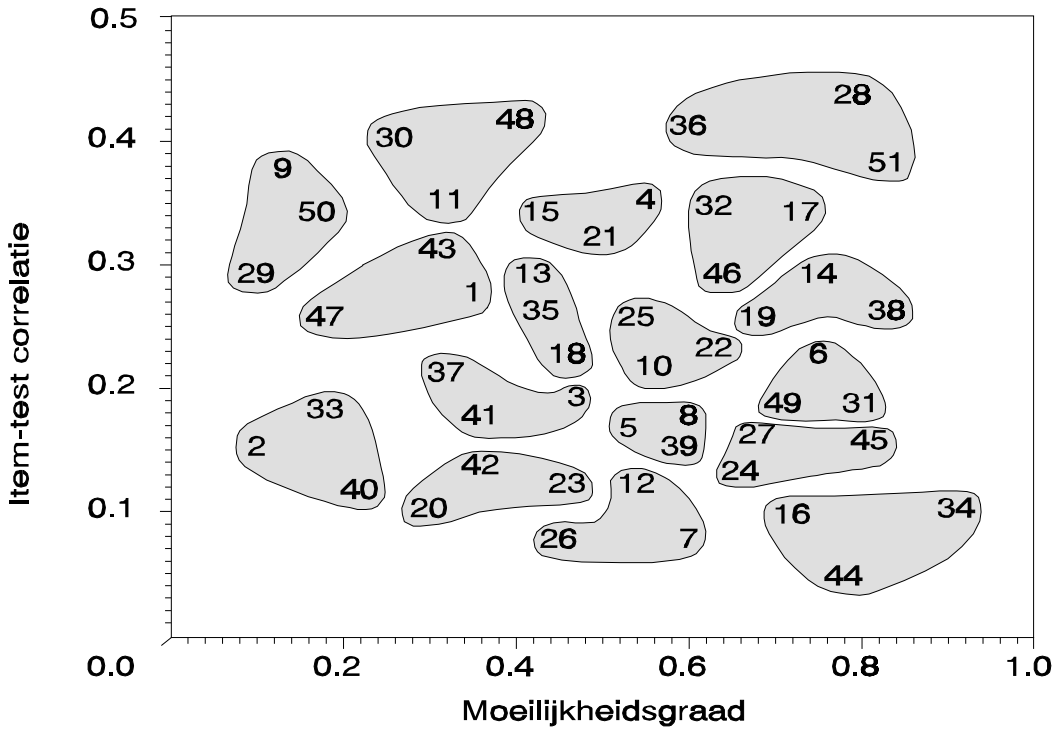
onder voorwaarde dat
$$\sum_{i=1}^{K}x_i = k, \tag{11.17}$$

$$\sum_{i=1}^{K}t_i\,x_i \le 35\,k, \tag{11.18}$$

$$x_i \in \{0,1\}, \quad i = 1,\ldots,K. \tag{11.19}$$

In het bovenstaande model wordt de betrouwbaarheid gemaximaliseerd door middel van een doelfunctie (11.16) die de voorkeur verwoordt voor items met hoge item-testcorrelaties. In dit model worden verder nog twee voorwaarden geformuleerd. Dat de toets uit *k* items moet bestaan wordt in voorwaarde (11.17) geformuleerd. De opname van deze voorwaarde in het model is noodzakelijk om de lengte van de toets te beperken omdat elk item met een positieve item-testcorrelatie de betrouwbaarheid van de toets verhoogt. In voorwaarde (11.18) staat dat er t_i seconden nodig zijn voor de beantwoording van item *i*. In de voorwaarden wordt echter ook gesteld dat de totale toets binnen 35 *k* seconden afgenomen moet zijn, wat de selectie van items met een relatief korte antwoordtijd impliceert.

Voor de samenstelling van parallelle toetsen ontwikkelde Gulliksen de 'matched random subtests method' (1950, p. 207 ev.). Hierbij wordt elk item afgebeeld als een punt in een grafiek met als abscis de moeilijkheidsgraad en als ordinaat de item-testcorrelatie. Op basis van deze itemparameters worden de items dan eerst simultaan gekoppeld en daarna wordt ieder item van elk gekoppeld paar of drietal random toegewezen aan een toets.



