Logica
voor
alfa's en informatici

Jan van Eijck & Elias Thijsse

Academic Service, Schoonhoven 1989

Inhoud

Hoofdstuk 1

Algemene inleiding

1.1 Doelstelling

De logica heeft een respectabele traditie die teruggaat tot vóór Aristoteles. Tot aan het midden van de vorige eeuw was logica een integraal onderdeel van de filosofie, maar toen het vak door het werk van Gottlob Frege een nieuwe impuls kreeg zijn ook wiskundigen zich met de ontwikkeling ervan gaan bemoeien, met als gevolg dat logica het werktuig bij uitstek werd voor wiskundig grondslagenonderzoek. Toen in de jaren dertig van deze eeuw de informatica ontstond was het de wiskundige en logicus Alan Turing die de theoretische computerwetenschap stevig verankerde in de logica. Twintig jaar later werd de taalwetenschap op een nieuwe leest geschoeid door Noam Chomsky, die daarmee de basis legde voor een samenwerking tussen logici en taalkundigen. En aan het eind van de zestiger jaren liet de Amerikaanse logicus Richard Montague zien dat de betekenis van zinnen uit de taal van alledag in principe beschreven kan worden met machinerie uit de logica.

Deze snelle historische vogelvlucht maakt duidelijk dat logica vandaag de dag een vak is dat ligt op het grensgebied van verschillende wetenschapsterreinen, een vak ook waaraan door vogels van diverse pluimage wordt gewerkt. Gevolg is dat logica op tal van manieren kan worden gepresenteerd. Hoe verteerbaar het resulterende gerecht is hangt af van de interesse en de voorkennis van de consument. Dit boek beoogt een inleiding te geven in de logica voor lezers die niet bij uitstek wiskundig geschoold zijn, maar die belangstelling hebben voor informatica of taalkunde. Daarom is gekozen voor een benadering die de formele aspecten niet schuwt, maar die geen wiskundige voorkennis veronderstelt.

1.2 Gebruiken en noemen van uitdrukkingen

We beginnen dit inleidende hoofdstuk met een paar opmerkingen over het verschil tusen het *gebruiken* en het *noemen* van woorden. In de zin die we nu opschrijven wordt de letter *a* één keer genoemd en nergens gewoon gebruikt. Diezelfde letter wordt viermaal gebruikt maar nergens direct genoemd in deze zin. Het onderscheid tussen gebruiken en noemen (Engels: use versus mention) is niet alleen van toepassing op schrijftekens, maar ook op woorden of taal-uitdrukkingen in het algemeen.

Wanneer we over eigenschappen van taal spreken moeten we vaak talige uitdrukkingen *noemen*. Voorbeeld:

(1) *In een Nederlandse zin kan* de *nooit het onderwerp zijn.*

U gelieve zelf na te gaan waarom in deze zin niet wordt gezondigd tegen het principe dat de zin uitdrukt.

Het *noemen* van taaluitingen gebeurt ook bij het gebruik van directe rede:

(2) *'Donder op, klootzak, uit m'n ogen', zei ze.*

De aanhalingstekens worden gebruikt om een passage die letterlijk voorkwam te *noemen*. In de indirecte rede ziet het verslag er wellicht heel anders uit:

(3) *Ze zei dat het misschien verstandiger was als we elkaar voorlopig even niet zouden zien.*

Het onderscheid tussen noemen en gebruiken van een taal-uitdrukking zullen we in dit boek maken met behulp van cursivering (dat daarnaast gebruikt wordt voor nadruk), of door enkele of dubbele aanhalingstekens te gebruiken, of door een uiting uit de context van de lopende tekst te halen en er wit omheen te zetten (en eventueel een voorbeeldnummer ervoor).

Het onderscheid tussen noemen en gebruiken is context-afhankelijk. In de tekst wordt voorbeeldzin (2) genoemd in plaats van gebruikt. In de voorbeeldzin zelf wordt het gedeelte tussen enkele aanhalingstekens genoemd in plaats van gebruikt.

1.3 Formele en empirische wetenschap

Logica is een formele wetenschap. Dit wil zeggen: zintuiglijke kennisverwerving (en dus: kennis over de zintuiglijk waarneembare buitenwereld) speelt bij het verder ontwikkelen van deze wetenschap geen rol. De formele wetenschap bij uitstek is de wiskunde. Formele wetenschappen staan tegenover

empirische wetenschappen. Een empirische wetenschap is een wetenschap waarbij empirische kennisverwerving wel een rol speelt. Voorbeelden van empirische wetenschappen zijn de natuurkunde, de biologie, de psychologie en de (empirische) taalkunde.

Voor het opstellen van een nieuwe logica-theorie is het onnodig om feitenmateriaal te verzamelen, enquêtes te houden, enzovoort. Net zo voor het opstellen van nieuwe theorieën in de wiskunde. Voor het *toepassen* van logische theorieën op de werkelijkheid, bij voorbeeld voor het ontwerpen van computerschakelingen, liggen de zaken anders. Datzelfde geldt voor het toepassen van wiskundige theorieën in de studie van de natuur. Zulke toepassingen veronderstellen kennis van het aspect van de werkelijkheid waarop de formele theorie wordt toegepast.

Het onderscheid tussen formele en empirische wetenschap wordt wel uitgedrukt in termen van het filosofische begrippenpaar kennis *a priori* versus kennis *a posteriori*. Een bewering die *a priori* waar of onwaar is, is een bewering waarvan de waarheid in principe kan worden vastgesteld met behulp van methoden waarin zintuiglijke waarneming geen rol speelt. Een bewering die *a posteriori* waar of onwaar is, is een bewering waarvan de waarheid of onwaarheid alleen kan worden vastgesteld op grond van zintuiglijke ervaring. Logica, wiskunde en formele taalkunde houden zich bezig met *a priori* theorieën, terwijl vakken zoals empirische taalkunde, natuurkunde en biologie zich bezighouden met *a posteriori* theorieën.

Zoals de titel van dit boek aangeeft willen wij logica presenteren aan alfa's (in het bijzonder: mensen met belangtelling voor empirische taalkunde) en aan informatici (in het bijzonder: mensen met belangstelling voor praktische informatica). Wat is het belang van het aanleren van formele vakken zoals logica (of formele taalkunde en automatentheorie) voor de beoefening van empirische taalkunde en voor de praktijk van de informatica?

Eerst het belang van logica voor de informatica. Het beschrijven van de *betekenis* van constructies uit programmeertalen met behulp van het begrippenapparaat uit de logica (of verruimingen daarvan) heeft vrucht gedragen. Het blijkt mogelijk om het *effect* van afzonderlijke programmeeropdrachten te beschrijven met behulp van *paren* van logische omschrijvingen. Hierbij geeft het tweede lid van een paar een omschrijving van de situatie die ontstaat wanneer de opdracht wordt uitgevoerd terwijl de computer in een toestand is waarin het eerste lid van het paar waar is. Door nu volgens de regels van deze omschrijvingemethode, ontwikkeld door C.A.R. Hoare en R.W. Floyd, een programma te annoteren met logische formules kan een programmeur zich vergewissen van de correctheid van dat programma. In de tweede plaats is de logica inspiratiebron geweest voor een geheel nieuwe stijl van programmeren, het zogenaamde *declaratief programmeren* dat wordt beoefend in programmeertalen zoals PROLOG.

Dan het verband tussen logica en empirische taalkunde. Empirische taalkunde houdt zich bezig met de bestudering van in de loop van de geschiedenis langzaam geëvolueerde mensentalen zoals het Nederlands, het Russisch of het Engels. Logica en formele taalkunde houden zich bezig met *geconstrueerde* talen: de taal van de propositielogica, de taal van de predikatenlogica (twee talen waarmee u verderop zult kennismaken), de programmeertaal **Pascal**, enzovoorts. Talen van het eerste soort zullen we vanaf nu *natuurlijke talen* noemen. Talen van het tweede soort heten *formele talen*.

Zo op het eerste gezicht lijkt het misschien alsof talen van het eerste soort weinig uitstaande hebben met talen van het tweede soort. Het zou kunnen zijn dat het Nederlands en de taal van de predikatenlogica evenveel of even weinig met elkaar gemeen hebben als een zwaluw en een Fokker F-27. Als dat het geval is, dan heeft het voor geïnteresseerden in de empirische taalwetenschap weinig zin om zich te verdiepen in formele vakken. Aan de andere kant: als het mogelijk is om een theorie te ontwerpen die de overeenkomsten tussen zwaluwen en Fokker Friendships blootlegt, dan moet dat wel een erg fundamentele theorie zijn, juist *omdat* deze twee soorten van dingen op het eerste gezicht weinig gemeen lijken te hebben. Net zo: een theorie die iets interessants weet te zeggen met betrekking tot zowel natuurlijke talen en formele talen kan zeer fundamentele inzichten verschaffen.

Logica is op minstens drie punten relevant voor de empirische taalkunde. In de eerste plaats heeft de logica *formele methoden* ontwikkeld om te definiëren wat de uitdrukkingen zijn van een bepaalde *symbooltaal*. Deze definitie-methoden—die ook gebruikt worden om programmeertalen te definiëren—zijn van groot nut gebleken voor het bestuderen van de uitdrukkingen van een natuurlijke taal (het Nederlands, het Engels); niet dat je de uitdrukkingen van het Nederlands kunt 'afbakenen' zoals je de formules van een logische taal of de klasse van syntactisch correcte **Pascal** programma's kunt afbakenen, maar je hebt dezelfde definitie-methoden nodig (al heb je er niet genoeg aan).

In de tweede plaats zijn logische talen vanwege hun grote precisie zeer geschikt om over betekenis-aspecten van natuurlijke taal te spreken. De uitdrukkingen van de symbooltalen die in de logica zijn ontwikkeld zijn altijd maar voor één manier van lezen vatbaar. Daar zijn die uitdrukkingen op gebouwd, zogezegd. Dat dit laatste bij natuurlijke taal niet het geval is moge blijken uit de volgende voorbeelden:

(4) *Lolita was jong en mooi of verdorven.*

(5) *Iedereen gelooft mij niet.*

Logische talen kunnen hier gebruikt worden als medium voor precisering van dubbelzinnige uitdrukkingen uit de natuurlijke taal.

Van voorbeeldzin (4) zijn bij voorbeeld de volgende twee preciseringen mogelijk in de taal van de propositielogica:

- $(p \wedge (q \vee r))$
- $((p \wedge q) \vee r)$

Hierbij staan de letters p, q en r respectievelijk voor 'Lolita was mooi', 'Lolita was jong' en 'Lolita was verdorven'; $\wedge$ staat voor 'en' en $\vee$ voor 'of'.

Voor voorbeeldzin (5) schaft de predikatenlogica raad. Hier zijn de twee mogelijke parafrases:

- $\forall x \neg Gxm$
- $\neg \forall x Gxm$

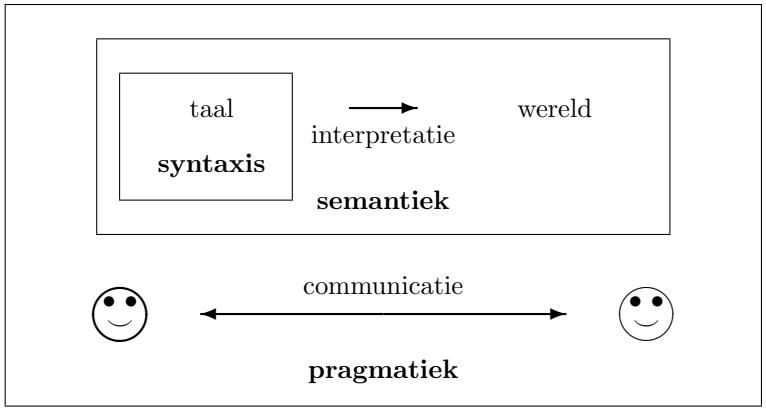
Legenda: $\forall x \ldots$: "voor alle x geldt dat…"; $\neg$: "niet"; Gxm: "x gelooft mij" (dat wil zeggen: "x gelooft wat ik zeg"). Nadere details volgen in de hoofdstukken 5 en 6.

De mogelijkheden die de logica biedt om over betekenisaspecten van taal te spreken gaan overigens nog veel verder, en daarmee komen we bij het derde raakvlak tussen logica en empirische taalkunde. In de logica is een zeer precieze uitleg voorhanden van het begrip 'betekenis' voor de formules uit een logische taal. Het ligt voor de hand om te proberen die uitleg (of een toepasselijke verruiming ervan) over te planten naar natuurlijke taal. Onderzoek in dit kader—onderzoek naar de "modeltheoretische semantiek van natuurlijke taal"—is zeer vruchtbaar gebleken, en dit onderzoek wordt momenteel toegepast in programma's die computergebruikers de mogelijkheid moeten bieden om in gewoon Engels of Nederlands te communiceren met een computer.

Wanneer we even afzien van aspecten van woordvorming of akoestische representatie (dat wil zeggen, wanneer we fonetiek, fonologie en morfologie buiten beschouwing laten), dan valler er in de studie van natuurlijke talen de volgende aspecten te onderscheiden:

- **syntaxis:** de studie van zinnen, zinsbouw, zinsontleding, grammaticaliteit;
- **semantiek:** de studie van uitdrukkingen van de taal in relatie tot wat ze uitdrukken (hun betekenis);
- **pragmatiek:** de studie van uitdrukkingen, inclusief hun betekenis, in diverse gebruikscontexten.

De syntaxis, semantiek en pragmatiek van een taal verhouden zich als volgt tot elkaar:



Dit plaatje maakt duidelijk wat de *hiërarchie* is tussen de aspecten 'syntaxis', 'semantiek' en 'pragmatiek'. Het bestuderen van de semantiek van een taal veronderstelt een grote mate van inzicht in de syntaxis, maar andersom is dat veel minder het geval. Het bestuderen van de pragmatiek van een taal veronderstelt inzicht in syntaxis en semantiek van die taal, en van de samenhang daartussen, maar niet of nauwelijks andersom. Het plaatje suggereert misschien ook een hiërarchie in respectabiliteit van de drie disciplines 'syntaxis', 'semantiek' en 'pragmatiek'. Op grond van het plaatje zou je syntaxis kunnen beschouwen als hulpje van de semantiek, en semantiek + syntaxis als hulpje bij de pragmatiek. In de praktijk van het onderzoek van natuurlijke talen liggen de zaken echter heel anders: syntaxis is relatief het verst ontwikkelde vak, semantiek is een goede tweede, maar helaas: de pragmatiek is een zorgenkindje dat nog niet zo goed wil groeien.

Dat het heel goed mogelijk is syntaxis te bedrijven zonder je om semantiek te bekommeren wordt duidelijk in de formele taaltheorie. Hier wordt een (formele) *taal* doorgaans beschouwd als een verzameling uitdrukkingen, zonder dat de leden van die verzameling een of andere plausibele betekenis hoeven te hebben.

Formele talen waarbij het wel zin heeft om te spreken over de betekenis van de uitdrukkingen van de taal zijn er ook. Voorbeelden zijn: de taal van de propositielogica, die van de predikatenlogica, en de programmeertaal **Pascal**. Ook bij natuurlijke talen gaan we ervan uit dat welgevormde uitdrukkingen van zo'n taal in principe dragers van betekenis zijn. De in de syntaxis van zo'n taal bestudeerde taaluitingen hebben een *betekenis* die in principe achterhaald kan worden door te kijken naar de betekenis van de woorden die erin voorkomen. Sommige woorden in een zin verwijzen naar *objecten* in de wereld: "Prins Claus", "de vliegbasis Gilze-Rijen". Andere verwijzen naar *eigenschappen van objecten*: "rood", "ziek". Weer

andere slaan op *relaties tussen objecten*: “bewondert”, “bemint”, “haat”. Nog weer andere hebben de logische functie van ‘bindmiddel’: “niet”, “en”, “of”, “elke”.

Door de informatie die de verschillende onderdelen van een zin leveren te combineren krijgen we de betekenis van de zin als geheel. We begrijpen op deze manier “Jan haat de vliegbasis Gilze-Rijen” op grond van het feit dat we weten waar “Jan” en “de vliegbasis Gilze-Rijen” op slaan, en welke relatie met “haat” wordt aangeduid. *Begrijpen* van de zin is overigens nog iets anders dan weten of de zin waar is. Begrijpen van een zin is veeleer iets in de trant van: weten *onder welke omstandigheden een zin waar is*, en dat is iets heel anders. We kunnen immers weten wat “haten” betekent zonder dat we precies weten wie wie of wat haat. Het bedrijven van semantiek is een vorm van begripsanalyse, en heeft niets te maken met verificatie van feiten of verwerven van kennis over de werkelijkheid.

Voor de conceptuele analyse die door semantici wordt bedreven is het meestal niet nodig om de werkelijkheid als geheel in de beschouwingen te betrekken. Vaak is alleen een klein stukje daarvan relevant voor het begrijpen van een tekst, en in een rationele reconstructie kunnen we dat gedeelte schematiseren tot een zogenaamd *model*. We zullen zien dat in de formele logica de begrippen “syntaxis”, “semantiek”, “interpretatie” en “model” exact gedefinieerd zijn.

1.4 Kennismaking met het jargon

In dit boek zullen we—terwille van de oriëntatie-mogelijkheden in de veelal Engelstalige vakliteratuur—naast het Nederlandse jargon ook waar nodig wat Engelse termen laten vallen, zonder ons al te zeer te bezondigen aan het oplepelen van half-verteerde brokstukken Engels vocabularium met een Nederlands syntactisch sausje, niet ongebruikelijk onder Nederlandse logici, taalkundigen en (vooral) informatici. In plaats van “deze propositie is vals” zullen we dus zeggen: “deze bewering is onwaar”, en “een value-assignment voor de variabelen” zullen we vervangen door: “het toekennen van waarden aan de variabelen”, of “een bedeling voor de variabelen”. Met het uitleggen van (nuttig, want het alledaags spraakgebruik aanvullend en preciserend) jargon maken we in deze paragraaf alvast een begin.

Vele taalfilosofen maken onderscheid tussen *zinnen* (Engels: sentences), *uitspraken* of *beweringen* (Engels: statements), *proposities* (Engels: propositions), *standen van zaken* (Engels: states of affairs), en nog zo het een en ander. Soms is het gemakkelijk om te kunnen zeggen dat verschillende *zinnen* een en dezelfde *bewering* kunnen uitdrukken. Bij voorbeeld: “het sneeuwt” en “il neige” drukken hetzelfde (dezelfde bewering) uit. Over de

andere onderscheidingen zullen we ons niet druk maken.

Vaak wordt onderscheiden tussen de *extensie* en de *intensie* van een woord of zin. Deze termen stammen uit de traditionele filosofie. De *intensie* van een woord is—ongeveer—de gedachte-inhoud van dat woord, ofwel: het geheel van wat een gebruiker van dat woord moet weten om het woord correct te kunnen gebruiken. De intensie of gedachte-inhoud van *de koning(in) der Nederlanden* is zoiets als "de persoon die in Nederland de erfelijke functie van staatshoofd bekleedt".

De *extensie* van een woord is datgene waar dat woord naar verwijst. De extensie van *de koningin der Nederlanden* is Beatrix. De extensie van *Nederlandse hoogleraar* is de verzameling van alle individuen die nu in Nederland de functie van hoogleraar bekleden. Soms worden in plaats van *extensie* ook de termen *referentie*, *verwijzing* of *denotatie* gebruikt. We zeggen ook: de spreker *refereert/verwijst* naar een (buitentalig) object met behulp van een (talige) uitdrukking. Het object waarnaar met een uitdrukking wordt verwezen wordt de *referent* van die uitdrukking genoemd.