Figuur 11.8

De constructie van drie parallelle tests door simultane koppeling van item op basis van moeilijkheidsgraad en item-testcorrelatie

Figuur 11.8 laat voor 51 items het resultaat zien van de eerste stap van deze twee-staps procedure, namelijk 17 gekoppelde drietallen. De tweede stap is dat item 2 aan bijvoorbeeld de eerste toets, item 33 aan de tweede toets, item 40 aan de derde toets, item 20 aan de tweede toets, enz. wordt toegewezen. Het resultaat van de procedure is drie parallelle toetsen die elk uit 17 items bestaan.

Van der Linden en Boekkooi-Timminga (1988) ontwikkelden een binair programmerings-model voor de 'matched random subtests method' van Gulliksen. Voor de constructie van twee parallelle toetsen luidt het model:

$$\text{minimaliseer} \quad \sum_{i=1}^{K-1} \sum_{j=i+1}^{K} [(\pi_i - \pi_j)^2 + (\rho_{it} - \rho_{jt})^2]^{½} \; x_{ij} \tag{11.20}$$

$$\text{onder voorwaarde dat} \quad \sum_{i=1}^{j-1} x_{ij} + \sum_{i=j+1}^{K} x_{ji} = 1 \quad (j = 1,...,K) \tag{11.21}$$

$$x_{ij} \varepsilon \{0,1\} \qquad (i = 1,...,K-1; j = i+1,...,K). \tag{11.22}$$

De eerste stap van Gulliksen's grafische methode vervangen Van der Linden en Boekkooi-Timminga door doelfunctie (11.20), minimalisatie van de som van de binnen-paar Euclidische afstanden, en door de voorwaarden in (11.21) die garanderen dat elk item tot niet meer dan één paar items behoort. De binaire beslisvariabele x_{ij} geeft aan of i en j wel een paar zijn $(x_{ij} = 1)$ of geen paar zijn $(x_{ij} = 0)$. De eerste stap resulteert in $K/2$ paren items. Ook voor de tweede stap, het random toewijzen van items aan toetsen, formuleren zij binaire programmeringsmodellen met als doelfuncties gelijke gemiddelden en/of varianties.

Van der Linden en Boekkooi-Timminga geven de voorkeur aan een model voor parallelle toetsconstructie waarbij de items eerst in paren, drietallen enzovoort gekoppeld worden en niet direct aan toetsen toegewezen worden. Dit laatste model zou tot minder wenselijke toetsen kunnen leiden omdat de ene toets dan bijvoorbeeld uit items met nagenoeg dezelfde moeilijkheidsgraad bestaat terwijl de andere toets uit items van nogal verschillende moeilijkheidsgraad bestaat. Toetsen met dezelfde itemparameters - en daardoor ook dezelfde toetsparameters - voor corresponderende items, worden sterk-parallelle toetsen genoemd, terwijl toetsen met alleen dezelfde toetsparameters, zwak-parallelle toetsen genoemd worden. Het probleem van het

construeren van een toets die identiek is aan een reeds bestaande toets, hier aangeduid met referentietoets, is een variant van het probleem van het samenstellen van sterk-parallelle toetsen. Een oplossing voor dit probleem met behulp van technieken uit de mathematische programmering wordt beschreven in Armstrong, Jones en Wu (1992). Hun oplossing bestaat hieruit dat eerst getracht wordt de items in de itembank zo goed mogelijk te koppelen aan de items uit de referentietoets. Daarna worden parallelle toetsen samengesteld die zo weinig mogelijk afwijken van de referentietoets.

Het samenstellen van parallelle toetsen vormt ook het uitgangspunt van de twee modellen beschreven door Verschoor en Sanders (1993). Het samenstellen van een enkele toets wordt opgevat als een speciaal geval van parallelle toetsconstructie, namelijk een toets die parallel is met zichzelf. Het doel van model 1 van Verschoor en Sanders is om onder bepaalde voorwaarden het aantal items van de samen te stellen parallelle toetsen te minimaliseren. Het doel van model 2 is om onder bepaalde voorwaarden de betrouwbaarheid van parallelle toetsen te maximaliseren. De twee modellen gaan uit van klassieke itemparameters, dat wil zeggen van een verzameling items die met het klassieke testmodel gecalibreerd zijn of waarvan de klassieke itemparameters afgeleid zijn van de itemparameters van een item-responstheorie model. Deze laatste mogelijkheid kan nuttig zijn voor personen die onvoldoende bekend zijn met itemresponstheorie maar toch gebruik willen maken van nieuwe technieken voor het samenstellen van toetsen.