De extensie van *eenhoorn* bevat niets. In het wiskundige jargon dat u spoedig (weer) vertrouwd in de oren zal klinken: het is de lege verzameling. De reden is dat er geen eenhoorns zijn (hoewel ze voortdurend opdraven om als voorbeelden te dienen in taalfilosofische discussies over intensies en extensies).

In de moderne semantiek wordt er een truc toegepast om *intensie* te definiëren in termen van *extensie*. De intensie van *de koningin der Nederlanden* is nu een recept om per situatie de juiste extensie uit te kiezen. Dus, variërend in tijd tussen 1945 en nu: Wilhelmina, Juliana, Beatrix. De intensie van *eenhoorn* kan in situaties die voldoende van onze prozaïsche werkelijkheid verschillen wel degelijk een verzameling zachtaardige, schuwe en ge-eenhoornde viervoeters opleveren. Vandaar dat we kunnen zeggen dat de betekenis (de intensie) van *gevleugeld paard* en *eenhoorn* van elkaar verschillen, terwijl de verwijzing-hier-en-nu (de extensie) in beide gevallen hetzelfde is, namelijk de lege verzameling.

Een terminologisch onderscheid dat van Frege stamt is dat tussen *Sinn* (Engels: sense) en *Bedeutung* (Engels: reference). Hierover zijn bibliotheken volgeschreven (zie bij voorbeeld [Dummett 1973] voor een uitvoerige beschouwing en verdere literatuurverwijzingen). In de moderne semantiek gaat men er meestal van uit dat het nieuwerwetse onderscheid tussen *intensie* (in de zin van: recept) en *extensie* Frege's onderscheid op een acceptabele manier preciseert. Zie voor deze en verwante begripsonderscheidingen ook [Lewis 1972].

Een begrip, tenslotte, waarover in kringen van empirisch taalkundigen veel gesproken wordt is het begrip *logische vorm*. In de Transformationeel-Generatieve traditie in de taalwetenschap (dat wil zeggen, de Chomsky-

traditie) slaat dit op een syntactische representatie van een zin, met daarin extra informatie gecodeerd die de dubbelzinnigheden eruit haalt. Daarbij kan het bij voorbeeld gaan om de dubbelzinnigheden die veroorzaakt worden door de aanwezigheid van pronomina of van *logische operatoren* (die we hierboven 'logisch bindmiddel' genoemd hebben). Voorbeelden:

(6) *Iedereen zei dat* hij *kwam.*

(7) *Alle hout is geen timmerhout.*

Zoals we hierboven al hebben aangegeven kan een dergelijke desambiguering ook bewerkstelligd worden door vertaling in een geschikte logische taal. Dergelijke desambiguerende vertalingen ('analyses') worden daarom ook wel "logische vormen" genoemd. Echter: logische vormen zijn nu niet meer *uniek*. Ze hangen nu in de eerste plaats van het gekozen analysemedium af (propositielogica, predikatenlogica, of nog een andere logische taal). In de tweede plaats zijn er zelfs binnen een formele taal in het algemeen meerdere logische formules mogelijk, omdat die formules *logisch gelijkwaardig* (of: *logisch equivalent*) zijn. Twee formules zijn logisch equivalent wanneer die formules dezelfde interpretatie krijgen, ongeacht hoe het model eruit ziet waarin wordt geïnterpreteerd. De volgende twee *logische vormen* (vertalingen in de predikatenlogica) voor "Niemand houdt van Marietje" zijn bij voorbeeld equivalent:

- $\neg\exists x\ Hxm$ (letterlijk: "niet: iemand houdt van Marietje")
- $\forall x\neg Hxm$ (letterlijk: "Iedereen houdt-niet-van Marietje")

In discussies over 'logische vorm' zijn spraakverwarringen aan de orde van de dag omdat de discussiepartners vaak niet hetzelfde bedoelen met de term. Een grondige vertrouwdheid met logica kan u tegen veel vruchteloos gekissebis behoeden. Voor we aan de logica zelf toe zijn moet er echter wat wiskundig voorwerk worden verricht.

Hoofdstuk 2

Kernbegrippen uit de naïeve verzamelingenleer

2.1 Terminologie en notatie-afspraken

Wat verzamelingen zijn weet u eigenlijk al, want u begrijpt wat er bedoeld wordt wanneer iemand het heeft over de verzameling van alle Nederlanders boven 21 jaar, en u hebt een idee van wat een postzegelverzameling is.

In de wiskunde is een *verzameling* simpelweg een of andere collectie van dingen (Engels voor 'verzameling': *set*). De dingen die samen in een verzameling zitten heten de *leden* of *elementen* van die verzameling (Engels: *members*, *elements*). We kunnen verzamelingen namen geven, bij voorbeeld door ze aan te duiden met hoofdletters van het Romeinse alfabet. We kunnen dan bij voorbeeld spreken over een of andere verzameling A.

Het begrip 'verzameling' is een grondbegrip dat zelf niet gedefinieerd kan worden. De omschrijving in termen van 'collectie van objecten' die we u zojuist hebben laten slikken is geen definitie maar een verschuiving van het definitie-probleem: immers, wat is een collectie? Net zo voor het begrip 'element': ook hier moeten we ons tevreden stellen met een vage kenschets. 'Verzameling' en 'element' zijn de meest fundamentele begrippen uit het wiskundige woordenboek: ze worden gebruikt om andere begrippen te definiëren, maar ze worden zelf niet gedefinieerd. U wordt geacht ermee vertrouwd te zijn; als dat niet zo is raakt u er vanzelf mee vertrouwd door dit hoofdstuk te lezen.

Verzamelingen spelen in wiskunde en logica een grote rol. Verzamelingenleer is een relatief jong onderdeel van de wiskunde, begonnen met het werk van Georg Cantor (1845–1918). Werken met verzamelingen maakt het

mogelijk om snel complexe structuren te maken. Nieuwe objecten kunnen worden gecreëerd door lagere te verzamelen; die nieuwe objecten kunnen heel andere eigenschappen hebben dan hun elementen. Om dit laatste in te zien hoeft u niets van wiskunde te weten: een Nederlandse postzegel kan de eigenschap hebben dat er een portret van Beatrix op staat, maar een verzameling van Nederlandse postzegels heeft die eigenschap niet.

De elementen van een verzameling A hoeven niets met elkaar gemeen te hebben (behalve dan het feit dat ze samen in A zitten). De elementen van een verzameling hoeven ook niet per se *materiële* dingen te zijn. Uit het feit dat de elementen van een verzameling abstracte dingen mogen zijn volgt dat die elementen bij voorbeeld zelf ook verzamelingen mogen zijn. Op deze mogelijkheid gaan we straks nader in.

Omdat de elementen van een verzameling niets met elkaar gemeen hoeven te hebben zouden we kunnen besluiten het getal 3, paus Johannes Paulus II en de stad Parijs samen in een verzameling te stoppen. Een manier om deze verzameling aan te duiden is:

$$A = \{3, \text{Johannes Paulus II}, \text{Parijs}\}.$$

De verzameling A wordt hier gekarakteriseerd door het opsommen van zijn elementen. Het feit dat het getal 3, de paus, en de stad Parijs samen in een verzameling zitten geven we aan met de accolades.

We kunnen verzamelingen op verschillende manieren definiëren. Wanneer het gaat om eindige verzamelingen (dat wil zeggen om verzamelingen waarvan je de leden kunt opsommen, zo dat die opsomming op een gegeven moment afgelopen is), dan kun je gewoon alle elementen *noemen*. Dit heet: definitie door opsomming. De introductie van de verzameling A hierboven is een voorbeeld van definitie door opsomming.

We kunnen verzamelingen ook invoeren door middel van *omschrijving*. Deze methode werkt ook voor oneindige verzamelingen. Voorbeeld:

$$B = \{x \mid x \text{ is een natuurlijk getal en } x \text{ is even}\}.$$

B is de verzameling van alle natuurlijke getallen (getallen uit het rijtje 0, 1, 2, 3, 4, enzovoorts) die deelbaar zijn door twee, ofwel: de verzameling van alle *even* natuurlijke getallen. Merk op dat B niet eindig is. B is verkregen door een deel te nemen van een andere verzameling. De verzameling van natuurlijke getallen beschouwen we als gegeven. Deze verzameling wordt vaak aangeduid als $\mathbf{N}$.

We voeren nu de volgende notatie in:

- $a \in B$ "a is een element van B."
- $a \notin B$ "a is geen element van B."

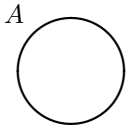
- $A \subseteq B$ "A is bevat in B"; "A is een deelverzameling van B" (dit wil zeggen: elk element van A is een element van B).
- $A \not\subseteq B$ "A is niet bevat in B"; "A is geen deelverzameling van B" (dit wil zeggen: niet elk element van A is een element van B).
- $A \subset B$ "A is echt bevat in B"; "A is een echte deelverzameling van B" (dit wil zeggen: elk element van A is een element van B en niet elk element van B is een element van A).
- $A \not\subset B$ "A is niet echt bevat in B"; "A is geen echte deelverzameling van B."

  Opmerking: om voor ons niet geheel duidelijke redenen zijn er in de middelbare school wiskunde notatie-afspraken gemaakt die enigszins afwijken van de internationaal gangbare afspraken die hierboven zijn vermeld. Met name:

  - $\subset$ wordt gebruikt voor "is bevat in".
  - $\underset{\neq}{\subset}$ wordt gebruikt voor "is echt bevat in".

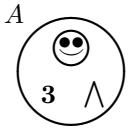
  In dit boek volgen wij het wetenschappelijk gangbare spraakgebruik, en *niet* de middelbare school notatie.

Om het denken over verzamelingen te vergemakkelijken is het nuttig om plaatjes te tekenen. Een verzameling A geven we als volgt aan met behulp van een cirkel:

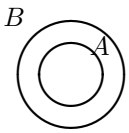


De punten die binnen de cirkel liggen zijn de elementen van A.

Wanneer A een eindige verzameling is, kunnen we de afzonderlijke elementen in de cirkel tekenen:



Het gegeven dat $A \subseteq B$ kan nu in het volgende plaatje worden uitgedrukt:



Als $A = B$ dan is het buitengebied leeg, als $A \subset B$ dan bevat het buitengebied een of meer elementen. We tekenen de plaatjes altijd zo algemeen mogelijk, dus we houden rekening met deze tweede mogelijkheid. Dit soort plaatjes heten Venndiagrammen (naar de logicus J. Venn, 1834–1923).

Verzamelingen die precies één element hebben worden *atomaire verzamelingen* of *singletons* genoemd (Engels: *singletons*, *singleton sets*). Dus: wanneer a een of ander willekeurig ding is, dan is $\{a\}$ een atomaire verzameling. De verzameling $\{a\}$ wordt wel aangeduid als: "singleton a". Let op: er is verschil tussen a en $\{a\}$, tussen het ding a en de atomaire verzameling met a als element. Iets concreter: er is verschil tussen de paus en de atomaire verzameling met de paus als element. Johannes Paulus II is een persoon; de verzameling met Johannes Paulus II als enige element is geen persoon, maar een eigenschap, namelijk de eigenschap Johannes Paulus II te zijn. In een plaatje:

de paus singleton de paus

We hebben hierboven opgemerkt dat de elementen van verzamelingen zelf ook weer verzamelingen kunnen zijn. Dus: als a en b willekeurige dingen zijn, dan is $\{a, b\}$ een verzameling. Maar nu is $\{\{a, b\}\}$ ook een verzameling, namelijk: de verzameling met de verzameling $\{a, b\}$ als enige element. Let op: $\{\{a, b\}\}$ en $\{a, b\}$ zijn verschillende verzamelingen; $\{\{a, b\}\}$ is een atomaire verzameling, $\{a, b\}$ is dat niet.

Opdracht 2.1 *Ga na of de volgende verzamelingen singletons zijn:*

1. $\{\{a\}\}$
2. $\{\{a\}, \{b\}\}$
3. $\{\{\{a\}, \{b\}\}\}$.

Opdracht 2.2 *Hier volgen wat beweringen over verzamelingen waarbij de zojuist ingevoerde notatie-afspraken worden gebruikt. Ga na wat de beweringen precies uitdrukken en of ze juist zijn. De objecten a en b zijn* verschillend*, en het zijn zelf geen verzamelingen.*

1. $a \in \{a\}$
2. $a \subseteq \{a\}$
3. $a \subseteq a$
4. $\{a\} \subseteq \{a\}$
5. $\{a\} \subseteq \{\{a\}\}$
6. $a \in \{a, b\}$
7. $\{a\} \in \{a, b\}$
8. $\{a\} \subset \{a, b\}$
9. $\{a\} \subseteq \{a, b\}$
10. $\{a\} \subseteq \{a, \{a\}\}$.

2.2 Gelijkheid van verzamelingen

Dat twee verzamelingen A en B aan elkaar gelijk zijn, kunnen we als volgt uitdrukken:

$$A = B.$$

Dat twee verzamelingen A en B *niet* aan elkaar gelijk zijn drukken we als volgt uit:

$$A \neq B.$$

Als twee verzamelingen aan elkaar gelijk zijn dan hebben ze dezelfde elementen. Met andere woorden: als $A = B$ dan geldt voor iedere x: $x \in A$ dan en slechts dan als $x \in B$. Met andere woorden: als $A = B$ geldt voor iedere x: als $x \in A$ dan $x \in B$ *en* als $x \in B$ dan $x \in A$. Nog anders gezegd: als $A = B$ dan geldt $A \subseteq B$ en $B \subseteq A$.

In het vervolg zullen we 'dan en slechts dan als' afkorten als *desda* (Engels voor 'dan en slechts dan als': *if and only if*; voor de afkorting 'desda': *iff*). In plaats van 'desda' wordt ook wel het symbool $\Longleftrightarrow$ gebruikt. De bewering 'p dan en slechts dan als q' is een combinatie van 'p indien q' (dat wil zeggen: 'als q dan p'), soms genoteerd als '$p \Leftarrow q$', en 'p slechts indien q' (dat wil zeggen 'als p dan q'), met notatie '$p \Rightarrow q$'.

Voor de hand liggende vraag: geldt nu ook *omgekeerd*, dat als twee verzamelingen A en B dezelfde elementen hebben, A gelijk is aan B? In de verzamelingenleer geldt de afspraak dat dit inderdaad zo is. Deze afspraak is vervat in het zogenaamde *extensionaliteits-axioma* of *axioma van uitgebreidheid*.

Axioma 2.1 (Extensionaliteit)

$A = B$ desda A en B dezelfde elementen hebben.

Het extensionaliteits-axioma kan als volgt worden geparafraseerd: $A = B$ desda voor alle x geldt: $x \in A$ desda $x \in B$. Wat het axioma in feite zegt is dat de identiteit van een verzameling volledig bepaald is door zijn elementen. Twee verzamelingen zijn alleen verschillend wanneer er een ding valt aan te wijzen dat element is van de ene maar niet van de andere verzameling. Uit het extensionaliteitsaxioma volgt meteen:

- $\{a, b\} = \{b, a\}$
- $\{a, b, c\} = \{b, c, a\}$.

Het maakt kennelijk niet uit in welke volgorde je de elementen van een verzameling opsomt. De elementen van een verzameling zijn *ongeordend*. Verder volgt uit het extensionaliteitsaxioma:

- $\{a, a\} = \{a\}$.

Immers, er valt geen element aan te wijzen dat in $\{a, a\}$ zit en niet in $\{a\}$, en omgekeerd net zo. Dus: het heeft geen zin om bij het opsommen van een verzameling een bepaald element meerdere keren te noemen.

Opdracht 2.3 *Ga na of de volgende verzamelingen singletons zijn:*

1. $\{\{a\}, \{a\}\}$
2. $\{\{a, b\}, \{b, a\}\}$.

2.3 De lege verzameling

Het ligt niet in het begrip verzameling opgesloten dat een verzameling minstens één element heeft. We kunnen ook verzamelingen met 0 elementen bekijken. Laten we een verzameling met 0 elementen een lege verzameling noemen. Het extensionaliteits-axioma levert nu op dat er *precies één* lege verzameling is. Immers, stel dat er meer dan één lege verzameling zou zijn. Dan zou je twee lege verzamelingen A en B kunnen nemen die dan—volgens het extensionaliteits-axioma—van elkaar zouden moeten verschillen in het feit dat A een element heeft dat B niet heeft of omgekeerd. Maar dit kan nu juist niet, omdat A en B allebei leeg zijn. Dus: A en B moeten—in tegenspraak met wat we hadden aangenomen—aan elkaar gelijk zijn. De slotsom is dat er precies één lege verzameling is. We noemen deze verzameling *de*

lege verzameling. De lege verzameling wordt vaak aangeduid als $\emptyset$. Soms wordt ook wel de notatie $\{\}$ gebruikt, maar wij houden het op $\emptyset$.

Bij de eerste kennismaking met de lege verzameling is het even wennen. De lege verzameling is een ding (zij het dan een abstract ding). Dit ding kan dus zelf voorkomen als element van verzamelingen. We kunnen dus bij voorbeeld hebben: $\{\emptyset\}$, ofwel: singleton de lege verzameling. Weer moeten we bedacht zijn op het verschil tussen $\emptyset$ en $\{\emptyset\}$. De lege verzameling is niet gelijk aan singleton de lege verzameling. Immers: $\emptyset$ heeft geen elementen, en $\{\emptyset\}$ heeft er precies één. We hebben:

- $\emptyset \in \{\emptyset\}$.

Verder heeft de lege verzameling de eigenschap dat hij bevat is in elke andere verzameling. Immers: omdat $\emptyset$ geen elementen heeft zal altijd gelden: als A een verzameling is, dan is elk element van $\emptyset$ element van A.

Opdracht 2.4 *Ga na dat* $\emptyset$ echt *bevat is in elke verzameling behalve* $\emptyset$.

Opdracht 2.5 *Wat drukken de volgende beweringen uit? Welke ervan zijn juist en welke niet?*

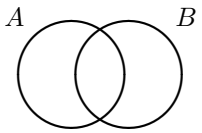
1. $\emptyset \subseteq \{a\}$
2. $\emptyset \subseteq \{\emptyset\}$
3. $\emptyset \in \{\emptyset\}$
4. $\emptyset \in \emptyset$
5. $\emptyset \notin \emptyset$
6. $\emptyset \in \{a\}$.

2.4 Vereniging, doorsnede en verschil

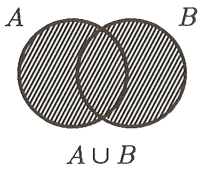
Laat A en B twee verzamelingen zijn. Dan is de *vereniging* (Engels: *union* of *join*) van A en B als volgt gedefinieerd.

Definitie 2.1 *De* **vereniging** *van A en B* $\stackrel{\text{def}}{=}$ *de verzameling van alle elementen uit A plus alle elementen uit B.*

We noteren de vereniging van A en B als $A \cup B$. Het is nuttig om plaatjes (Venn diagrammen) te tekenen. We tekenen een Venn diagram altijd zo algemeen mogelijk:



Nu is $A \cup B$ het gearceerde gebied:



Een paar voorbeelden van het gebruik van $\cup$:

- $\{a, b\} \cup \{a\} = \{a, b\}$
- $\{a, b\} \cup \{a, c\} = \{a, b, c\}$
- $\{a\} \cup \{b\} = \{a, b\}$
- $\{a\} \cup \{\{a\}\} = \{a, \{a\}\}$
- $\{a, b\} \cup \emptyset = \{a, b\}$.