Model 1 beoogt om met zo weinig mogelijk items parallelle toetsen samen te stellen die een gespecificeerde betrouwbaarheid, gemiddelde toetsscore en standaarddeviatie hebben. De formulering van model 1 luidt:

minimaliseer
$$\sum_{i=1}^{K} x_{i1} \tag{11.23}$$

onder voorwaarde dat
$$\sum_{i=1}^{K} x_{i1} = \sum_{i=1}^{K} x_{it}, \qquad t = 2,...,T \tag{11.24}$$

$$\alpha^{\ell} \leq \alpha_t \leq \alpha^{u}, \qquad t = 1,...,T \tag{11.25}$$

$$\pi^{\ell} \sum_{i=1}^{K} x_{it} \leq \sum_{i=1}^{K} \pi_i \, x_{it} \leq \pi^{u} \sum_{i=1}^{K} x_{it}, \qquad t = 1,...,T \tag{11.26}$$

$$\sigma_x^{l} \leq \sigma_x \leq \sigma_x^{u} \tag{11.27}$$

$$\sum_{t=1}^{T} x_{it} \leq 1 \qquad i = 1,...,K. \tag{11.28}$$

De doelfunctie (11.23) beoogt het minimaliseren van het aantal items van de parallelle toetsen. In voorwaarde (11.24) staat dat voor alle T toetsen geldt dat ze uit evenveel items als de eerste toets dienen te bestaan. Voor elk item i is beslisvariabele x_{it} gedefinieerd als 1 indien item i in toets t is opgenomen en als 0 indien item i niet in toets t is opgenomen. In voorwaarde (11.25) worden de ondergrens, α^{ℓ}, en de bovengrens, α^{u}, van coëfficiënt alpha gespecificeerd. In voorwaarde (11.26) worden een ondergrens en een bovengrens van de moeilijkheidsgraad van de toetsen gespecificeerd. In voorwaarde (11.27) worden de onder- en bovengrens van de standaarddeviatie van de toetsen gespecificeerd. In voorwaarde (11.28) staat dat de toetsen niet dezelfde items mogen bevatten.

Het model 2 van Verschoor en Sanders beoogt parallelle toetsen samen te stellen met een zo hoog mogelijke betrouwbaarheid gegeven een bepaald aantal items de gemiddelde toetsscore en de standaarddeviatie. De formulering van model 2 luidt:

maximaliseer $\quad \text{minimum} \quad \alpha_t \qquad (11.29)$

onder voorwaarde dat $\quad \sum_{i=1}^{K} x_{it} = k, \qquad t = 1,...,T \qquad (11.30)$

$$\pi^{\ell} \leq \sum_{i=1}^{K} \pi_i \, x_{it} \leq \pi^{u}, \quad t = 1,...,T \qquad (11.31)$$

$$\sigma_x^{l} \leq \sigma_x \leq \sigma_x^{u} \qquad (11.32)$$

$$\sum_{t=1}^{T} x_{it} \leq 1 \qquad i = 1,...,K. \qquad (11.33)$$

Het maximaliseren van de betrouwbaarheden van parallelle toetsen staat in de doelfunctie (11.29). Dit doel wordt gerealiseerd door een maximinmodel, dat de betrouwbaarheid van de toets met de laagste betrouwbaarheid maximaliseert. In voorwaarde (11.30) wordt gespecificeerd dat de toetsen uit een vooraf bepaald gelijk aantal items dienen te bestaan. De betekenis van de voorwaarden (11.32) en (11.33) is gelijk aan die van de voorwaarden (11.27) en (11.28). Uiteraard is het ook bij de modellen mogelijk nog andere voorwaarden, bijvoorbeeld de verdeling van items over leerstofcategorieën, te specificeren.