Het verenigen van twee verzamelingen is een *bewerking* die je op verzamelingen uitvoert, en waarvan het resultaat een nieuwe verzameling is. Zo'n bewerking wordt een *operatie* genoemd. De operatie 'verenigen' (aangeduid met het symbool $\cup$) heeft een aantal elementaire eigenschappen die we hier opsommen:

- $A \cup A = A$ (deze eigenschap heet *idempotentie*)
- $A \cup B = B \cup A$ (deze eigenschap heet *commutativiteit*)
- $A \cup (B \cup C) = (A \cup B) \cup C$ (deze eigenschap heet *associativiteit*).

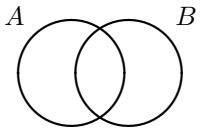
Het bewijs van de idempotentie van $\cup$: een willekeurig ding x zit in $A \cup A$ desda x in een van de twee verenigde verzamelingen zit, dus: desda x in A zit of in A, dat wil zeggen desda x in A zit. Dus $A \cup A = A$.

Opdracht 2.6 *Bewijs op dezelfde manier dat $\cup$ commutatief en associatief is.*

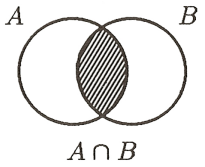
Uit de associativiteit van $\cup$ volgt dat we willekeurig veel verzamelingen kunnen verenigen zonder ons druk te maken om de haakjes (dat wil zeggen om de *volgorde* bij het toepassen van de verenigings-operatie). We kunnen dus in plaats van $A \cup (B \cup C)$ of $(A \cup B) \cup C$ zonder bezwaar schrijven: $A \cup B \cup C$.

Definitie 2.2 *De* **doorsnede** *of* **doorsnijding** *(Engels:* intersection *of* meet*) van twee verzamelingen A en $B \stackrel{\text{def}}{=}$ de verzameling van de dingen die zowel element van A als van B zijn.*

De doorsnijding van A en B wordt aangegeven als: $A \cap B$. Weer is een plaatje nuttig. Laat dit de verzamelingen A en B zijn:



Dan is het gearceerde gebied de verzameling $A \cap B$:



Een paar voorbeelden (neem aan dat a, b en c onderling *verschillende* dingen zijn):

- $\{a\} \cap \{b\} = \emptyset$
- $\{a, b, c\} \cap \{a, b\} = \{a, b\}$
- $\{a, b\} \cap \{b, c\} = \{b\}$
- $\{a\} \cap \{a\} = \{a\}$
- $\{a, b, c\} \cap \emptyset = \emptyset$.

Net als 'verenigen' heeft 'doorsnijden' de volgende eigenschappen:

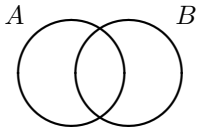
- $A \cap A = A$ (idempotentie)
- $A \cap B = B \cap A$ (commutativiteit)
- $A \cap (B \cap C) = (A \cap B) \cap C$ (associativiteit).

Opdracht 2.7 *Bewijs de idempotentie, commutativiteit en associativiteit van* $\cap$.

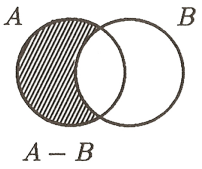
We kunnen nu ook een argument geven waarom een abstract object als $\emptyset$ handig is. Stel dat A en B verzamelingen zijn zonder gemeenschappelijke elementen. Ook in dit geval willen we toch dat $A \cap B$ een verzameling is: dat is nu dus $\emptyset$. De derde elementaire operatie op verzamelingen die we behandelen is het nemen van het *verschil* van twee verzamelingen A en B (Engels: *the difference of* A *and* B).

Definitie 2.3 *Het* **verschil** *van* A *en* $B \stackrel{\text{def}}{=}$ *de verzameling van alle elementen van* A *die niet in* B *zitten.*

Notatie voor het verschil van A en B: $A - B$. We tekenen weer een plaatje. Hier zijn weer de verzamelingen A en B:



$A - B$ is nu het gearceerde gedeelte:

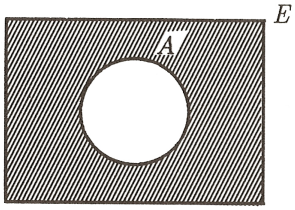


Voorbeelden (neem weer aan dat a, b, en c onderling verschillende dingen zijn):

- $\{a\} - \{b\} = \{a\}$
- $\{a, b, c\} - \{a, b\} = \{c\}$
- $\{a, b\} - \emptyset = \{a, b\}$
- $\{a, \{a\}\} - \{a\} = \{\{a\}\}$
- $\{a, \{a\}\} - \{\{a\}\} = \{a\}$
- $\emptyset - \{a, b, c\} = \emptyset$.

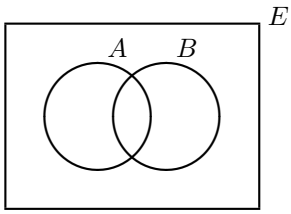
Opdracht 2.8 *Is de verschil-operatie idempotent? Commutatief? Associatief?*

Wanneer A een deelverzameling is van E noemen we het verschil van E en A ook wel: het *complement* van A ten opzichte van E. Het gearceerde gedeelte in het plaatje is het complement van A ten opzichte van E:



Wanneer duidelijk is ten opzichte van *welke verzameling* er complementen worden genomen kunnen we het vermelden van de verzameling E ook achterwege laten. De notatie voor het complement van A wordt nu: A^c. Andere notaties die ook wel worden gebruikt zijn: A' en $\overline{A}$.

Opdracht 2.9 *Beschouw het volgende plaatje:*



Geef de volgende verzamelingen aan door middel van arceren (teken zo nodig nieuwe plaatjes):

1. A^c
2. $(A^c)^c$
3. $A^c \cup B^c$
4. $A^c \cap B^c$
5. $(A \cup B)^c$
6. $(A \cap B)^c$.

De complementen zijn steeds ten opzichte van E.

2.5 Deel- en machtsverzamelingen

De notatie $A \subseteq B$ voor "A is een deelverzameling van B" hebben we hierboven al ingevoerd. "A is een deelverzameling van B" betekent per definitie: elk element van A is een element van B.

Nu is de verzameling van alle deelverzamelingen van een verzameling A zelf weer een verzameling. Deze verzameling noemen we de *machtsverzameling* van A (Engels: *the power set of* A). Notatie: $\mathcal{P}(A)$, of ook wel: $\mathbf{POW}(A)$. De machtsverzameling van een verzameling A is als volgt gedefinieerd:

Definitie 2.4 $\mathcal{P}(A) \stackrel{\text{def}}{=} \{B \mid B \subseteq A\}$.

Merk op dat voor elke verzameling A geldt dat de lege verzameling element is van $\mathcal{P}(A)$. Immers: $\emptyset$ is een deelverzameling van elke verzameling, dus ook van A. Verder geldt voor elke verzameling A dat A een element is van $\mathcal{P}(A)$. De verzameling A zelf is immers ook een deelverzameling van A.

Voorbeelden van machtsverzamelingen (we nemen weer aan dat a, b en c onderling verschillende dingen zijn):

- $\mathcal{P}(\emptyset) = \{\emptyset\}$
- $\mathcal{P}(\{a\}) = \{\emptyset, \{a\}\}$
- $\mathcal{P}(\{a, b\}) = \{\{a, b\}, \{a\}, \{b\}, \emptyset\}$
- $\mathcal{P}(\{a, b, c\}) = \{\{a, b, c\}, \{a, b\}, \{a, c\}, \{b, c\}, \{a\}, \{b\}, \{c\}, \emptyset\}$.

Uit de voorbeelden lezen we af: de machtsverzameling van een verzameling zonder elementen heeft 1 element ; die van een verzameling met 1 element heeft 2 elementen; die van een verzameling met 2 elementen heeft 4 elementen; die van een verzameling met 3 elementen heeft 8 elementen. Algemeen (voor eindige verzamelingen): als een verzameling A n elementen heeft dan heeft $\mathcal{P}(A)$ 2^n elementen. Het bewijs zullen we hier nog niet geven (zie opdracht 5.7).

Opdracht 2.10 *Bewijs dat als* $A \subseteq B$*, dan* $\mathcal{P}(A) \subseteq \mathcal{P}(B)$.

Opdracht 2.11 *Bewijs het omgekeerde van wat u in opdracht 2.10 hebt bewezen: als* $\mathcal{P}(A) \subseteq \mathcal{P}(B)$ *dan* $A \subseteq B$.

We kunnen de operatie van het nemen van de machtsverzameling van een verzameling *herhaald* toepassen. We kunnen dus spreken over $\mathcal{P}(\mathcal{P}(A))$—of voor het leesgemak: $\mathcal{PP}(A)$—enzovoorts.

Voorbeeld (schrikt u vooral niet; het is gewoon een kwestie van domweg de definitie toepassen):

- $\mathcal{PP}(\{\emptyset\}) = \mathcal{P}(\{\emptyset, \{\emptyset\}\}) = \{\{\emptyset, \{\emptyset\}\}, \{\emptyset\}, \{\{\emptyset\}\}, \emptyset\}$.

Opdracht 2.12 *Schrijf de volgende verzamelingen uit:*

1. $\mathcal{PP}(\emptyset)$
2. $\mathcal{PP}(\{a\})$
3. $\mathcal{PPP}(\{a\})$
4. $\mathcal{PP}(\{a, b\})$.

2.6 Geordende paren, rijtjes en producten

Wanneer we twee dingen a en b samenvoegen in een verzameling $\{a, b\}$ dan zijn—zoals we hebben gezien—die twee elementen ten opzichte van elkaar *ongeordend*. Dit wil zeggen: $\{b, a\}$ is dezelfde verzameling als $\{a, b\}$.

Het kan echter voorkomen dat we juist wel geïnteresseerd zijn in de volgorde van a en b. Neem het geval waarin a en b de coördinaten zijn voor een plaatsbepaling op het aardoppervlak, waarbij de eerste coordinaat de lengtegraad aangeeft, en de tweede de breedtegraad. a en b mogen nu niet worden verwisseld, want 15° Oosterlengte, 80° Noorderbreedte geeft een andere plaats aan dan 80° Oosterlengte, 15° Noorderbreedte. Een ander voorbeeld. Neem aan dat a en b schrijftekens zijn waarvan we de links-rechts volgorde willen vastleggen. Weer mogen a en b niet worden verwisseld.

Wanneer we de twee elementen a en b willen samenvoegen tot een paar waarvan de volgorde van belang is, dan spreken we van *het geordend paar* van a en b. De notatie die we invoeren is: $\langle a, b\rangle$. Ronde haken worden ook wel gebruikt; het geordend paar van a en b wordt dan aangegeven als: (a, b). Engels voor 'geordend paar' : *ordered pair*. Als a en b twee verschillende dingen zijn, dan hebben we: $\langle a, b\rangle \neq \langle b, a\rangle$. Verder geldt:

$$\langle a, b\rangle = \langle c, d\rangle \iff a = c \text{ en } b = d.$$

Als we een willekeurig geordend paar hebben kunnen we spreken over het *eerste lid* van het paar en het *tweede lid* van het paar. Het eerste lid van $\langle a, b\rangle$ is a, het tweede lid is b.

We kunnen het begrip 'geordend paar' generaliseren tot 'geordend n-tal' (Engels: *ordered n-tuple*). Een geordend 1-tal $\langle a\rangle$ is niets anders dan het object a. En geordend 2-tal hebben we al gedefinieerd. Een geordend 3-tal is een rijtje $\langle a, b, c\rangle$. Voor geordende drietallen geldt:

$$\langle a, b, c\rangle = \langle e, f, g\rangle \iff a = e \text{ en } b = f \text{ en } c = g.$$

Voor langere rijtjes gaat het net zo.

We zien nu dat $\langle a \rangle \neq \langle a, a \rangle$. Immers: $\langle a \rangle$ is een geordend 1-tal en $\langle a, a \rangle$ een geordend paar. Dus: het meerdere keren opnemen van een en hetzelfde element maakt bij geordende rijtjes wel degelijk verschil.

Met behulp van het begrip 'geordend paar' kunnen we nu een nieuw begrip invoeren. Het *(Cartesisch) product* van twee verzamelingen A en B, notatie $A \times B$, is per definitie de verzameling van alle geordende paren waarvan het eerste element in A zit en het tweede element in B. Anders gezegd:

Definitie 2.5 $A \times B \stackrel{\text{def}}{=} \{\langle a, b \rangle \mid a \in A \text{ en } b \in B\}$.

Het epitheton 'Cartesisch' is een hommage aan de Franse wiskundige en filosoof René Descartes (Cartesius) (1596–1650), die geordende paren gebruikte om punten in een vlak aan te duiden, in de door hem ontwikkelde analytische meetkunde.

Voorbeelden van Cartesische producten (laat a en b twee verschillende dingen zijn):

- $\{a\} \times \{a, b\} = \{\langle a, a \rangle, \langle a, b \rangle\}$
- $\{a, b\} \times \{a, b\} = \{\langle a, a \rangle, \langle b, b \rangle, \langle a, b \rangle, \langle b, a \rangle\}$
- $\{a, b\} \times \{a\} = \{\langle a, a \rangle, \langle b, a \rangle\}$.

Het is duidelijk dat als A en B verschillend zijn, $A \times B \neq B \times A$.

We kunnen ook Cartesische producten nemen van meer dan twee verzamelingen: dit levert verzamelingen geordende 3-tallen, 4-tallen, enzovoorts. Dus:

$$A \times B \times C = \{\langle a, b, c \rangle \mid a \in A, b \in B, c \in C\}.$$

Meestal wordt voor $A \times A$ een speciale notatie gebruikt: A^2. Net zo voor $A \times A \times A$: A^3, enzovoorts. In het algemeen staat A^n dus voor de verzameling van alle geordende n-tallen uit A.

Voorbeelden (a en b zijn weer twee verschillende dingen):

- $\{a, b\}^2 = \{\langle a, a \rangle, \langle b, b \rangle, \langle a, b \rangle, \langle b, a \rangle\}$
- $\{a, b\}^3 =$
  $\{\langle a, a, a \rangle, \langle a, a, b \rangle, \langle a, b, b \rangle, \langle b, b, b \rangle, \langle b, b, a \rangle, \langle b, a, a \rangle, \langle a, b, a \rangle, \langle b, a, b \rangle\}$.

Opdracht 2.13 *Hoeveel elementen heeft* $\{1, 2, 3\}^3$ *?*

2.7 Relaties en hun eigenschappen

Wat relaties zijn weten we allemaal. We weten bij voorbeeld dat 'beminnen' een relatie is die tussen mensen kan bestaan of juist niet bestaan, dat 'eigenaar zijn van' een relatie is die tussen mensen en dingen kan bestaan, en dat 'groter zijn dan' een relatie is die tussen getallen kan bestaan.

Dit zijn allemaal voorbeelden van relaties tussen twee partners. We noemen ze daarom *twee-plaatsige relaties*. Er bestaan ook relaties die drie partners van node hebben. Een voorbeeld is 'geven aan': deze relatie geldt tussen een subject (de gever), een direct object (het gegevene) en een indirect object (de begunstigde). 'Geven aan' is een voorbeeld van—u vermoedde het al—een *drieplaatsige relatie*.

In het algemeen kun je de plaatsen die de partners in een relatie innemen niet straffeloos verwisselen. Uit het feit dat Jan Marietje bemint hoeft immers nog niet te volgen dat Marietje ook Jan bemint. Hieruit zien we dat tweeplaatsige relaties gelden tussen geordende paren, drieplaatsige relaties tussen geordende drietallen, enzovoorts.

Het inzicht dat we de partners in een relatie moeten beschouwen als de leden van een rijtje leidt tot een mooie abstracte kijk op relaties. Daarbij beschouwen we een relatie als een verzameling, en wel—u zag het al aankomen—als een verzameling van rijtjes. We definiëren de zaak als volgt.

Definitie 2.6 *R is een* **tweeplaatsige relatie** *tussen de elementen van A en de elementen van B $\stackrel{\text{def}}{\iff}$ R is een deelverzameling van het Cartesisch product van A en B.*

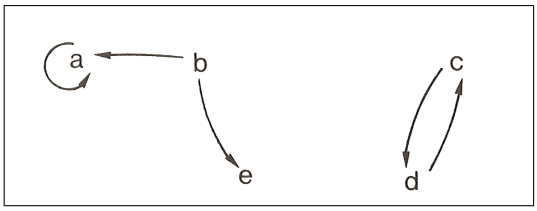
Anders gezegd: $R \subseteq A \times B$. We zeggen ook wel: R is een relatie *binnen* $A \times B$. Net zo voor een drieplaatsige relatie tussen elementen van A, B en C, enzovoorts.

Voorbeeld 2.1 Laat M een verzameling mensen zijn. Dan is de relatie 'beminnen' de verzameling B van geordende paren die bevat is in M^2 en die zodanig is dat $\langle a, b\rangle \in B$ desda persoon a persoon b bemint.

Neem voor het gemak even aan dat M bestaat uit 5 individuen, laten we zeggen: $M = \{a, b, c, d, e\}$. Stel dat a zichzelf bemint, en verder bemint b zowel a als e, en beminnen c en d elkaar. Neem aan dat er verder niet bemind wordt. Dan is de relatie B gelijk aan de volgende verzameling:

$$\{\langle a, a\rangle, \langle b, a\rangle, \langle b, e\rangle, \langle c, d\rangle, \langle d, c\rangle\}.$$

Als altijd is een plaatje nuttig. Wanneer R een relatie is binnen A^2, dan kunnen we elementen van R aangeven door het intekenen van pijlen in een



plaatje van de verzameling A. Een pijl $\longrightarrow$ van x naar y betekent dat $\langle x, y\rangle$ element is van R. Voor ons voorbeeld levert dit het volgende plaatje op:

Hier komen nog een paar definities.

Definitie 2.7 *Als R een tweeplaatsige relatie is binnen $A \times B$, dan is het* **domein** *van R de verzameling van dingen uit A die tot zeker ding in B in relatie R staan.*

Gangbare notatie voor het domein (Engels: *domain*) van R: $\mathrm{dom}(R)$.

Definitie 2.8 *Het* **bereik** *van R is de verzameling van dingen uit B waartoe zeker ding in A in de relatie R staat.*

Gangbare notatie voor het bereik (Engels: *range*) van R: $\mathrm{rng}(R)$. We kunnen een en ander nu ook als volgt formuleren. Als $R \subseteq A \times B$, dan:

- $\mathrm{dom}(R) = \{a \mid a \in A$ en er is een $b \in B$ met $\langle a, b\rangle \in R\}$
- $\mathrm{rng}(R) = \{b \mid b \in B$ en er is een $a \in A$ met $\langle a, b\rangle \in R\}$.

In woorden: $\mathrm{dom}(R)$ is de verzameling *eerste* leden van R, $\mathrm{rng}(R)$ de verzameling *tweede* leden. In het 'beminnen'-voorbeeld hierboven, waar B de relatie

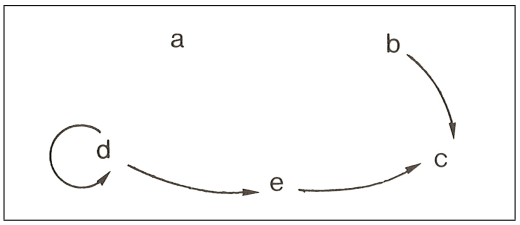
$$\{\langle a, a\rangle, \langle b, a\rangle, \langle b, e\rangle, \langle c, d\rangle, \langle d, c\rangle\}$$

op $M = \{a, b, c, d, e\}$ is, hebben we:

- $\mathrm{dom}(B) = \{a, b, c, d\}$
- $\mathrm{rng}(B) = \{a, c, d, e\}$.