Model 2 illustreren we hier voor het samenstellen van twee parallelle toetsen aan de hand van de reeds eerder gebruikte itembank met vijfhonderd rekenitems. Onze wensen specificeren we met het volgende model:

maximaliseer $\quad \text{minimum} \{ \alpha_1, \alpha_2 \}$

onder voorwaarde dat $\sum_{i=1}^{K} x_{i1} = \sum_{i=1}^{K} x_{i2} = 20$

$$10.0 \leq \sum_{i=1}^{K} \pi_i \, x_{it} \leq 11.0, \qquad t = 1, 2$$

$$\sum_{i=1}^{K} A_{mi} \, x_{it} = 5, \qquad t = 1, 2; \; m = 10, \ldots, 13$$

$$\sum_{t=1}^{2} x_{it} \leq 1, \qquad i = 1, \ldots, K.$$

In de doelfunctie van het model staat dat de betrouwbaarheden van de twee toetsen zo hoog mogelijk moeten worden. In de eerste voorwaarde wordt de eis geformuleerd dat de twee toetsen uit precies twintig items moeten bestaan. De tweede voorwaarde geeft de grenzen voor de moeilijkheidsgraad van de toetsen aan. In dit geval wordt gespecificeerd dat de gemiddelde toetsscore tussen de 10 en 11 scorepunten moet komen te liggen. Dat de twee toetsen vijf items uit elke leerstofcategorie dienen te bevatten, staat in de derde voorwaarde. In de vierde voorwaarde wordt geëist dat de twee toetsen niet dezelfde items mogen bevatten. De resultaten staan in tabel 11.3.

Tabel 11.3

Constructie van twee parallelle toetsen met model 2

Toets 1				Toets 2			
Item	p	r_{it}	Cat.	Item	p	r_{it}	Cat.
11	0.50	0.406	10	3	0.40	0.368	10
71	0.67	0.375	10	94	0.69	0.349	10
460	0.46	0.341	10	214	0.14	0.340	10
466	0.58	0.380	10	345	0.83	0.365	10
485	0.50	0.470	10	389	0.26	0.348	10
90	0.69	0.378	11	33	0.51	0.364	11
249	0.49	0.358	11	62	0.58	0.369	11
293	0.82	0.343	11	203	0.75	0.337	11
426	0.74	0.360	11	299	0.56	0.361	11
433	0.67	0.402	11	455	0.45	0.443	11
119	0.40	0.379	12	7	0.36	0.477	12
360	0.19	0.406	12	148	0.47	0.306	12
378	0.20	0.387	12	213	0.50	0.356	12
414	0.42	0.316	12	428	0.64	0.422	12
431	0.49	0.364	12	465	0.49	0.356	12
92	0.58	0.454	13	113	0.70	0.392	13
291	0.58	0.336	13	199	0.20	0.403	13
331	0.76	0.361	13	253	0.60	0.453	13
334	0.57	0.360	13	338	0.55	0.363	13
410	0.24	0.408	13	499	0.64	0.398	13
Gemiddelde score: 10.51				Gemiddelde score: 10.33			
α : 0.769				α : 0.769			
s_x: 4.04				s_x: 4.03			

Tabel 11.3 laat zien dat we er zeer goed in geslaagd zijn om twee parallelle toetsen samen te stellen die aan het model voldoen. De betrouwbaarheden zijn hoog en identiek, terwijl de gemiddelde scores en ook de standaarddeviaties van de toetsen nagenoeg gelijk zijn. Merk op dat er in het model geen voorwaarden voor de standaarddeviaties van de toetsen gespecificeerd werden. Ook wordt aan de voorwaarde voldaan dat er vijf items uit elke leerstofcategorie afkomstig moeten zijn. We zien dat de itemparameters binnen elke leerstofcategorie niet gelijk zijn en dat we dus zwak-parallelle toetsen samengesteld hebben.