Opdracht 2.14 *Laat R de relatie zijn die met behulp van pijlen is getekend in onderstaand plaatje. Wat zijn* $\mathrm{dom}(R)$ *en* $\mathrm{rng}(R)$*?*

We kunnen nu *eigenschappen* van relaties gaan bestuderen. Dit onderwerp zullen we hier niet uitputtend behandelen; we noemen alleen een paar belangrijke eigenschappen van tweeplaatsige relaties waarvan zowel het domein als het bereik bevat is in een of andere verzameling A. Zulke relaties heten: tweeplaatsige relaties *op* A. Een tweeplaatsige relatie op A is dus een deelverzameling van A^2, ofwel: een relatie *binnen* A^2.



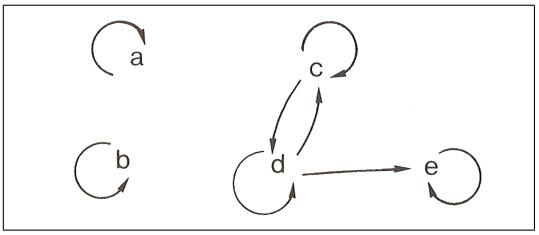
Opdracht 2.15 *Ga na dat 'groter dan' een tweeplaatsige relatie is op* $\mathbf{N}$*, waarbij* $\mathbf{N}$ *de verzameling van de natuurlijke getallen aanduidt, dat wil zeggen* $\mathbf{N} = \{0, 1, 2, 3, \ldots\}$.

Definitie 2.9 *Een relatie* $R \subseteq A^2$ *heet* **reflexief** *wanneer voor elke* $a \in A$ *geldt:* $\langle a, a \rangle \in R$.

Voorbeeld 2.2 De relatie 'waardering hebben voor' tussen de leden van een groep mensen is reflexief wanneer het zo is dat iedereen in die groep zichzelf waardeert (en mogelijkerwijs ook nog anderen).

Voorbeeld 2.3 De relatie 'groter of gelijk zijn aan' tussen natuurlijke getallen is reflexief: elk getal is immers groter dan of gelijk aan zichzelf.

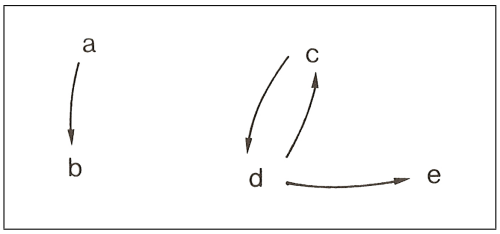
Voorbeeld 2.4 De relatie die met behulp van pijlen is aangegeven in het volgende plaatje is reflexief:



Definitie 2.10 *Een relatie* $R \subseteq A^2$ *heet* **irreflexief** *wanneer voor geen enkele* $a \in A$ *geldt dat* $\langle a, a \rangle \in R$.

Voorbeeld 2.5 De relatie 'groter zijn dan' tussen natuurlijke getallen is irreflexief: geen enkel getal is immers groter dan zichzelf.

Voorbeeld 2.6 De relatie die met behulp van pijlen is getekend in het volgende plaatje is irreflexief:



Opdracht 2.16 *Teken een plaatje van een relatie die noch reflexief, noch irreflexief is.*

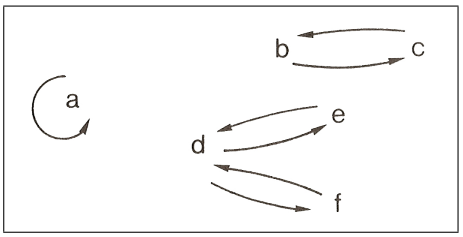
Definitie 2.11 *Een relatie* $R \subseteq A^2$ *is* **symmetrisch** *wanneer voor elk paar* $\langle a, b\rangle \in R$ *geldt:* $\langle b, a\rangle \in R$.

Voorbeeld 2.7 De relatie 'familie zijn van' tussen mensen is symmetrisch.

Voorbeeld 2.8 De relatie 'even oud zijn als' tussen mensen is symmetrisch.

Voorbeeld 2.9 De relatie 'beminnen' is—helaas—vaak *niet* symmetrisch.

Voorbeeld 2.10 In de situatie van het volgende plaatje is de relatie aangegeven door de pijlen symmetrisch:



Definitie 2.12 *Een relatie* $R \subseteq A^2$ *heet* **asymmetrisch** *wanneer voor geen enkel paar* $\langle a, b\rangle \in R$ *geldt dat* $\langle b, a\rangle \in R$.

Voorbeeld 2.11 De relatie 'ouder zijn dan' tussen mensen is asymmetrisch.

Voorbeeld 2.12 De relatie 'ouder zijn van' tussen mensen is asymmetrisch.

Opdracht 2.17 *Laat zien dat uit de definitie van* asymmetrie *volgt dat elke asymmetrische relatie irreflexief is.*

Let op: het niet symmetrisch zijn van een relatie is iets anders dan het asymmetrisch zijn van een relatie. Zie de volgende opdracht.

Opdracht 2.18 *Teken een plaatje van een relatie die noch symmetrisch, noch asymmetrisch is.*

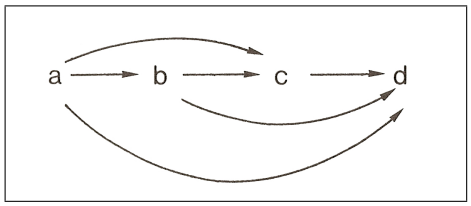
Definitie 2.13 *Een relatie* $R \subseteq A^2$ *heet* **transitief** *wanneer uit het gegeven dat* $\langle a, b\rangle \in R$ *en* $\langle b, c\rangle \in R$ *volgt:* $\langle a, c\rangle \in R$.

Voorbeeld 2.13 De relatie 'ouder zijn dan' tussen mensen is transitief: Als Jan ouder is dan Willem, en Willem is ouder dan Kees, dan is Jan ouder dan Kees.

Voorbeeld 2.14 De relatie 'groter zijn dan' tussen natuurlijke getallen is transitief.

Voorbeeld 2.15 De relatie 'groter dan of gelijk zijn aan' tussen natuurlijke getallen is transitief.

Voorbeeld 2.16 De relatie die met behulp van pijlen is getekend in het volgende plaatje is transitief:



Definitie 2.14 *Een relatie* $R \subseteq A^2$ *heet* **intransitief** *wanneer voor geen enkele* a, b *en* c *in* A *geldt:* $\langle a, b\rangle \in R$, $\langle b, c\rangle \in R$ *en* $\langle a, c\rangle \in R$.

Voorbeeld 2.17 De relatie 'vader zijn van' tussen mensen is intransitief.

Weer geldt: er bestaan relaties die noch transitief, noch intransitief zijn.

Opdracht 2.19 *Teken een plaatje van een relatie die noch transitief, noch intransitief is.*

Definitie 2.15 *Een relatie* $R \subseteq A^2$ *heet* **antisymmetrisch** *wanneer voor elke* a, b *in* A *geldt: als* $\langle a, b\rangle \in R$ *en* $\langle b, a\rangle \in R$, *dan* $a = b$.

Voorbeeld 2.18 De relatie $\leq$ op de verzameling $\mathbf{N}$ van de natuurlijke getallen is antisymmetrisch.

Opdracht 2.20

1. *Laat zien dat een asymmetrische relatie altijd anti-symmetrisch is.*
2. *Laat zien dat het omgekeerde niet geldt. U kunt dit laten zien door een voorbeeld te geven van een anti-symmetrische relatie die niet asymmetrisch is.*

Opdracht 2.21 *Beschouw een willekeurige verzameling A. Merk op dat $\subseteq$ een tweeplaatsige relatie is op $\mathcal{P}(A)$. Maak aan de hand van de hierboven gegeven definities een lijstje van de eigenschappen van deze relatie. Ook $\subset$ is een tweeplaatsige relatie op $\mathcal{P}(A)$. Maak ook voor deze relatie een eigenschappenlijstje.*

Zoals de laatste opdracht illustreert kunnen relaties allerlei combinaties vertonen van eigenschappen.

Definitie 2.16 *Een relatie die transitief, reflexief en antisymmetrisch is wordt een* **partiële orde** *genoemd.*

Definitie 2.17 *Een relatie die transitief, irreflexief en asymmetrisch is heet een* **strikte partiële orde**.

In feite is elke relatie die transitief en irreflexief is tevens asymmetrisch (ga dit na). Net zo is elke relatie die transitief en asymmetrisch is tevens irreflexief (ga na). Hieruit volgt dat het in de definitie van *strikte partiële orde* voldoende is om naast transitiviteit ofwel irreflexiviteit ofwel asymmetrie te eisen.

Opdracht 2.22 *Laat zien dat de relatie $\leq$ op de verzameling $\mathbf{N}$ van de natuurlijke getallen een partiële orde is.*

Opdracht 2.23 *Laat zien dat elke strikte partiële orde bevat is in een partiële orde.*

2.8 Functies

Een functie is een manier om elk van de elementen van een bepaalde verzameling in verband te brengen met een element van een andere verzameling. Voorbeelden van functies kent u uit de krant; de grafiekjes waarin de werkloosheidscijfers vanaf een jaar geleden tot nu maand voor maand zijn af

te lezen zijn functies. Elke maand wordt in verband gebracht met een getal rond de 800.000 dat het aantal geregistreerde werklozen in die maand aangeeft.

Een functie wordt ook wel een *afbeelding* genoemd. We zeggen: een functie beeldt de elementen van een verzameling A af op de elementen van een verzameling B. Een functie die de elementen van A afbeeldt op elementen van B heet *een functie van A naar B*. Functies duiden we vaak aan met de letters f, g en h. "f is een functie van A naar B" wordt vaak als volgt genoteerd:

$$f : A \to B.$$

Wanneer $f : A \to B$ noemen we A het *domein* en B het *codomein* van f.

De dingen uit A waaraan door de functie f dingen uit B worden toegekend heten de *argumenten* of *originelen* van de functie f. De dingen uit B die door f aan dingen uit A worden toegekend heten de *beelden* of *waarden* van f.

We zeggen: *toepassen* van de functie op het argument a geeft als waarde het element b. Notatie voor "f toegepast op a": $f(a)$. Als $f : A \to B$, en f toegepast op $a \in A$ levert $b \in B$ als waarde op hebben we dus: $f(a) = b$.

Om het begrip *functie* te verduidelijken gaan we weer plaatjes tekenen. Hier is een (slechts bij grove benadering juist) plaatje van een werkloosheidsfunctie. Het plaatje laat het aantal ingeschreven werkloze academici per jaar zien, van 1978 tot en met 1987:[1]

1978	$\longrightarrow$	4200
1979	$\longrightarrow$	4800
1980	$\longrightarrow$	5200
1981	$\longrightarrow$	6400
1982	$\longrightarrow$	9000
1983	$\longrightarrow$	11500
1984	$\longrightarrow$	16000
1985	$\longrightarrow$	16400
1986	$\longrightarrow$	16000
1987	$\longrightarrow$	17000

Laten we deze werkloosheidsfunctie even g noemen. Uit het plaatje blijkt dat het domein van g de verzameling is van alle jaren vanaf 1979 tot en met 1987. Het codomein van g is de verzameling van natuurlijke getallen. Uit het plaatje kunnen we bij voorbeeld aflezen welke waarde g toekent aan 1987: $g(1987) = 17000$. Merk op dat niet elk natuurlijk getal hoeft op te treden als waarde. Dat het codomein van de functie de verzameling van

[1] De gegevens zijn afgelezen uit een grafiekje in *Carrière*, 12 december 1987.

natuurlijke getallen is wil alleen zeggen dat elke waarde die de functie kan aannemen een natuurlijk getal is.

Nog een voorbeeld. Als we een verzameling mensen bekijken—noem die verzameling A—dan is "moeder van" een functie. Het is een manier om elk mens in A in verband te brengen met een element van een verzameling mensen B die alle moeders van mensen uit A bevat. Immers: gegeven een bepaalde persoon a, dan is er altijd een persoon te vinden die de moeder is van a. Hier is een voorbeeld-tabel:

Wilhelmina	$\longrightarrow$	Emma
Juliana	$\longrightarrow$	Wilhelmina
Beatrix	$\longrightarrow$	Juliana
Willem-Alexander	$\longrightarrow$	Beatrix

Noem deze functie h. Nu is $h(\text{Juliana}) = \text{Wilhelmina}$. De functie h kent aan argument Juliana de waarde Wilhelmina toe. Algemeen: de functie h kent aan elk element uit de verzameling

$$\{\text{Wilhelmina}, \text{Juliana}, \text{Beatrix}, \text{Willem-Alexander}\}$$

de moeder toe van dat element.

Abstract beschouwd is een functie met domein A en codomein B niets anders dan een bijzondere vorm van een tweeplaatsige relatie binnen $A \times B$. Beschouw het voorbeeld van de koninklijke moeders: de functie uit de tabel kan als volgt worden genoteerd als een verzameling:

$$\{\langle \text{Wilhelmina}, \text{Emma}\rangle, \langle \text{Juliana}, \text{Wilhelmina}\rangle,\\ \langle \text{Beatrix}, \text{Juliana}\rangle, \langle \text{Willem-Alexander}, \text{Beatrix}\rangle\}.$$

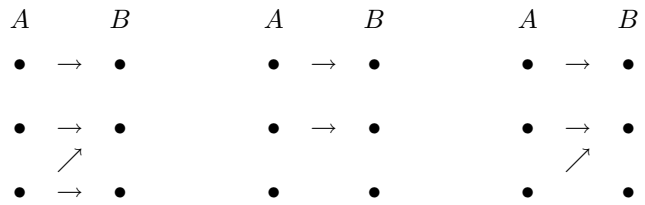
Deze verzameling is een tweeplaatsige relatie binnen het product:

$$\{\text{Wilhelmina}, \text{Juliana}, \text{Beatrix}, \text{Willem-Alexander}\} \times\\ \{\text{Emma}, \text{Wilhelmina}, \text{Juliana}, \text{Beatrix}\}.$$

In het algemeen kunnen we een functie altijd beschouwen als een verzameling paren, waarbij de argumenten van de functie optreden als eerste lid van een paar, en de toegekende waarden als tweede lid. Dus: elke functie $f : A \to B$ is een relatie binnen $A \times B$.

Is nu ook elke relatie R binnen $A \times B$ een functie van A naar B? Nee, dat is niet zo, want het bijzondere aan een functie is, dat je voor elk argument een waarde moet kunnen aflezen. Met andere woorden: de waarde die aan een bepaald argument wordt toegekend moet *uniek zijn*. Verder moet bij

een functie aan *elk* element uit A een element uit B worden toegekend. Bij een relatie hoeft dit niet. Enkele plaatjes ter verduidelijking:



De pijlen in de plaatjes links en midden geven een relatie tussen A en B aan, maar deze relatie is geen functie van A naar B. De reden in het plaatje links: sommige objecten uit A zijn gekoppeld aan meer dan één object in B. De reden in het middelste plaatje: niet alle objecten uit A hebben een waarde. De pijlen in het derde plaatje, tenslotte, geven een relatie binnen $A \times B$ aan die tevens een functie is van A naar B: elk element van A is uniek gekoppeld aan een element van B.

Voorbeeld 2.19 De relatie tussen personen en hun geboorte-datum is een functie.

Voorbeeld 2.20 De relatie tussen personen en hun vaders is een functie.

Voorbeeld 2.21 De relatie tussen personen en hun broer is geen functie, want sommige mensen hebben geen broers, of ze hebben er meer dan één.

Voorbeeld 2.22 De relatie tussen personen en hun jongste zusje is geen functie, want sommige mensen hebben geen zusjes.

Voorbeeld 2.23 De relatie tussen landen en hun hoofdsteden is een functie (want elk land heeft precies één hoofdstad).

Voorbeeld 2.24 De relatie tussen landen en hun hoofdsteden annex residentiesteden is geen functie, want in sommige landen vallen hoofdstad en residentiestad niet samen.

Voorbeeld 2.25 De relatie tussen IBM-personal computers en hun serienummers is een functie, want elke PC heeft een uniek serienummer.

Voorbeeld 2.26 De relatie tussen natuurlijke getallen en hun kwadraten is een functie, want elk natuurlijk getal heeft een uniek kwadraat.

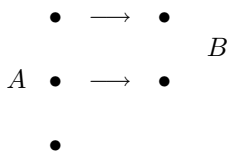
Voorbeeld 2.27 De relatie tussen natuurlijke getallen en hun derdemachtswortels is een functie.

Uit een relatie die geen functie is kan soms simpelweg een functie worden geconstrueerd door de domein-verzameling van de relatie anders te kiezen.

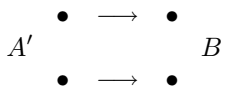
Voorbeeld 2.28 De relatie tussen mannen en hun huidige-echtgenotes in $M \times V$ (waarbij M de verzameling van alle mannen is, en V de verzameling van alle vrouwen) is geen functie, want sommige mannen zijn niet getrouwd. Wanneer we M inperken tot de verzameling M' van getrouwde mannen, dan is diezelfde relatie wel een functie: elke getrouwde man kan uniek worden afgebeeld op zijn huidige wettige echtgenote (we gaan hier even uit van een monogame samenleving).

Een relatie die strikt genomen geen functie is omdat zij niet voor elk argument een waarde oplevert, maar die bij inperking van het domein *wel* een functie is voor dat nieuwe domein wordt wel een *partiële functie* genoemd.

Voorbeeld 2.29 Hier is een voorbeeld van een partiële functie:



Inperking van het domein geeft:



Opdracht 2.24 *Ga na of de volgende relaties tevens functies zijn:*

1. *de relatie kind–ouder;*
2. *de relatie tussen een persoon en de verzameling van zijn/haar ouders.*

Nu kunnen we een precieze verzamelingstheoretische definitie van het begrip 'functie' geven:

Definitie 2.18 *f is een* **functie** *van A naar B* $\stackrel{\text{def}}{\iff}$

- $f \subseteq A \times B$*;*
- *bij elke $a \in A$ is er precies één $b \in B$ zodanig dat $\langle a, b\rangle \in f$.*

Opdracht 2.25 *Ga na of er—volgens deze definitie—functies bestaan met als domein de lege verzameling. Zo nee, waarom niet? Zo ja, welke?*

De *domein-* en *bereik*-definities voor relaties zijn nu ook van toepassing op functies. We hebben dus voor functie $f : A \to B$ het volgende: $\mathrm{dom}(f) = A$ en $\mathrm{rng}(f) \subseteq B$.

De verzameling van alle functies met domein A en codomein B wordt wel aangeduid als B^A. Dus:

$$B^A = \{f \mid f \text{ is een functie van } A \text{ naar } B\}.$$

Opdracht 2.26 *Wat zijn de elementen van* $\{c, d\}^{\{a,b\}}$ *?*

Definitie 2.19 *Als f een functie is van A naar B, en $A' \subseteq A$, dan is $f[A']$,* **het beeld van A' onder f,** *de verzameling*

$$\{b \in B \mid \textit{er is een } a \in A' \textit{ met } f(a) = b\}.$$

Definitie 2.20 *Als f een functie is van A naar B, en $B' \subseteq B$, dan is $f^{-1}[B']$,* **het volledig origineel van B' onder f,** *de verzameling*

$$\{a \in A \mid \textit{er is een } b \in B' \textit{ met } f(a) = b\}.$$

Uit deze definities volgt onmiddellijk dat als $f : A \to B$, dan $f[A] = \mathrm{rng}(f)$ en $f^{-1}[B] = A$.