11.4 Het samenstellen van toetsen in de generaliseerbaarheidstheorie.

In de bespreking van de generaliseerbaarheidstheorie (Cronbach et al., 1972) in hoofdstuk 3 werd een onderscheid gemaakt tussen een generaliseerbaarheidsstudie (G-studie) en een decisiestudie (D-studie). Hier laten we zien hoe de schattingen van variantiecomponenten uit een G-studie gebruikt kunnen worden in een D-studie om te bepalen hoeveel observaties, meestal items of vragen, er per meetobject, meestal een persoon, nodig zijn om de belangrijkste foutenbronnen te controleren of om een gewenste generaliseerbaarheids-coëfficiënt te realiseren.
Voor designs met één facet kan het minimum aantal observaties per persoon als volgt bepaald worden. In hoofdstuk 3 werd de betrouwbaarheidscoëfficiënt van een één-facet random-model gekruist design, ρ^2, gedefinieerd als:

$$\rho^2 = \frac{\sigma_p^2}{\sigma_p^2 + \frac{\sigma_{res}^2}{n_v}}, \tag{11.34}$$

waarbij σ_p^2 de variantiecomponent voor personen, σ_{res}^2 de variantiecomponent voor de persoon × facet *v* interactie plus de meetfouten, en n_v het aantal observaties of condities van facet *v* in de D-studie is. Wanneer we (11.34) herschrijven en voor ρ^2 een specifieke betrouwbaarheidscoëfficiënt nemen, dan is het minimum aantal observaties per persoon voor het realiseren van die specifieke coëfficiënt gelijk aan:

$$n_v = \frac{\rho^2\sigma_{res}^2}{\sigma_p^2 - \rho^2\sigma_p^2}. \tag{11.35}$$

Zowel (11.34) als (11.35) illustreren de Spearman-Brown formule uit de klassieke testtheorie: verhoging (verlaging) van het aantal items resulteert in een verhoging (verlaging) van de betrouwbaarheid.
De Spearman-Brown formule kan ook als optimaliseringsprobleem geschreven worden:

minimaliseer n_v (11.36)

onder voorwaarde dat $\rho^2 = g$. (11.37)

In dit optimaliseringsprobleem staat in de doelfunctie (11.36) dat het aantal items, n_v, geminimaliseerd moet worden. In voorwaarde (11.37) staat ρ^2 voor de

betrouwbaarheids-coëfficiënt en g voor de waarde van een specifieke betrouwbaarheidscoëfficiënt.
Aangezien de waarde van de doelfunctie, het aantal items, per definitie geheeltallig is, is het bovenstaande model geformuleerd als:

minimaliseer n_v (11.38)

onder voorwaarde dat $\rho^2 \geq g$, (11.39)

n_v geheeltallig. (11.40)

De opname van drempelvoorwaarde (11.39), een relaxatie van (11.37), en de geheeltalligheidseis (11.40) zorgen voor een oplossing met een geheeltallig aantal items. Vanwege dat laatste kunnen de vergelijkingen (11.38), (11.39) en (11.40) beschouwd worden als een generalisatie van de Spearman-Brown formule voor één-facet designs.

De Spearman-Brown formule, dat wil zeggen de samenhang tussen aantal observaties en betrouwbaarheid, geldt niet voor designs die uit verschillende facetten bestaan. We lichten dit toe aan de hand van het twee-facet random-model gekruist design. De generaliseerbaarheidscoëfficiënt voor dit design is gedefinieerd als:

$$\rho^2 = \frac{\sigma_p^2}{\sigma_p^2 + \frac{\sigma_{pv}^2}{n_v} + \frac{\sigma_{pb}^2}{n_b} + \frac{\sigma_{res}^2}{n_v n_b}}, \qquad (11.41)$$

waarbij σ_p^2 de variantiecomponent voor personen, σ_{pv}^2 de variantiecomponent voor de persoon × facet v interactie, σ_{pb}^2 de variantiecomponent voor de persoon × facet b interactie, σ_{res}^2 de variantiecomponent voor de persoon × facet v × facet b interactie plus de meetfouten is, n_v en n_b de aantallen condities van respectievelijk facet v en facet b in de D-studie zijn. Het totale aantal observaties voor dit design wordt aangegeven met $L = n_v\ n_b$, het produkt van het aantal condities van de twee facetten. Aan formule (11.42) kunnen we zien dat het verhogen van bijvoorbeeld het aantal condities van een facet met een grote foutenvariantie een groter effect zal hebben op de generaliseerbaarheidscoëfficiënt dan het verhogen van het aantal condities van een facet met een kleine foutenvariantie. Met multi-facet designs is het dan ook mogelijk dat de generaliseerbaarheidscoëfficiënt verhoogd wordt terwijl het aantal observaties verlaagd wordt. Vanwege het multi-dimensionale karakter van de foutenvariantie in de generaliseerbaarheidstheorie, is het probleem van het bepalen van het minimum aantal

observaties veel complexer voor multi-facet designs dan voor één-facet designs. Sanders, Theunissen en Baas (1989) laten zien hoe dit probleem met behulp van een branch-and-bound algoritme kan worden opgelost. Hiervoor wordt het probleem eerst in termen van mathematische programmering geformuleerd als:

minimaliseer $L = n_v n_b$ (11.42)