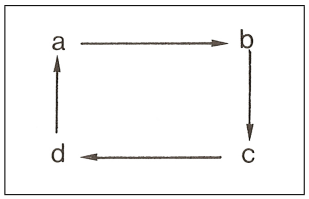
Een functie f kan worden toegepast op een waarde van een functie g, mits de desbetreffende waarde van g maar in het domein van f ligt. In het algemeen: als $g : A \to B$ en $f : B \to C$, dan kunnen we, voor een $a \in A$, eerst a afbeelden op een element van B met behulp van g, en vervolgens dit element van B afbeelden op een element van C met behulp van f. Het element van c waar we zo terecht komen kan worden aangeduid als $f(g(a))$. Deze procedure van eerst g en vervolgens f toepassen levert ons in feite een *nieuwe* functie op, een functie met domein A en codomein C. Deze functie wordt de *compositiefunctie* $f \circ g$ (spreek uit "f na g") genoemd. We kunnen $f(g(a))$ nu ook noteren als $f \circ g(a)$. Wederom: plaatjes kunnen dit verduidelijken:

$$\begin{array}{ccccc} A & \overset{g}{} & B & \overset{f}{} & C \\ a & \to & u & \to & x \\ \\ b & \to & v & \to & y \\ & & & \nearrow & \\ c & \to & w & & \\ & \nearrow & & & \\ d & & & & \end{array} \qquad\qquad \begin{array}{ccc} A & \overset{f\circ g}{} & C \\ a & \to & x \\ \\ b & & \\ & \searrow & \\ c & \to & y \\ & \nearrow & \\ d & & \end{array}$$

Wanneer f een functie is met een en dezelfde verzameling A als domein en als codomein, dan kunnen we de functie op waarden van zichzelf toepassen. Dit heet *functie-iteratie*. Voorbeeld: de moeder van de moeder van de moeder van Willem Alexander. Of: het kwadraat van het kwadraat van 2.

Definitie 2.21 *Een functie met een en dezelfde verzameling A als domein en als codomein wordt een* **eenplaatsige operatie** *op A genoemd.*

Opdracht 2.27 *Beschouw het volgende plaatje.*



De relatie die getekend is geeft voor ieder element van A precies één ding waartoe dat element in relatie staat. Ditzelfde in andere woorden gezegd: de relatie is functioneel. *Noem deze functionele relatie (dat wil zeggen: functie) voor het gemak f.*

1. *Bepaal $f(a)$.*
2. *Bepaal $f(f(a))$.*
3. *Bepaal $f(f(f(a)))$.*
4. *Bepaal $f(f(f(f(a))))$.*
5. *Bepaal $f(b)$.*
6. *Bepaal $f(f(f(f(b))))$.*

Opmerking: $f(f(a))$ kan ook worden geschreven als: $f \circ f(a)$, $f(f(f(a)))$ als $f \circ f \circ f(a)$, enzovoorts.

Opdracht 2.28 *Breid het plaatje uit de vorige opdracht uit met pijlen die de functionele relatie $f \circ f$ aanduiden (gebruik pijlen van de vorm $\Longrightarrow$).*

Merk op dat het domein van een functie f zelf een Cartesisch product kan zijn. In dat geval bestaan de argumenten van f uit rijtjes. Laat $f : A^2 \to B$ zo'n functie zijn. Dan kent f aan elk paar $\langle x, y \rangle \in A^2$ een element van B toe. In plaats van $f(\langle x, y \rangle)$ schrijft men meestal: $f(x, y)$. De functie f kent waarden toe aan tweetallen argumenten.

Een functie $f : A^2 \to A$ wordt een tweeplaatsige operatie op A genoemd; een functie $f : A^3 \to A$ heet een drieplaatsige operatie op A, enzovoorts.

Voorbeeld 2.30 De functie + die aan elk paar van natuurlijke getallen $\langle m, n\rangle$ het getal $m + n$ toekent is een tweeplaatsige operatie op $\mathbf{N}$. Deze operatie wordt in de wandeling 'optellen' genoemd. Merk op dat de optel-operatie *commutatief* is: de volgorde waarin de argumenten worden genomen maakt niet uit.

We besluiten met nog wat notatie. Als $f : \mathbf{N} \to \mathbf{N}$ (dat wil zeggen f is een functie met als domein en als codomein de natuurlijke getallen), en f is de functie die ieder getal afbeeldt op zijn kwadraat (anders gezegd: f is de functie 'kwadrateren'), dan drukken we dit wel uit met behulp van het voorschrift $f(x) = x^2$. De functie $g : \mathbf{N} \to \mathbf{N}$ die elk getal eerst kwadrateert en er dan 1 bij optelt heeft als voorschrift $g(x) = x^2 + 1$. Als h staat voor de functie die twee natuurlijke getallen bij elkaar optelt en het resultaat vervolgens kwadrateert, dan wordt het functie-voorschrift:

$$h(x, y) = (x + y)^2.$$

In feite legt zo'n functie-voorschrift uit wat een functie doet in termen van operaties die al eerder gegeven zijn en waarvan de werking bekend wordt verondersteld, zoals 'optellen' en 'machtsverheffen'.

2.9 Bijzondere functies

Functies die als codomein de verzameling $\{0, 1\}$ hebben noemen we *karakteristieke functies*. Een karakteristieke functie $f : A \to \{0, 1\}$ doet in feite niets anders dan een *deelverzameling* van A karakteriseren, namelijk de deelverzameling van die elementen van A die op 1 worden afgebeeld. Weer kan een tabel helpen:

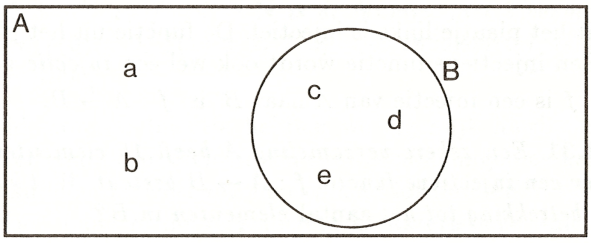
A	f	$\{0, 1\}$
a	$\longrightarrow$	1
b	$\longrightarrow$	0
c	$\longrightarrow$	1
d	$\longrightarrow$	1
e	$\longrightarrow$	0

De functie f uit het plaatje karakteriseert de verzameling $\{a, c, d\} \subset A$.

Opdracht 2.29 *Het volgende plaatje laat een verzameling A met een echte deelverzameling B zien.*

Som de elementen op van de functie $f : A \to \{0, 1\}$ die B karakteriseert.

Er bestaat een één–één correspondentie tussen karakteristieke functies met domein A en deelverzamelingen van A. Het verband is als volgt:



- Met de karakteristieke functie $f : A \to \{0, 1\}$ correspondeert de volgende deelverzameling A' van A:

  $A' = \{a \in A \mid f(a) = 1\}$.

- Als $A' \subseteq A$ dan is de karakteristieke functie $f : A \to \{0, 1\}$ die A' karakteriseert de volgende functie:

  $f = \{\langle a, 1\rangle \mid a \in A'\} \cup \{\langle a, 0\rangle \mid a \in A - A'\}$.

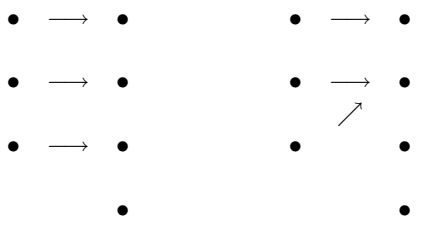
Opdracht 2.30 *Ga na dat* $\{\langle a, 1\rangle \mid a \in A'\} \cup \{\langle a, 0\rangle \mid a \in A - A'\}$ *inderdaad*

1. *een functie is,*
2. *een karakteristieke functie is,*
3. *de functie is die* A' *karakteriseert.*

We geven nu enkele definities van *eigenschappen* die functies kunnen hebben.

Definitie 2.22 *Een functie* f *heet* **injectief** *of* **één–één-duidig** *als* f *aan verschillende elementen uit zijn domein verschillende waarden toekent.*

Iets formeler: als $a \neq b$ dan $f(a) \neq f(b)$. Dit komt op hetzelfde neer als het volgende: als $f(a) = f(b)$ dan $a = b$. In een paar plaatjes:



De functie uit het plaatje links is injectief. De functie uit het plaatje rechts is dat niet. Een injectieve functie wordt ook wel een *injectie* genoemd. De notatie voor 'f is een injectie van A naar B' is: $f : A \xrightarrow{i} B$.

Opdracht 2.31 *Een zekere verzameling A heeft 15 elementen. Verder is gegeven dat er een injectieve functie $f : A \to B$ bestaat. Wat kunt u hieruit afleiden met betrekking tot het* aantal *elementen in B?*

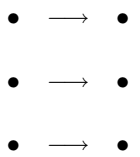
Definitie 2.23 *Een functie f heet* **surjectief** *wanneer elk element uit het codomein van f optreedt als functie-waarde.*

Iets formeler: als f een functie is van A naar B, dan is er voor elke $b \in B$ een $a \in A$ met $b = f(a)$. Dus: een functie $f : A \to B$ is surjectief desda $B = \mathrm{rng}(f)$. We verduidelijken een en ander weer met plaatjes:

De functie uit het plaatje links is surjectief; de functie uit het plaatje rechts is dat niet. Een surjectieve functie wordt wel een *surjectie* genoemd. Notatie voor surjecties: $f : A \xrightarrow{s} B$. Voor een surjectie zegt men ook wel: f is een functie van A *op* B (Engels: *onto*).

Definitie 2.24 *Een functie f heet* **bijectief** *(spreek uit: bi-jectief) wanneer f zowel injectief als surjectief is.*

Een bijectieve functie heet in de wandeling een *bijectie*. Notatie voor bijecties: $f : A \xrightarrow{i,s} B$. Een plaatje van een bijectie:



Als f een bijectie is van A naar B dan brengt f een één–één-duidige correspondentie tot stand tussen de leden van A en de leden van B: bij iedere $a \in A$ hoort precies één $b \in B$ met $b = f(a)$, en bij iedere $b \in B$ hoort precies één $a \in A$ met $b = f(a)$ (ga dit zelf na aan de hand van de eigenschappen van een bijectie).

Opdracht 2.32 $\mathbf{N}$ *is de verzameling van natuurlijke getallen (de verzameling $\{0, 1, 2, 3, ...\}$, en $f : \mathbf{N} \to \mathbf{N}$ is de functie 'met 2 vermenigvuldigen' (dat wil zeggen $f(0) = 0$, $f(1) = 2$, $f(2) = 4$, enzovoorts). Is f surjectief? Injectief? Bijectief?*

Een voorbeeld van een bijectie hebben we eigenlijk hierboven al gezien, toen we het verband bespraken tussen deelverzamelingen en karakteristieke functies. We kunnen dat verband nu expliciet formuleren in termen van het begrip 'bijectie'. Laat K de verzameling karakteristieke functies zijn met domein A, voor een willekeurige A, met andere woorden $K = \{0,1\}^A$. Nu is er een bijectie $g : K \to \mathcal{P}(A)$. Ga na dat g wordt vastgelegd door het voorschrift $g(f) = \{a \mid f(a) = 1\}$ (waarbij f een functie $\in K$ is).

Als f een bijectie is, dan kunnen we de richting van de functie omdraaien door eenvoudigweg elk element $\langle a, b\rangle$ van f te veranderen in $\langle b, a\rangle$. Het resultaat zal weer een functie zijn, en wel: een bijectie. Immers, omdat f bijectief is, is er voor elke waarde van f een uniek argument dat die waarde oplevert. De functie die door zo uit een bijectie f ontstaat door 'omdraaien' heet de *inverse functie* van f, en wordt genoteerd als: f^{-1}. Dus: als $f : A \to B$ een bijectie is, dan is f^{-1} de functie van B naar A die aan elke $b \in B$ het uniek bepaalde element $a \in A$ koppelt waarvoor geldt dat $f(a) = b$.

Voorbeeld 2.31 De functie f uit het linkerplaatje is een bijectie, en dus is ook f^{-1} uit het rechterplaatje een bijectie:

A	f	B		B	f^{-1}	A
a	$\longrightarrow$	w		w	$\longrightarrow$	a
b	$\longrightarrow$	x		x	$\longrightarrow$	b
c	$\longrightarrow$	y		y	$\longrightarrow$	c
d	$\longrightarrow$	z		z	$\longrightarrow$	d

Voorbeeld 2.32 De functie

$$f : \mathbf{N} \to \{x \mid x \in \mathbf{N} \text{ en } x \text{ is even}\}$$

die gedefinieerd wordt door $f(n) = 2n$ is een bijectie. De inverse functie

$$f^{-1} : \{x \mid x \in \mathbf{N} \text{ en } x \text{ is even}\} \to \mathbf{N}$$

is gedefinieerd door $f^{-1}(m) = m/2$.

We zullen in het volgende hoofdstuk zien dat bijecties een grote rol spelen bij het redeneren over oneindige verzamelingen. De meeste verzamelingen die we u hierboven hebben voorgeschoteld waren eindig, maar dat was alleen om didactische redenen. Verzamelingenleer gaat pas echt geestverruimend werken wanneer we de innerlijke blik richten op het oneindige. In hoofdstuk 3 nemen we u mee om een kijkje te nemen in 'Cantor's paradijs' van oneindige verzamelingen.

Hoofdstuk 3

Eindige en oneindige verzamelingen

3.1 Eindigheid en oneindigheid

Intuïtief gesproken is een eindige verzameling een verzameling die de eigenschap heeft dat we de elementen van die verzameling kunnen tellen, en wel zo dat dat telproces op een gegeven ogenblik is afgerond. Let wel: het gaat er hier om dat het telproces *in principe* kan worden afgesloten. Eindige verzamelingen kunnen welhaast onvoorstelbaar groot zijn, zonder dat dit iets toe of af doet aan hun eindigheid.

Volgens het Boeddhisme kost het doorlopen van de cirkel van wedergeboorten, nodig om de Verlichting te bereiken, aeonen van tijd. Vraag aan de Boeddha: hoe lang duurt een aeon? Antwoord: "Stel je een machtige rotsformatie voor van vier mijlen hoog, breed en diep, een volkomen massief blok zonder een barst of scheurtje. Stel dat er aan het eind van iedere eeuw een man zou komen die met een doek van Benares eenmaal langs de rots zou strijken. Die bergrots zou dan eerder zijn weggesleten dan er een aeon voorbij is." Zie [Humphreys 1951].

Bevat een aeon nu oneindig veel seconden? Nee. Wel onvoorstelbaar veel, maar dat is niet hetzelfde. Het aantal seconden in een aeon is eindig, want hoewel een aeon onvoorstelbaar lang duurt, hij is op een gegeven moment verstreken. Dus: er bestaat een natuurlijk getal (zij het astronomisch groot) dat dat aantal seconden van een aeon aangeeft.

Goed, u heeft nog wel even tijd om aan logica en formele taalkunde te besteden voordat u de Verlichting bereikt. Een formele definitie van *eindigheid* en *oneindigheid* maakt gebruik van het begrip *bijectie* (zie § 2.9).

Een bijectie of één–één-correspondentie tussen een verzameling A en een verzameling B is een functie f zo dat bij iedere $a \in A$ precies een $b \in B$ zo dat $f(a) = b$, en bij iedere $b \in B$ precies een $a \in A$ zo dat $b = f(a)$. Door middel van een bijectie kunnen we nu een *willekeurige* eindige verzameling in verband brengen met een *speciaal* soort eindige verzameling. Voor elke verzameling met n elementen is er immers een 1–1-correspondentie met de verzameling $\{0, 1, \ldots, n-1\}$. Een dergelijke verzameling wordt wel een *echt beginstuk* van $\mathbf{N}$ genoemd. Een meer exacte en algemene notatie hiervoor is $\{x \in \mathbf{N} \mid x < n\}$.

Definitie 3.1 *Verzameling A heet* **eindig** *wanneer er een $n \in \mathbf{N}$ te vinden is zo dat er een bijectie bestaat tussen $\{x \in \mathbf{N} \mid x < n\}$ en A.*

Voorbeeld 3.1 $\emptyset$ is eindig. Waarom? Omdat er een bijectie bestaat tussen de verzameling $\{x \in \mathbf{N} \mid x < 0\}$ en de lege verzameling.

Voorbeeld 3.2 De verzameling $\{a, b, c\}$ is eindig. De volgende functie is immers een bijectie tussen $\{x \in \mathbf{N} \mid x < 3\}$ en $\{a, b, c\}$:

$$\begin{array}{ccc} 0 & \longrightarrow & \text{a} \\ 1 & \longrightarrow & \text{b} \\ 2 & \longrightarrow & \text{c} \end{array}$$

Voorbeeld 3.3 De verzameling $\{1, 3, 5, 7, 9\}$ is eindig. De volgende functie is een bijectie tussen de verzameling $\{x \in \mathbf{N} \mid x < 5\}$ en deze verzameling:

$$\begin{array}{ccc} 0 & \longrightarrow & 1 \\ 1 & \longrightarrow & 3 \\ 2 & \longrightarrow & 5 \\ 3 & \longrightarrow & 7 \\ 4 & \longrightarrow & 9 \end{array}$$

Het zal duidelijk zijn dat het aanbrengen van een bijectie tussen een verzameling A en een beginstuk van de natuurlijke getallen niets anders is dan wat wij in het alledaagse taalgebruik ‘tellen’ noemen.

De definitie van *oneindige verzameling* is nu simpel:

Definitie 3.2 *Verzameling A heet* **oneindig** *wanneer A niet eindig is.*

Het is gemakkelijk in te zien dat de verzameling van natuurlijke getallen $\mathbf{N}$ oneindig is volgens deze definitie. Ook de verzameling van alle zinnen van het Nederlands is oneindig. Datzelfde geldt voor de verzameling van alle zinnen van het Nederlands die met ‘Ik’ beginnen.

Opdracht 3.1 *Wat is er mis met de volgende redenering: "Als je een willekeurige Nederlandse bijzin 'B' neemt, en je zet daar 'dat Jan dacht dat' voor, dan krijg je een nieuwe Nederlandse (bij)zin. Hieraan kun je zien dat er Nederlandse zinnen zijn die je met behulp van dit recept altijd weer langer kunt maken. Daaruit volgt dat er in het Nederlands oneindig lange zinnen bestaan."*

Als twee *eindige* verzamelingen even groot zijn, kunnen we dat ook uitleggen in termen van bijecties. Twee eindige verzamelingen A en B zijn *even groot* wanneer er een getal n is zo dat er een bijectie is van $\{x \in \mathbf{N} \mid x < n\}$ naar A en een bijectie van $\{x \in \mathbf{N} \mid x < n\}$ naar B. Maar dan is er ook een directe bijectie van A naar B (ga na).

Is het nu zo dat alle oneindige verzamelingen even groot zijn? Het ligt voor de hand om de bijectie-uitleg van 'even groot zijn' ook op oneindige verzamelingen te gaan toepassen. Daarbij stappen we—om de associatie met *eindigheid* te vermijden—over op een andere terminologie. In plaats van over *even groot als* zullen we het nu hebben over *gelijkmachtig met*. Gelijkmachtigheid is een begrip dat zowel op eindige als op oneindige verzamelingen van toepassing is. Hier is de definitie van *gelijkmachtigheid* (Engels: equipollence):

Definitie 3.3 *Verzameling A heet* **gelijkmachtig met** *verzameling B (notatie: $A =_1 B$) wanneer er een bijectie van A naar B bestaat.*

De definities van 'eindig' en 'oneindig' waren nogal prozaïsch. Het begrip *gelijkmachtigheid* maakt echter een meer opwindende blik op het oneindige mogelijk. De definitie van 'gelijkmachtigheid' heeft namelijk als merkwaardig gevolg dat bij voorbeeld de verzameling $\mathbf{N}$ en de verzameling $\mathbf{N} - \{0\}$ gelijkmachtig ('even groot') zijn. Immers, de functie $f : \mathbf{N} \to \mathbf{N} - \{0\}$ gedefinieerd door $f(n) = n + 1$ is een bijectie (zie het plaatje).

$$\begin{array}{ccc} 0 & \longrightarrow & 1 \\ 1 & \longrightarrow & 2 \\ 2 & \longrightarrow & 3 \\ 3 & \longrightarrow & 4 \\ 4 & \longrightarrow & 5 \\ & \vdots & \end{array}$$

We zien aan dit voorbeeld dat de oneindige verzameling $\mathbf{N}$ een echte deelverzameling heeft van dezelfde machtigheid. Het kan worden bewezen dat *elke* oneindige verzameling echte deelverzamelingen heeft van dezelfde machtigheid.

Aan de andere kant zal het u niet lukken een eindige verzameling te vinden die gelijkmachtig is met een van zijn echte deelverzamelingen. Dat dit geen toeval is blijkt uit de volgende stelling.

Stelling 3.1 *Als A een eindige verzameling is, dan is er geen echte deelverzameling van A waarmee A gelijkmachtig is.*

Bewijs: We gebruiken inductie naar het aantal elementen van A.

- Basisstap. A heeft 0 elementen. In dit geval is A gelijk aan $\emptyset$, en $\emptyset$ heeft geen echte deelverzamelingen. De stelling is dus waar voor het basisgeval.

- Inductiestap. De inductie-hypothese luidt: als A hoogstens n elementen heeft, dan is er geen echte deelverzameling van A waarmee A gelijkmachtig is. Veronderstel dat A een verzameling is met $n+1$ elementen.
  We doen een *reductio ad absurdum*. In een *reductio ad absurdum*, ook wel een bewijs uit het ongerijmde genoemd, veronderstellen we datgene wat we willen weerleggen, en leiden we vervolgens uit die veronderstelling een contradictie af. Daarmee is de veronderstelling weerlegd.
  Stel dat B een echte deelverzameling van A is waarmee A gelijkmachtig is. Dit wil zeggen: $B \subset A$, er is minstens één element a van A dat niet in B zit, en er is een bijectie f van A naar B. Noem het element van B waarop a door f wordt afgebeeld b. Beschouw nu de functie van B naar B die ontstaat door het domein van de functie f te beperken tot B. De gebruikelijke notatie hiervoor is $f\lceil B$. De functie $f\lceil B$ is een injectie van B naar B. Het bereik van deze functie is de verzameling $f[B] = \{f(x) \mid x \in B\}$. Deze bereik-verzameling is uiteraard een deelverzameling van B, maar het kan geen *echte* deelverzameling zijn van B. Dat zou in strijd zijn met de inductie-hypothese, die op B van toepassing is omdat B hoogstens n elementen heeft. Dus $f[B] = B$. Hieruit volgt dat $b = f(a')$ voor zeker element a' van B. Omdat f een bijectie is moet a' gelijk zijn aan a; dus: $a \in B$. Hiermee zijn we in tegenspraak geraakt met de veronderstelling die we over a hadden gemaakt. ∎

Opmerking tussendoor: hier en in het vervolg zullen we een bewijs steeds afsluiten met het symbool ∎.

Omdat elke oneindige verzameling gelijkmachtig is met echte deelverzamelingen van zichzelf, terwijl geen enkele eindige verzameling dat is, kunnen we—als we dat willen—*oneindige verzameling* definiëren als: verzameling die gelijkmachtig is met een van zijn echte deelverzamelingen.

3.2 Aftelbaar en overaftelbaar

De definitie van *gelijkmachtigheid* uit § 3.1 had als merkwaardig gevolg dat de verzameling $\mathbf{N}$ en de verzameling $\mathbf{N} - \{0\}$ gelijkmachtig ('even groot') zijn.

Het kan nog gekker: de verzameling van alle natuurlijke getallen $\mathbf{N}$ is 'even groot' als de verzameling $\mathcal{O}$ van de oneven natuurlijke getallen. Immers, $f : \mathbf{N} \to \mathcal{O}$, gedefinieerd door $f(n) = 2n + 1$, is een bijectie. Zie het plaatje:

$$\begin{array}{ccc} 0 & \longrightarrow & 1 \\ 1 & \longrightarrow & 3 \\ 2 & \longrightarrow & 5 \\ 3 & \longrightarrow & 7 \\ 4 & \longrightarrow & 9 \\ & \vdots & \end{array}$$

Net zo zijn er bijecties te vinden tussen de verzameling van natuurlijke getallen en de verzameling van even natuurlijke getallen, tussen de verzameling van even natuurlijke getallen en de verzameling van oneven natuurlijke getallen, tussen de verzameling van natuurlijke getallen en de verzameling van priemgetallen (getallen die alleen deelbaar zijn door zichzelf en door 1), enzovoorts.

Alle oneindige verzamelingen die we tot nu toe gezien hebben waren gelijkmachtig met $\mathbf{N}$. Voor dit soort oneindigheid voeren we een apart begrip in:

Definitie 3.4 *Een verzameling die gelijkmachtig is met* $\mathbf{N}$ *heet* **aftelbaar** *(Engels: denumerable, countably infinite).*

Definitie 3.5 *Wanneer A aftelbaar is en f is een bijectie tussen A en* $\mathbf{N}$*, dan noemen we f een* **aftelling** *van A.*

We geven nog een aantal voorbeelden van verzamelingen die gelijkmachtig zijn met $\mathbf{N}$.

Voorbeeld 3.4 Dat de verzameling $\mathbf{Z}$ van de *gehele getallen* (positieve getallen, negatieve getallen, en het getal 0)

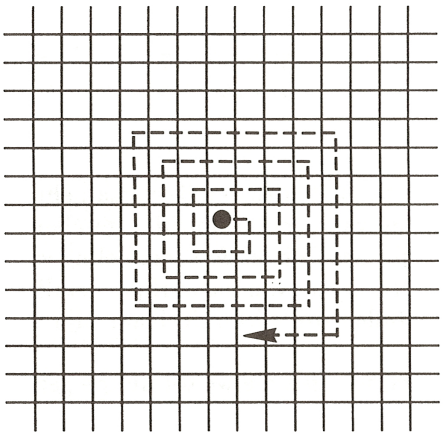
$$\ldots -4, -3, -2, -1,\ 0,\ 1,\ 2,\ 3,\ 4,\ \ldots$$

gelijkmachtig is met $\mathbf{N}$ volgt uit het bestaan van de volgende aftelling:

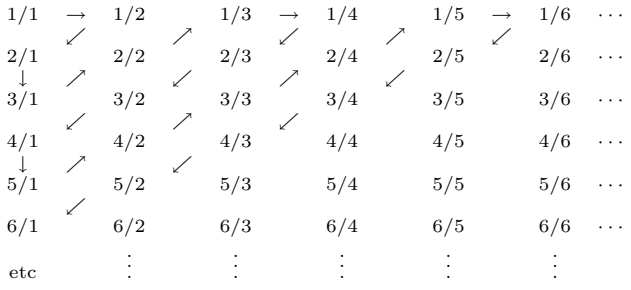
$$
\begin{array}{ccc}
0 & \longrightarrow & 0 \\
1 & \longrightarrow & 1 \\
2 & \longrightarrow & -1 \\
3 & \longrightarrow & 2 \\
4 & \longrightarrow & -2 \\
 & \vdots &
\end{array}
$$

Opdracht 3.2 *Geef een formele definitie van bijectie uit voorbeeld 3.4.*

Voorbeeld 3.5 Is de verzameling van alle velden van een oneindig schaakbord aftelbaar? Ja, kijk maar:



Voorbeeld 3.6 Moeilijker: is de verzameling van de positieve breuken aftelbaar? Het lijkt op het eerste gezicht van niet: tussen elk tweetal natuurlijke getallen liggen immers oneindig veel breuken. Cantor toonde echter aan dat de positieve breuken aftelbaar zijn, door de volgende fraaie aftelinstructie te geven:



Dit is nog niet helemaal een bijectie, want bepaalde getallen komen meerdere keren voor, telkens in een andere gedaante, bij voorbeeld 1/1, 2/2, 3/3, enzovoorts. Sla ze na de eerste keer gewoon over, en je hebt een bijectie.

Opdracht 3.3 *Laat zien dat de verzameling van* **alle** *breuken (positieve breuken, negatieve breuken, en het getal 0) aftelbaar is.*

Voorbeeld 3.7 De verzameling van alle eindige rijtjes letters uit het alfabet is aftelbaar. Immers, deze verzameling kan als volgt worden gerangschikt: eerst de rijtjes van lengte 1 in alfabetische volgorde (dat zijn dus gewoon de letters van het alfabet: $a, b, c, \dots$), dan de rijtjes van lengte twee in alfabetische volgorde ($aa, ab, ac, \dots$), dan de rijtjes van lengte drie in alfabetische volgorde, enzovoort. Deze rangschikking is een aftelling. Merk op dat de grootte van dit (eindige) alfabet er niet toe doet: met 1000 in plaats van 26 letters kunnen we dezelfde redenering gebruiken. Zelfs kunnen we dit resultaat uitbreiden tot de verzameling rijtjes over een *aftelbaar* 'alfabet' A: zie opdracht 3.4.

Opdracht 3.4 *Bewijs dat de verzameling van alle eindige rijtjes symbolen uit een aftelbaar alfabet weer aftelbaar is. (Waarschuwing: U moet nu een andere aftelling maken dan in voorbeeld 3.7 hierboven).*

Opdracht 3.5 *Ergens op een mooi plekje staat een uitzonderlijk groot hotel, een hotel met aftelbaar veel kamers: het Hilbert Hotel. Het hotel is genoemd naar de Duitse logicus en wiskundige David Hilbert. Op zekere dag zijn alle kamers bezet; er zijn dus aftelbaar veel gasten ondergebracht. Dan meldt zich nog iemand bij de receptie. De manager krabt zich even achter het oor, en bedenkt dan een manier om deze extra gast ook nog onder te brengen. Wat moet er gebeuren? (Hint: oneindig veel gasten die al zijn ondergebracht moeten verhuizen.)*

Opdracht 3.6 *We zijn nog steeds bij het Hilbert Hotel, dat helemaal is volgeboekt. Er komt een bus voorrijden; geen gewone bus maar een Hilbert bus: er zitten aftelbaar veel passagiers in. Ook die worden allemaal ondergebracht. Hoe gebeurt dat?*

Opdracht 3.7 *Juist als de portier van het Hilbert Hotel de deur op het nachtslot wil doen (er liggen aftelbaar veel gasten te ronken in aftelbaar veel kamers) komen aftelbaar veel Hilbert bussen voorrijden (elk met ... juist ja). IJlings wordt de manager gewekt. Valt hier nog iets aan te doen? Na enig heen en weer gepraat blijkt dat het Hilbert Hotel groot genoeg is om ook al deze gasten nog onder te brengen. Wat moet er gebeuren?*

Tot nu toe hebben we gelijkmachtigheid van verzamelingen steeds aangetoond door het geven van een bijectief verband tussen die verzamelingen. In een aantal gevallen is het evenwel helemaal niet zo makkelijk die bijectie expliciet te geven. We volstonden bijvoorbeeld met het globaal aanduiden van

een aftelling van $\mathbf{N} \times \mathbf{N}$ (of equivalent: de verzameling positieve breuken); het is waarachtig niet eenvoudig deze slingerende aftelling in een formule te omschrijven! Toch kunnen we dit soort resultaten zonder wiskundige hoogstandjes bewijzen. We moeten dan echter met *injecties* werken in plaats van met bijecties.

Voor eindige verzamelingen geldt dat als A kleiner is dan B, er een injectie van A naar B bestaat, en ook omgekeerd, als er zo'n injectie is, dan is A hoogstens even groot als B. De volgende definitie ligt nu voor de hand:

Definitie 3.6 *We zeggen dat verzameling A* **kleinere of gelijke machtigheid heeft** *dan verzameling B (notatie $A \leq_1 B$) wanneer er een injectie bestaat van A naar B.*

De nieuw ingevoerde notatie suggereert een nuttig verband tussen de verschillende relaties, namelijk:

$$\text{als } A \leq_1 B \text{ en } B \leq_1 A, \text{ dan } A =_1 B.$$

Om dit feit, want dat is het, te kunnen bewijzen, is het handig om ook een oneindig aantal verzamelingen te kunnen manipuleren. We generaliseren daarom de operaties *verenigen* en *doorsnijden*. Stel bij voorbeeld dat we de volgende oneindige verzameling van verzamelingen hebben:

$$A = \{A_0, A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8, \dots\}.$$

Elk van de A_i kan zelf zowel eindig als oneindig zijn, dat doet er nu even niet toe. We definiëren nu de vereniging van A als volgt:

Definitie 3.7 $\bigcup A = \{x \mid \textit{er is een } Y : Y \in A \textit{ en } x \in Y\}$.

Het is niet al te moeilijk in te zien dat dit voor het voorbeeld dat we net hadden neerkomt op het volgende:

$$\bigcup A = A_0 \cup A_1 \cup A_2 \cup A_3 \cup A_4 \cup A_5 \cup A_6 \cup A_7 \cup A_8 \cup \dots$$

Hiermee kunnen we willekeurig grote collecties van verzamelingen verenigen tot een nieuwe verzameling. Net zo definiëren we de doorsnede van A:

Definitie 3.8 $\bigcap A = \{x \mid \textit{voor alle } Y : \textit{ als } Y \in A \textit{ dan } x \in Y\}$.

Voor het voorbeeld komt dit neer op het volgende:

$$\bigcap A = A_0 \cap A_1 \cap A_2 \cap A_3 \cap A_4 \cap A_5 \cap A_6 \cap A_7 \cap A_8 \cap \dots$$

Opdracht 3.8 *Schrijf de volgende verzamelingen uit:*

1. $\bigcup \emptyset$

2. $\bigcup\{\emptyset\}$

3. $\bigcup\{\emptyset, \{\emptyset\}\}$

4. $\bigcup\{\{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$

5. $\bigcap\{\{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$

6. $\bigcup\{\emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\}\}$.

Wanneer we een rij van verzamelingen $A_0, A_1, A_2, A_3, \ldots$ hebben, dan noteren we de vereniging van de rij ook wel als $\bigcup_n A_n$, en de doorsnede als $\bigcap_n A_n$.

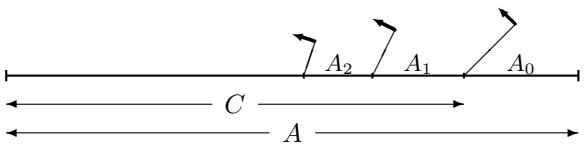
Met dit geschut kunnen we het eerder genoemde feit over de relatie van $A \leq_1 B$ en $A =_1 B$ bewijzen. We herformuleren dit eerst als:

Stelling 3.2 (Cantor-Bernstein) *Als er een injectie bestaat van A naar B, en er bestaat een injectie van B naar A, dan bestaat er een bijectie van A naar B.*

Bewijs: Laat f de injectie van A naar B zijn, en g de injectie van B naar A. Het beeld van B onder g noemen we C. Het is duidelijk dat $C \subseteq A$ en dat g een bijectie van B naar C is. Als we nu kunnen bewijzen dat er een bijectie van A naar C bestaat, dan zijn we klaar. (Waarom? Teken een plaatje.)

Merk op dat we al wel weten dat $g \circ f$ een *injectie* is van A naar C. We zullen nu laten zien dat het mogelijk is uit een injectie tussen A en zijn deelverzameling C een bijectie te fabriceren. De techniek die we daarbij gebruiken is 'naar binnen vouwen van de randjes', en dat een oneindig aantal malen. We noemen $g \circ f$ voor het gemak: j.

Nu is $C \subseteq A$ en $j : A \xrightarrow{i} C$. Laat $A - C = A_0$. We vouwen het randje A_0 naar binnen, door het beeld van A_0 onder j te nemen. j was een injectie van A naar C, dus het beeld van dit randje valt binnen C. Noem het beeld van het randje A_1. Dit nieuwe randje vouwen we opnieuw naar binnen, door er j op los te laten. Het volgende randje heet A_2, enzovoort. In een plaatje zien de eerste drie vouwen naar binnen er als volgt uit:



Algemeen: als we randje nummer n hebben, dat wil zeggen A_n, dan is dit het recept voor verder naar binnen vouwen: $A_{n+1} = j[A_n]$.Tenslotte vegen we alle randjes in één verzameling bij elkaar: $\bigcup_n A_n$.

Deze randjesverzameling is een deelverzameling van C. Wanneer u dacht dat de randjesverzameling 'te groot' zou worden door de oneindig doorgaande vouwpartij, dan laat u zich weer misleiden door de intuïtie van eindigheid: ofwel C is eindig en dan $C = A$ dus $A_n = \emptyset$ (en er is altijd ruimte om *niks* om te vouwen) ofwel C is oneindig groot en dan past de oneindige randjesverzameling er in: j maakt steeds kortere randjes. De bijectie h tussen A en C kan nu als volgt worden gedefinieerd (voor willekeurige $a \in A$):

$$h(a) = \begin{cases} a & \text{als } a \notin \bigcup_n A_n \\ j(a) & \text{als } a \in \bigcup_n A_n. \end{cases}$$

We moeten nu nog laten zien dat $h : A \to C$ een bijectie is:

- Ten eerste: h is inderdaad een *functie* van A naar C. Immers de functiewaarde ligt eenduidig vast en $h(a) \in C$ voor beide gevallen van het functievoorschrift.
- Ten tweede: h is injectief. Met andere woorden: als $a \neq b$ dan $h(a) \neq h(b)$. Neem aan dat $a \neq b$. Drie mogelijkheden:
  1. $a \in \bigcup_n A_n$ en $b \in \bigcup_n A_n$. Nu is $h(a) = j(a)$ en $h(b) = j(b)$. Maar $a \neq b$, en j is een injectie, dus: $j(a) \neq j(b)$.
  2. $a \notin \bigcup_n A_n$ en $b \notin \bigcup_n A_n$. Nu is $h(a) = a$ en $h(b) = b$, en omdat $a \neq b$ hebben we: $h(a) \neq h(b)$.
  3. $a \in \bigcup_n A_n$ en $b \notin \bigcup_n A_n$. Nu is $h(a) = j(a) \in \bigcup_n A_n$ en $h(b) = b \notin \bigcup_n A_n$. Dus $h(a) \neq h(b)$.
  4. $a \notin \bigcup_n A_n$ en $b \in \bigcup_n A_n$. Als in 3.
- Ten derde: h is surjectief, ofwel: voor elke $c \in C$ is er een $a \in A$ zo dat $c = h(a)$. Stel $c \in C$. Dan $c \notin A_0$. Twee mogelijkheden:
  1. $c \in \bigcup_n A_n$, laten we zeggen $c \in A_m$, voor zekere $m \geq 1$ (want c zat niet in A_0). Dus is er een $d \in A_{m-1}$ zo dat $c = j(d) = h(d)$.
  2. $c \notin \bigcup_n A_n$. Nu is $c = h(c)$.