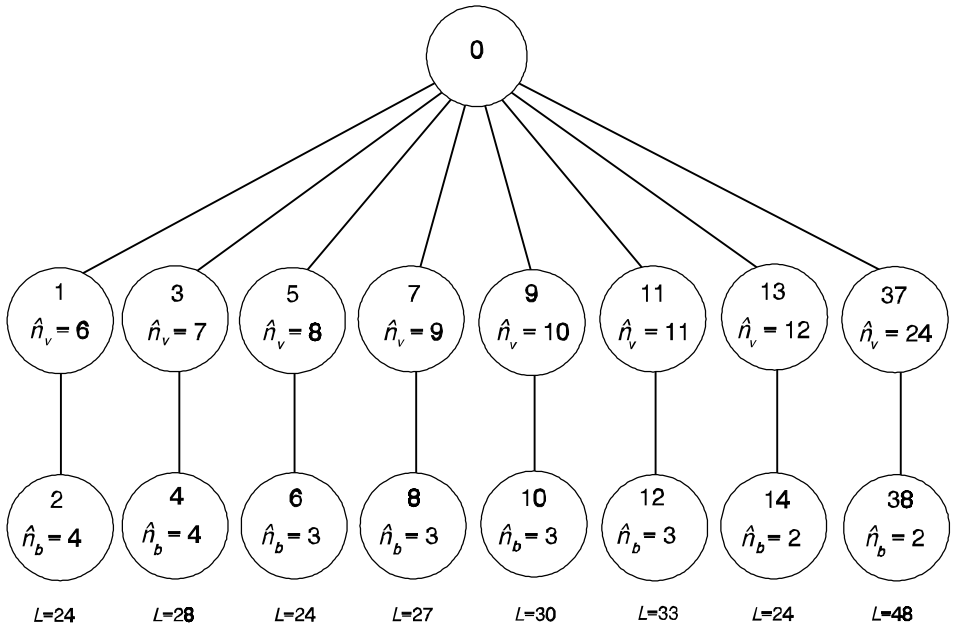
onder voorwaarde dat $\rho^2 \geq g$, (11.43)

$n_v \geq n_b$, (11.44)

n_v en n_b geheeltallig. (11.45)

In de formulering van dit optimaliseringsprobleem is L de waarde van de doelfunctie (11.42) als verschillende aantallen condities, n_v en n_b, voor facet v en facet b gebruikt worden. In de drempelvoorwaarde (11.43) staat ρ^2 voor de generaliseerbaarheidscoëfficiënt van een twee-facet random-model gekruist design en g voor de laagste waarde van de generaliseerbaarheidscoëfficiënt die als acceptabel beschouwd wordt. Voorwaarde (11.44) is een van de vele lineaire ongelijkheidsvoorwaarden die gebruikt kunnen worden. Voor het onderstaande geldt dat een optimale oplossing voor het twee-facet design probleem ook zonder deze voorwaarde verkregen kan worden. Het voordeel van een algoritme met deze voorwaarde is echter dat het irrelevante deel van de oplossingsruimte uitgesloten wordt, waardoor het aantal vertakkingen van de branch-and-bound gereduceerd wordt. In voorwaarde (11.45) staat dat mogelijke waarden voor n_v en n_b geheeltallig moeten zijn.

Nadat het probleem geformuleerd is als een optimaliseringsprobleem, worden grenzen geconstrueerd om het zoekproces te reduceren. In Sanders, Theunissen en Baas (1989) staat hoe die grenzen bepaald worden. Het zoekproces voor een twee-facet random-model gekruist design met $\hat{\sigma}^2_p = 5.435$, $\hat{\sigma}^2_{pv} = 3.421$, $\hat{\sigma}^2_{pb} = 1.140$, $\hat{\sigma}^2_{res} = 11.850$ en $g \geq .80$ staat in de zoekboom in figuur 11.9. De nummering van de knopen geeft aan hoe het zoekproces verloopt. De generaliseerbaarheidscoëfficiënten voor verschillende aantallen condities staan vermeld in tabel 11.4.