Merk nu tenslotte op dat $g^{-1} \circ h$ een bijectie is van A naar B. ∎

Opdracht 3.9 *In het voorafgaande bewijs werd de indruk gewekt dat de "randjes" steeds verschillend waren, dat wil zeggen dat $A_n \cap A_m = \emptyset$ als $n \neq m$. Bewijs dat dit inderdaad zo is. Gebruik dat voor elke injectie i geldt dat $i[X \cap Y] = i[X] \cap i[Y]$.*

Voorbeeld 3.8 Een eenvoudige toepassing van de stelling van Cantor-Bernstein zien we bij de aftelbaarheid van $\mathbf{Q}$ (de verzameling van alle breuken). Omdat $\mathbf{N} \subseteq \mathbf{Q}$, is er uiteraard een injectie van $\mathbf{N}$ naar $\mathbf{Q}$: de identieke functie f zodat $f(n) = n$ voor elke n. En voor de injectie g de andere kant op nemen we voor elke uitgedeelde breuk $\frac{m}{n}$ met $m, n \in \mathbf{N}$ de waarde

$$g(\frac{m}{n}) = 2^m 5^n$$

en voor negatieve breuken

$$g(-\frac{m}{n}) = 3^m 5^n.$$

Controleren dat dit injecties zijn en ... klaar.

We hebben tot nu toe gebruik gemaakt van bijecties en injecties om iets te weten te komen over de machtigheid van verzamelingen. De volgende stelling laat zien dat ook surjecties gebruikt kunnen worden om de grootte van verzamelingen te vergelijken.

Stelling 3.3 *Als $f : \mathbf{N} \to A$ een surjectie is, dan is A hoogstens aftelbaar.*

Bewijs: Er zijn twee mogelijkheden: ofwel de verzameling

$$B = \{n \in \mathbf{N} \mid \text{ er is geen } k \text{ in } \mathbf{N}, k < n, \text{ met } f(k) = f(n)\}$$

is eindig, of die verzameling is aftelbaar. In het eerste geval zitten er eindig veel elementen in A, in het tweede geval zijn dat er aftelbaar veel. Immers:

$$g = \{\langle n, f(n)\rangle \mid n \in B\}$$

is een bijectie van B naar A. ∎

Opdracht 3.10 *Laat zien: als A aftelbaar is en B eindig, dan zijn $A \cup B$ en $A - B$ aftelbaar.*

Opdracht 3.11 *Laat zien: als A en B beide aftelbaar zijn, dan is $A \cup B$ aftelbaar.*

Opdracht 3.12 *Laat zien: als A en B aftelbaar zijn, dan is $A \times B$ aftelbaar.*

Het begint er een beetje op te lijken dat alle oneindige verzamelingen gelijkmachtig zijn met $\mathbf{N}$. Dat dat *niet* zo is, is door Cantor aangetoond. Cantor liet zien dat de machtsverzameling van $\mathbf{N}$ niet gelijkmachtig is met $\mathbf{N}$.

Allereerst herinneren we eraan dat in § 2.9 in essentie al bewezen is dat $\mathcal{P}(\mathbf{N}) =_1 \{0,1\}^{\mathbf{N}}$; er was immers een 1–1-correspondentie tussen deelverzamelingen en karakteristieke functies. We redeneren daarom verder met $\{0,1\}^{\mathbf{N}}$ in plaats van $\mathcal{P}(\mathbf{N})$. Dat $\{0,1\}^{\mathbf{N}}$ *minstens* even groot is als $\mathbf{N}$ is gemakkelijk in te zien (zie volgende opdracht).

Opdracht 3.13 *Laat zien dat* $\{0,1\}^{\mathbf{N}}$ *minstens even groot is als* $\mathbf{N}$*. U kunt dit doen door het aangeven van een* injectie *van* $\mathbf{N}$ *naar* $\{0,1\}^{\mathbf{N}}$*.*

Cantor bewees dat $\{0,1\}^{\mathbf{N}}$ wezenlijk groter is dan $\mathbf{N}$, met behulp van zijn beroemde diagonaal-argument.

Stelling 3.4 (Diagonaalstelling)

De verzameling $\{0,1\}^{\mathbf{N}}$ *is niet aftelbaar.*

Bewijs: Stel dat de verzameling $\{0,1\}^{\mathbf{N}}$ wél aftelbaar zou zijn. Dan zouden we een aftelling $f_0, f_1, f_2, f_3, \ldots$ hebben van karakteristieke functies. We leiden een tegenspraak af door het construeren van een karakteristieke functie f^* die *niet* in de aftelling voorkomt.

De functies $f_0, f_1, f_2, \ldots$ uit de aftelling waarvan we het bestaan veronderstellen kunnen als volgt worden gerangschikt (dit is natuurlijk maar een voorbeeld; de functies zouden er ook anders kunnen uitzien):

	0	1	2	3	4	5	6	$\cdots$
f_0	1	0	0	0	0	0	0	$\cdots$
f_1	0	1	0	1	0	0	1	$\cdots$
f_2	1	0	0	1	1	0	0	$\cdots$
f_3	0	0	0	0	1	1	0	$\cdots$
f_4	1	0	0	0	0	1	1	$\cdots$
f_5	1	0	0	0	0	1	0	$\cdots$
f_6	1	0	0	0	0	0	1	$\cdots$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	

Beschouw nu de waarheidswaarde-toekenningen op de oneindige diagonaal van dit plaatje. De functie f^* die je krijgt door deze waarheidswaarden *om te keren* (in het voorbeeld: de functie f^* met $f^*(0) = 0, f^*(1) = 0, f^*(2) = 1, f^*(3) = 1, f^*(4) = 1, f^*(5) = 0$, enzovoort) is *ongelijk* aan elke functie in de aftelling. Immers, hij verschilt van f_i in de waarde die aan het i-de argument wordt toegekend. Hiermee is de veronderstelling dat $\{0,1\}^{\mathbf{N}}$ kan worden afgeteld weerlegd. ∎

Het is hopelijk duidelijk waarom de redeneerwijze uit deze stelling 'Cantor's diagonaal-argument' wordt genoemd. Het diagonaal-argument kan ook worden gebruikt om te laten zien dat de verzameling van de reële getallen (gebruikelijke aanduiding: $\mathbf{R}$) niet aftelbaar is. De reële getallen zijn alle breuken plus alle irrationale getallen. Als u weet dat zo'n reëel getal kan worden geschreven als een decimale breuk die achter de komma oneindig doorloopt, kunt u zelf bedenken hoe het diagonaal-argument hier

moet worden gehanteerd. Dat $\mathbf{R}$ niet aftelbaar is, is overigens geen toeval: $\{0,1\}^{\mathbf{N}}$ is gelijkmachtig met de verzameling $\mathbf{R}$ der reële getallen (zie [Van Dalen e.a. 1975] voor het bewijs).

Cantor's diagonaalstelling levert ons een eerste voorbeeld van een verzameling die oneindig is, maar niet aftelbaar. Hiervoor voeren we nieuw jargon in:

Definitie 3.9 *Een oneindige verzameling die niet aftelbaar is heet* **overaftelbaar** *(Engels: non-denumerable, uncountable).*

Opdracht 3.14 *Laat zien dat de verzameling van alle* **eindige** *deelverzamelingen van* $\mathbf{N}$ *aftelbaar is.*

Opdracht 3.15 *Een* **co-finiete** *deelverzameling van* $\mathbf{N}$ *is een deelverzameling* A *van* $\mathbf{N}$ *met de eigenschap dat* $\mathbf{N} - A$ *eindig is (dus: een verzameling met een eindig complement). Laat zien dat de verzameling van alle* co-finiete *deelverzamelingen van* $\mathbf{N}$ *aftelbaar is.*

Cantor's diagonaalstelling is in feite een speciaal geval van een algemenere stelling, die ook door Cantor is bewezen. Deze stelling voert ons via steeds hogere machtigheden binnen in een wiskundig paradijs van steeds grotere oneindige verzamelingen (Cantor's paradijs). We moeten dan nog eerst dit begrip 'groter' precies maken:

Definitie 3.10 *Verzameling* A *heeft* **kleinere machtigheid** *dan verzameling* B *(notatie:* $A <_1 B$*) wanneer* $A \leq_1 B$ *en* $A \neq_1 B$*.*

Stelling 3.5 (Algemene diagonaalstelling)

Voor elke verzameling A*:* A *heeft een kleinere machtigheid dan* $\mathcal{P}(A)$*.*

Bewijs: Voor het gemak werken we eerst het geval af dat $A = \emptyset$. In dit geval is $|A| = |\emptyset| = 0$ en $|\mathcal{P}(A)| = |\{\emptyset\}| = 1$, dus de stelling geldt.

Voor niet lege A laten we twee dingen zien. In de eerste plaats: $\mathcal{P}(A)$ heeft minstens dezelfde machtigheid als A. In de tweede plaats: A en $\mathcal{P}(A)$ hebben niet dezelfde machtigheid.

Het eerste is gemakkelijk: de functie $f : A \to \mathcal{P}(A)$ die wordt gedefinieerd door $f(a) = \{a\}$, voor elke $a \in A$, is een injectie. Het tweede gaat weer volgens de strategie van de vorige stelling. We nemen aan dat een bijectie F tussen A en $\mathcal{P}(A)$ gegeven is, en we construeren een deelverzameling B van A die geen F-beeld is van enig element in A. De constructie van $B \subseteq A$ gaat als volgt. Kies

$$B = \{b \in A \mid b \notin F(b)\}.$$

B is geen F-beeld van enig element van A. Veronderstel namelijk van wel. Neem aan dat we hebben: $F(c) = B$, voor zekere $c \in A$. Zit c in B? Stel van wel. Dan volgt uit de definitie van B dat $c \notin B$. Stel van niet. Dan volgt uit de definitie van B dat $c \in F(c)$, dat wil zeggen $c \in B$. Beide mogelijkheden leiden dus tot een tegenspraak. Uit het feit dat B geen F-beeld is van enig element van A volgt dat we mogen concluderen dat er geen bijectie tussen A en $\mathcal{P}(A)$ bestaat. ∎

3.3 Beslisbaarheid en opsombaarheid

We beginnen met een paar definities:

Definitie 3.11 *Een deelverzameling X van* $\mathbf{N}$ *is* **beslisbaar** *wanneer er een mechanische procedure bestaat om bij een gegeven element x van* $\mathbf{N}$ *uit te maken of $x \in X$ dan wel $x \notin X$.*

Definitie 3.12 *Een deelverzameling X van* $\mathbf{N}$ *is* **opsombaar** *wanneer er een mechanische procedure bestaat die de elementen van* $\mathbf{N}$ *in een of andere volgorde opsomt.*

Het is mogelijk het begrip 'mechanische procedure' verder te preciseren, maar we zullen dat nu alleen globaal doen. Globaal gesproken is een mechanische procedure een procedure die precies genoeg is om er een computerprogramma van te maken.

Uiteraard zal een procedure die een oneindig grote verzameling opsomt nooit stoppen. Een computerprogramma dat een oneindige deelverzameling X van $\mathbf{N}$ opsomt zal tot in lengte van dagen getallen uitbraken, en elke $x \in X$ zal vroeg of laat (na verloop van eindig veel tijd) worden afgedrukt, al weten we natuurlijk niet wanneer.

U vindt het misschien vreemd, maar er bestaan deelverzamelingen van $\mathbf{N}$ die niet beslisbaar zijn. Dit wil zeggen: er zijn verzamelingen natuurlijke getallen die de eigenschap hebben dat er geen computerprogramma te schrijven valt dat bij invoer van een willekeurig natuurlijk getal uit kan maken of dat getal tot de verzameling behoort of niet. Nog erger: er zijn verzamelingen natuurlijke getallen waarvoor zelfs geen programma te schrijven valt dat de getallen uit de verzameling in een of andere volgorde *opsomt* (zo dat je, als je maar lang genoeg bij de computer blijft staan, elk getal uit de verzameling vroeg of laat te voorschijn ziet komen). Met andere woorden: er bestaan deelverzamelingen van $\mathbf{N}$ die niet opsombaar zijn.

Wanneer we gebruik maken van ons pas verworven inzicht in de verschillende 'graden van oneindigheid' kunnen we de bewering dat er niet-opsombare getallenverzamelingen bestaan aannemelijk maken. We hebben

gezien dat de verzameling $\mathcal{P}(\mathbf{N})$ niet aftelbaar is. We redeneren nu even intuïtief. Een verzameling $X \in \mathcal{P}(\mathbf{N})$ is opsombaar wanneer er een computerprogramma bestaat in een of andere programmeertaal—zeg **Pascal**—dat die verzameling opsomt (dat wil zeggen: de elementen ervan één voor één afdrukt). Programmeertalen zijn altijd gebaseerd op een eindige verzameling schrijftekens, een uitgebreid 'alfabet'. Elk programma is een eindig rijtje schrijftekens uit dat alfabet. Zoals in § 3.2 is aangetoond, is de verzameling van eindige rijtjes over een eindig alfabet aftelbaar. De verzameling mogelijke programma's binnen de gegeven programmeertaal is dus ook aftelbaar. Omdat de verzameling deelverzamelingen van $\mathbf{N}$ *overaftelbaar* is moeten er wel deelverzamelingen van $\mathbf{N}$ zijn die door geen enkel programma worden opgesomd.

Strikt genomen dienen beweringen over beslisbaarheid en opsombaarheid allereerst te worden gepreciseerd vooraleer ze kunnen worden bewezen. De logicus Alonzo Church heeft voorgesteld *beslissingsprocedure* te preciseren als *recursieve procedure*, een begrip waarvoor hij een zeer precieze wiskundige definitie beschikbaar had. Alan Turing had het idee om *beslissingsprocedure* te lezen als: 'procedure die, als hij wordt uitgevoerd op een Turingmachine, in eindig veel stappen tot een resultaat leidt.' Bij dat voorstel hoorde natuurlijk weer een precieze—wiskundige—definitie van wat een Turingmachine is. Het werd al gauw duidelijk dat de beide preciseringsvoorstellen voor *beslissingsprocedure* op hetzelfde neerkwamen: de procedures die een Turingmachine in eindig veel stappen kan uitvoeren zijn precies de recursieve procedures.

Het preciseringsvoorstel voor *beslisbaar zijn* van Church wordt wel **de these van Church** genoemd. Die these luidt als volgt:

Hypothese 3.1 (These van Church)

Elke mechanische beslissingsprocedure is een recursieve procedure.

Een bewijs leveren van deze these is natuurlijk niet mogelijk: het is immers een voorstel om het vage en intuïtieve begrip *mechanisch beslisbaar* te lezen als *recursief*. De these van Church is in feite geen echte hypothese maar veeleer een *uitdaging* om met een mechanische beslissingsprocedure aan te komen die niet recursief is (in de precieze zin van de wiskundige definitie). Dit is tot nu toe aan niemand gelukt.

3.4 Equivalentieklassen en kardinaalgetallen

We zullen nu gaan uitleggen hoe de relatie *gelijkmachtig zijn met* kan worden gebruikt om zogenaamde *kardinaalgetallen* te definiëren: getallen die de grootte van (eindige of oneindige) verzamelingen aangeven.

Wanneer we de klasse V van alle verzamelingen beschouwen, dan is $=_1$ een relatie op V. Deze relatie heeft de volgende eigenschappen:

- $=_1$ is reflexief. Immers: voor elke verzameling A geldt dat er een bijectie van A naar A bestaat; de identieke functie op A, $\{\langle a, a\rangle \mid a \in A\}$, die we noteren als id_A, is zo'n bijectie.

- $=_1$ is symmetrisch. Immers: als f een bijectie is tussen A en B, dan is f^{-1} een bijectie tussen B en A. Hieruit volgt meteen: als $A =_1 B$, dan $B =_1 A$.

- $=_1$ is transitief. Immers: als f een bijectie is van A naar B, en g is een bijectie van B naar C, dan is de compositiefunctie $g \circ f$ een bijectie van A naar C (ga dit zelf na; teken een plaatje). Dus: als $A =_1 B$ en $B =_1 C$, dan $A =_1 C$.

Opdracht 3.16 *Welke eigenschappen heeft de relatie $\leq_1$?*

Om wat naders te kunnen zeggen over de eigenschappen van de relatie $=_1$ stappen we even over naar eigenschappen van relaties in het algemeen.

Definitie 3.13 *Een tweeplaatsige relatie R op een verzameling A die de drie eigenschappen reflexiviteit, symmetrie en transitiviteit bezit heet een* **equivalentierelatie**.

Voorbeeld 3.9 Op de verzameling M van mensen is de relatie *even groot zijn als* een equivalentierelatie.

Dat dit zo is is gemakkelijk na te gaan:

- Ieder mens is even groot als hijzelf;

- als persoon a even groot is als persoon b, dan is persoon b even groot als persoon a;

- als persoon a even groot is als persoon b, en persoon b is even groot als persoon c, dan is persoon a even groot als persoon c.

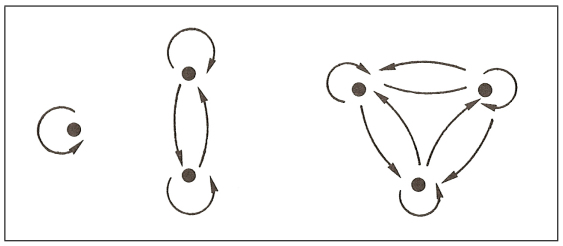
Voorbeeld 3.10 Op de verzameling M van mensen is de relatie *even oud zijn als* een equivalentierelatie. (Ga dit zelf na.)

Voorbeeld 3.11 Op de verzameling $\mathbf{N}$ van de natuurlijke getallen is de relatie R gedefinieerd door xRy desda $x + y$ is even een equivalentierelatie.

We gaan even na dat dit een equivalentierelatie is:

- voor elk getal n geldt dat $2n$ even is, dus R is reflexief;
- als de som van m en n even is, dan is de som van n en m dat ook, dus R is symmetrisch;
- als de som van m en n even is, en de som van n en k ook, dan zijn m en k hetzij allebei even, hetzij allebei oneven, dus dan is de som van m en k ook even. Dus: R is transitief.

Voorbeeld 3.12 De pijl-relatie in het volgende plaatje is een equivalentierelatie (ga zelf na).



Opdracht 3.17 *Bedenk zelf nog twee voorbeelden van equivalentierelaties. Neem als domein: de verzameling van alle mensen.*

Gegeven het feit dat R een equivalentierelatie is op een verzameling A kunnen we gaan kijken naar de *equivalentieklasse* van een element a van A.

Definitie 3.14 *Zij A een verzameling, en R een equivalentierelatie op A. De* **equivalentieklasse** *van een element a van A, modulo R, is de verzameling van alle elementen van A die in de R-relatie staan tot a.*

Notatie voor de equivalentieklasse van $a \in A$, modulo R: $[a]_R$. Elk element van de equivalentieklasse $[a]_R$ heet een **representant** van $[a]_R$. Nog wat jargon: we zeggen dat de equivalentierelatie R op A een verzameling van equivalentieklassen **induceert**. Dit jargon geeft aanleiding tot een nieuwe definitie.

Definitie 3.15 *De* **quotiëntverzameling** *van A modulo R is de verzameling equivalentieklassen die op A geïnduceerd wordt door een equivalentierelatie R.*

Notatie voor de quotientverzameling van A modulo R: A/R. Merk op dat A/R de verzameling $\{[a]_R \mid a \in A\}$ is.