Figuur 11.9
Zoekboom van het twee-facet voorbeeld

De startoplossing $(\hat{n}_v = \hat{n}_b = 6)$ met een waarde voor de doelfunctie gelijk aan 36, is in knoop 2 van figuur 11.9 vervangen door een nieuwe 'beste' oplossing met de waarde 24. Oplossingen met dezelfde waarde voor de doelfunctie als knoop 2 zijn $(\hat{n}_v, \hat{n}_b) = (8,3)$ in
knoop 6 en $(\hat{n}_v, \hat{n}_b) = (12,2)$ in knoop 14. Het zoekproces eindigt in knoop 38 met de oplossing $(\hat{n}_v, \hat{n}_b) = (24,2)$ met de waarde 48 voor de doelfunctie die hoger is dan de tot dan toe beste oplossing. Op het eind van het zoekproces blijken er dus drie kandidaten voor een optimale oplossing te zijn: $(\hat{n}_v, \hat{n}_b) = (6,4)$, $(\hat{n}_v, \hat{n}_b) = (8,3)$ en $(\hat{n}_v, \hat{n}_b) = (12,2)$. Volgens tabel 11.4 zou $(\hat{n}_v, \hat{n}_b) = (8,3)$ als de optimale oplossing beschouwd kunnen worden omdat het in een hogere generaliseerbaarheidscoëfficiënt resulteert dan de andere oplossingen. Veelal zullen echter ook andere overwegingen dan het realiseren van een specifieke generaliseerbaarheidscoëfficiënt een rol spelen wanneer een meetinstrument geconstrueerd wordt. Als in het voorbeeld facet *v* items zouden zijn en facet *b* beoordelaars, dan zouden er aanzienlijke verschillen in de kosten per conditie van deze twee facetten bestaan. Omdat beoordelaars waarschijnlijk duurder zijn dan items, zal in het algemeen de voorkeur gegeven worden aan meer items en minder beoordelaars te nemen. Hiervoor dient de voorwaarde $n_v \geq n_b$ vervangen te

worden door een voorwaarde als $n_v \geq 5n_b$. Deze voorwaarde en de specificatie $g \geq 0.80$ geeft de optimale oplossing $(n_v, n_b) = (12, 2)$.

Tabel 11.4.
Waarden voor n_v, n_b, L, variantiecomponenten en ρ^2

n_v	n_b	L	$\hat{\sigma}_p^2$	$\frac{\hat{\sigma}_{pv}^2}{n_v}$	$\frac{\hat{\sigma}_{pb}^2}{n_b}$	$\frac{\hat{\sigma}_{res}^2}{n_v n_b}$	ρ^2
6	4	24	5.4	.57017	.285	.49375	.80166
6	6	36	5.4	.57017	.190	.32917	.83303
7	3	21	5.4	.48871	.380	.56429	.79135
7	4	28	5.4	.48871	.285	.42321	.81952
8	3	24	5.4	.42763	.380	.49375	.80681
9	3	27	5.4	.38011	.380	.43889	.81926
10	3	30	5.4	.34210	.380	.39500	.82951
11	3	33	5.4	.31100	.380	.35909	.83808
12	2	24	5.4	.28508	.570	.49375	.80117
24	2	48	5.4	.14254	.570	.24688	.84996
36	2	72	5.4	.09503	.570	.16458	.86757

Tabel 11.4 laat zien dat hoewel de verschillen tussen aantallen condities soms aanzienlijk zijn, de verschillen tussen de generaliseerbaarheidscoëfficiënten van die designs slechts gering zijn. Dat heeft te maken met de ongevoeligheid van hogere waarden van de coëfficiënt voor zelfs ingrijpende wijzigingen in het design. Let wel dat het verschil van slechts één conditie voor één facet een substantieel verschil kan betekenen voor te maken onderzoekskosten en dergelijke. Met een twee-facet gekruist design kan het verschil van één conditie betekenen dat één beoordelaar minder nodig is om bijvoorbeeld de antwoorden van honderd studenten op tien vragen te beoordelen.