Voorbeeld 3.13 De deelverzameling van de verzameling van alle mensen die gegeven is door $[\text{Jan van Eijck}]_{\text{Even-oud-als}}$ is de verzameling van de mensen die even oud zijn als Jan van Eijck. Jan van Eijck is een representant van die verzameling.

Opdracht 3.18 *De relatie R op* $\mathbf{N}$ *gedefinieerd door xRy desda $x + y$ is even, induceert twee equivalentieklassen. Ga na welke deelverzamelingen van* $\mathbf{N}$ *dit zijn.*

De relatie *even oud zijn als* induceert (op een gegeven moment) ruim honderd equivalentieklassen op de verzameling van alle mensen (namelijk de klasse van mensen die nog geen jaar oud zijn, de klasse van mensen die een jaar oud zijn, de klasse van mensen die twee jaar oud zijn, enzovoorts tot we bij de oudste aardbewoner zijn aangekomen.

Opdracht 3.19 *Hoeveel equivalentieklassen zijn er dan als de oudste persoon 120 jaar is?*

Opdracht 3.20 *Hoeveel verschillende equivalentieklassen induceert de pijlrelatie op de verzameling uit voorbeeld 3.12?*

Strikt genomen moeten we nog een paar dingen *bewijzen* over de verzameling van equivalentieklassen die wordt geïnduceerd door een equivalentierelatie.

Stelling 3.6 *Als A een verzameling is en R een equivalentierelatie op A, dan is elk element a van A lid van een element van de verzameling van equivalentieklassen die geïnduceerd wordt door R.*

Bewijs: Zij A een verzameling, en zij R een equivalentierelatie op A. Omdat R reflexief is, hebben we voor elk element a van A: aRa. Hieruit volgt meteen:

$$a \in \{y \in A \mid yRa\},$$

dat wil zeggen $a \in [a]_R$. Dus: elke $a \in A$ is lid van *minstens* één equivalentieklasse. ∎

Dat elke $a \in A$ lid is van *hoogstens* één equivalentieklasse volgt uit de volgende stelling:

Stelling 3.7 *Als A een verzameling is, en R is een equivalentierelatie op A, dan geldt, voor willekeurige elementen a en b uit A: als $[a]_R \cap [b]_R \neq \emptyset$ dan $[a]_R = [b]_R$.*

Bewijs: Stel dat $[a]_R \cap [b]_R \neq \emptyset$. Met andere woorden: er is een $c \in A$ met cRa en cRb. Neem nu een willekeurig element x van $[a]_R$. We hebben dan xRa. Uit cRa en de symmetrie van R volgt: aRc. Transitiviteit van R levert nu: xRc. We hadden al cRb, dus nogmaals transitiviteit van R toepassen levert op: xRb. Met andere woorden: $x \in [b]_R$. Omdat x een willekeurig

element van $[a]_R$ was, hebben we hiermee aangetoond dat $[a]_R \subseteq [b]_R$. Op dezelfde manier valt aan te tonen dat $[b]_R \subseteq [a]_R$. Dus $[a]_R = [b]_R$. ∎

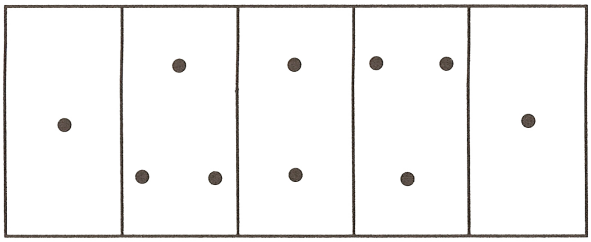
Blijkbaar deelt de verzameling equivalentieklassen die geïnduceerd wordt door een equivalentierelatie R op A, de verzameling A op in *onderling disjuncte* deelverzamelingen (deelverzamelingen die onderling geen enkel element gemeenschappelijk hebben). Zo'n opdeling van een verzameling V heet een *partitie* van V. Iets preciezer gezegd:

Definitie 3.16 *Een verzameling $\mathcal{A}$ van deelverzamelingen van A heet een* **partitie** *van A als $\mathcal{A}$ voldoet aan de volgende voorwaarden:*

- $A = \bigcup\{X \mid X \in \mathcal{A}\}$*;*
- *voor alle $X, Y \in \mathcal{A}$: $X = Y$ of $X \cap Y = \emptyset$.*

De eerste voorwaarde zegt dat elk element $a \in A$ lid is van *minstens* één verzameling uit de partitie; de tweede voorwaarde zegt dat elk element $a \in A$ lid is van *hoogstens* één verzameling uit de partitie.

We gaan weer over op aanschouwelijk onderricht. Hier is een plaatje van een partitie van een verzameling:



Opdracht 3.21 *Het is eenvoudig in te zien dat elke partitie $\mathcal{A}$ op een verzameling A ondubbelzinnig een equivalentierelatie vastlegt. Geef de equivalentierelatie R die de partitie in het bovenstaande plaatje induceert.*

Dit was een uitweiding over equivalentierelaties. Terug naar ons verhaal over de relatie $=_1$ die we hierboven hebben ingevoerd om de grootte van verzamelingen te peilen. Uit het feit dat $=_1$ een equivalentierelatie is volgt dat $=_1$ een partitie induceert op de klasse van alle verzamelingen. Voor elke verzameling A kunnen we nu de equivalentieklasse van A modulo $=_1$ beschouwen, dat wil zeggen de klasse $[A]_{=_1} = \{B \mid B =_1 A\}$.

Twee verzamelingen X en Y zitten in dezelfde equivalentieklasse modulo $=_1$ wanneer er een bijectie bestaat van X naar Y. Dus: intuïtief gesproken zitten twee verzamelingen in dezelfde equivalentieklasse modulo $=_1$

als ze 'evenveel' elementen hebben; deze equivalentieklassen duiden dus de *grootte* van verzamelingen aan. Om deze reden noemden Cantor, Frege en Russell dergelijke equivalentieklassen *kardinaalgetallen*. Het kardinaalgetal 0 is volgens deze definitie niets anders dan de equivalentieklasse $[\emptyset]_{=_1}$. Het kardinaalgetal 1 is gelijk aan de equivalentieklasse $[\{\emptyset\}]_{=_1}$.

Opdracht 3.22 *Hoeveel elementen telt de equivalentieklasse* $[\emptyset]_{=_1}$*? Hoeveel elementen telt een representant van deze equivalentieklasse?*

Opdracht 3.23 *Hoeveel elementen telt een representant van de equivalentieklasse* $[\mathcal{P}(\{\emptyset\})]_{=_1}$ *?*

In plaats van de wat moeizame notatie $[V]_{=_1}$ schrijven we meestal $|V|$. Het kardinaalgetal dat de grootte aangeeft van de verzameling $\mathbf{N}$ duiden we aan als $\aleph_0$ (spreek uit: 'alef nul'; $\aleph$ is de eerste letter uit het Hebreeuwse alfabet); we schrijven dus $|\mathbf{N}| = \aleph_0$. We zeggen ook: "de *kardinaliteit* van een aftelbare verzameling is alef nul." Evenzo is er de afspraak voor de kardinaliteit voor de verzameling reële getallen dat $|\mathbf{R}| = \aleph$.

Als we willen *rekenen* met kardinaalgetallen zullen we moeten afspreken wat we daarbij bedoelen met optellen, vermenigvuldigen en machtsverheffen. Opnieuw geldt dat deze definities zijn geïnspireerd door het eindige geval:

Definitie 3.17 *Als voor de verzamelingen A en B geldt dat $A \cap B = \emptyset$, $|A| = k$ en $|B| = \ell$, dan zijn $k + \ell$, $k \cdot \ell$, k^ℓ , $k = \ell$, $k \leq \ell$ en $k < \ell$ gedefinieerd door:*

$$
\begin{aligned}
&k + \ell = |A \cup B| \\
&k \cdot \ell = |A \times B| \\
&k^\ell = |A^B| \\
&k = \ell \Leftrightarrow A =_1 B \\
&k \leq \ell \Leftrightarrow A \leq_1 B \\
&k < \ell \Leftrightarrow k \leq \ell \ \&\ k \neq \ell.
\end{aligned}
$$

Het feit dat A en B disjunct zijn speelt eigenlijk alleen bij de optellingsregel een rol; maar ook daar is die rol marginaal want *gegeven* k en ℓ kunnen we altijd disjuncte A en B kiezen van geschikte kardinaliteit. In alle gevallen moet nog wel bewezen worden dat de resulterende kardinaliteit (binnen de gestelde condities) niet afhangt van de keuze van A en B; we laten dit over aan de lezer.

Voor de rekenkundige bewerkingen op kardinaalgetallen gelden een aantal normale algebraïsche eigenschappen:

Stelling 3.8 *Voor willekeurige kardinaalgetallen k, ℓ en m geldt:*

$k + \ell = \ell + k$	*commutativiteit van* $+$
$k \cdot \ell = \ell \cdot k$	*commutativiteit van* $\cdot$
$k + (\ell + m) = (k + \ell) + m$	*associativiteit van* $+$
$k \cdot (\ell \cdot m) = (k \cdot \ell) \cdot m$	*associativiteit van* $\cdot$
$k \cdot (\ell + m) = k \cdot \ell + k \cdot m$	*distributiviteit*
$k + \ell \leq k \cdot \ell$.	

Bewijs: Deze gelijkheden volgen direct uit de bijbehorende eigenschappen van operaties op verzamelingen. Kies om de ongelijkheid aan te tonen disjuncte verzamelingen A en B zodat $|A| = k$ en $|B| = \ell$, en vaste $a \in A$ en $b \in B$, dan is er een injectie i van $A \cup B$ naar $A \times B$ zodat voor elke x: $i(x) = \langle x, b\rangle$ als $x \in A$ en $i(x) = \langle a, x\rangle$ als $x \in B$. ∎

Stelling 3.9 *Voor willekeurige kardinaalgetallen k, ℓ en m geldt:*

$$\left.\begin{array}{l} k^\ell \cdot k^m = k^{\ell+m} \\ k^\ell \cdot m^\ell = (k \cdot m)^\ell \\ (k^\ell)^m = k^{\ell \cdot m} \end{array}\right\} \quad \textit{exponent–wetten}$$

Bewijs: Nogal bewerkelijk, maar niet echt lastig. We laten dit over aan de lezer. ∎

De belangrijkste feiten over $\aleph_0$ en $\aleph$ zijn al genoemd (of zelfs bewezen) in § 3.2:

Stelling 3.10 *Er geldt:*

$$\begin{array}{ll} \aleph_0 + \aleph_0 = \aleph_0 & \aleph_0 \cdot \aleph_0 = \aleph_0 \\ \aleph_0 + \aleph = \aleph & \aleph_0 \cdot \aleph = \aleph \\ \aleph + \aleph = \aleph & \aleph \cdot \aleph = \aleph \end{array}$$

$$\aleph_0 < 2^{\aleph_0} = \aleph = \aleph_0^{\aleph_0} = \aleph^{\aleph_0}.$$

Bewijs: zie § 3.2 voor een bewijs van de eerste regel en de vaststelling dat $2^{\aleph_0} = \aleph$. We maken daarna een sprong naar vermenigvuldiging van $\aleph$. Met gebruik van wat eerder bewezen is, volgt nu $\aleph \cdot \aleph = 2^{\aleph_0} \cdot 2^{\aleph_0} = 2^{\aleph_0 + \aleph_0} = 2^{\aleph_0} = \aleph$. We kunnen nu door *scherp afschatten* ($\leq$ is vanwege 'Cantor-Bernstein' anti-symmetrisch) hieruit de optellingseigenschap voor $\aleph$ afleiden:

$$\aleph \leq \aleph + \aleph \leq \aleph \cdot \aleph = \aleph.$$

De andere eigenschappen zijn analoog te bewijzen. ∎

In § 2.9 hebben we al laten zien dat er voor iedere verzameling A een bijectie bestaat tussen $\mathcal{P}(A)$ en $\{0,1\}^A$. De algemene diagonaalstelling toont dus aan dat het kardinaalgetal van $\{0,1\}^A$ groter is dan dat van A, voor elke A, met andere woorden, voor elk kardinaalgetal $k : k < 2^k$. Derhalve moet er een rij van steeds groter wordende oneindige kardinaalgetallen bestaan. Het kleinste oneindige kardinaalgetal is het kardinaalgetal van $\mathbf{N}$, dat we $\aleph_0$ hebben gedoopt. We noemen het eerstvolgende kardinaalgetal $\aleph_1$, het daaropvolgende $\aleph_2$, enzovoorts. Zo ontstaat de zogenaamde 'rij der alefs', de rij van oneindige kardinaalgetallen gerangschikt naar grootte:

$$\aleph_0, \aleph_1, \aleph_2, \aleph_3, \aleph_4, \ldots$$

We weten echter ook dat $\aleph_0 < \aleph$. Dus waar zit $\aleph$ in de rij van de $\aleph_n$? Deze vraag heeft de gemoederen van wiskundigen en logici gedurende vele decennia beziggehouden. Omdat $\aleph$ het kardinaalgetal is van $\mathbf{R}$, en $\mathbf{R}$ als een (continue) getallenlijn wordt voorgesteld, wordt dit probleem dat van de 'machtigheid van het continuum' genoemd. De discussie over deze vraag werd geopend met de presentatie van Cantor's 'transfinite Mengenlehre' (1895), en afgesloten met de publicatie van een artikel van Paul Cohen (1963).

Cantor vermoedde dat er geen verzamelingen bestaan met een machtigheid die tussen die van $\mathbf{N}$ en $\{0,1\}^{\mathbf{N}}$ in ligt. Hij wist al dat de machtigheid van $\mathbf{R}$ gelijk is aan die van $\{0,1\}^{\mathbf{N}}$. Zijn vermoeden luidde dus dat de machtigheid van het continuum het kardinaalgetal is dat onmiddellijk volgt op $\aleph_0$, dat wil zeggen: $\aleph_1$. Het lukte hem echter niet om dit te bewijzen. Cantor's *continuumhypothese* luidde als volgt:

Hypothese 3.2 (CH) $\aleph = \aleph_1$.

Omdat $2^{\aleph_0} = \aleph$, mogen we de continuumhypothese ook als volgt formuleren:

Hypothese 3.2 (CH–herformulering) $2^{\aleph_0} = \aleph_1$.

Deze formulering geeft aanleiding tot het opstellen van een *algemene continuumhypothese*, afgekort 'ACH' (Engels: Generalized Continiuum Hypothesis, GCH) waarvan CH een speciaal geval is:

Hypothese 3.3 (ACH) *Voor alle* $\alpha : 2^{\aleph_\alpha} = \aleph_{\alpha+1}$.

Dat dit niet slechts een bewerings over alefs is, blijkt beter in de equivalente formulering:

Hypothese 3.3 (ACH–herformulering) *Voor elke oneindige k geldt dat er geen kardinaalgetal tussen k en 2^k in ligt.*

In 1938 toonde Kurt Gödel aan dat, gesteld dat de zogenaamde Zermelo-Fraenkel axioma's voor de verzamelingenleer consistent zijn (zie ook § 3.5), het toevoegen van ACH deze theorie in elk geval niet inconsistent maakt. In 1963 liet Cohen zien dat als de Zermelo-Fraenkel axioma's consistent zijn, ook het toevoegen van de *negatie* van ACH de theorie niet inconsistent maakt. Kortom, we kunnen beide kanten uit: we kunnen zowel ACH als ¬ACH toevoegen aan ZF zonder het systeem in gevaar te brengen. We zeggen wel: ACH is *onafhankelijk* van de Zermelo-Fraenkel axioma's.

3.5 Paradoxen

U heeft wellicht gemerkt dat we nu al een paar keer het woord *klasse* hebben gebruikt op plaatsen waar u misschien het woord *verzameling* zou hebben verwacht: we hebben gesproken over de *klasse* van alle verzamelingen en over de *klasse* $[A]_{=_1}$. Dit spraakgebruik is niet toevallig; het is bedoeld om enkele voetangels en klemmen van de naïeve verzamelingenleer te vermijden.

Cantor's formulering van de verzamelingenleer (nu aangeduid als *naïeve verzamelingenleer*) leidde tot het optreden van tegenspraken, in deze context meestal 'paradoxen' genoemd. De bekendste contradictie werd in 1902 door Bertrand Russell (1872–1970) gesignaleerd; zij staat bekend als de Russell-paradox. Voor een huis-, tuin- en keukenverzameling V geldt altijd $V \notin V$. Bij voorbeeld: de verzameling van alle theelepeltjes is zelf geen theelepeltje. Beschouw nu de verzameling $R = \{V \mid V \notin V\}$. Dan geldt dat R in zichzelf zit als hij de kenmerkende eigenschap bezit, maar dan heeft hij zichzelf juist niet tot element. In formule:

$$R \in R \iff R \notin R.$$

En dit is absurd, zoals u al zonder veel logische training kunt vaststellen.

Cantor had overigens zelf drie jaar voor Russell al een probleem gevonden in zijn theorie van kardinaalgetallen, maar daar pas in 1932 over gepubliceerd. Veronderstel maar dat er een verzameling van alle verzamelingen bestaat, een *universele verzameling* U. Dan $\mathcal{P}(U) \subseteq U$ en dus $\mathcal{P}(U) \leq_1 U$, maar dit is in strijd met de algemene diagonaalstelling (stelling 3.5).

Ook de Russell-paradox volgt uit de aanname van het bestaan van een universele verzameling U. Merk op dat dan $U \in U$, en dat is al vreemd, maar het echte probleem ontstaat door uit U de deelverzameling R te lichten van alle verzamelingen die geen element zijn van zichzelf.

Om beide paradoxen te vermijden moeten we in elk geval af van de aanname dat er een verzameling van alle verzamelingen bestaat. Dat kan bij voorbeeld als volgt. We gaan uit van een aantal bona fide verzamelingen (met name: de lege verzameling en de verzameling van de natuurlijke

getallen), en van een aantal bona fide operaties op verzamelingen (met name: verenigen, doorsnijden en het vormen van machtsverzamelingen), dat wil zeggen: het uitvoeren van deze operaties op bona fide verzamelingen levert weer bona fide verzamelingen op. Verder nemen we het zogenaamde *separatie-axioma* aan:

Axioma 3.1 (separatie) *Als je een verzameling X hebt en een eigenschap E, dan bestaat er een verzameling Y die precies die elementen uit X bevat met eigenschap E.*

Nu kunnen we de volgende stelling formuleren:

Stelling 3.11 *Uit elke verzameling A kunnen we met de 'Russell-eigenschap'* **geen element zijn van zichzelf** *een verzameling B lichten. We hebben dan: $B = \{V \mid V \in A \text{ en } V \notin V\} \notin A$.*

Bewijs: Dat $B = \{V \mid V \in A \text{ en } V \notin V\}$ een verzameling is volgt uit het feit dat A een verzameling is plus het separatie-axioma. We moeten nu nog aantonen dat $B \notin A$. Uit de definitie van B volgt dat voor alle W:

$$W \in B \iff W \in A \text{ en } W \notin W.$$

Wat voor alle W geldt, geldt zeker ook voor B. Dus:

$$B \in B \iff B \in A \text{ en } B \notin B. \tag{1}$$

We passen nu een *reductio ad absurdum* toe. Neem aan dat we zouden weten dat $B \in A$. Daarmee zou (1) gereduceerd worden tot een contradictie:

$$B \in B \iff B \notin B.$$

Dus de aanname dat $B \in A$ is onjuist, en daarmee is bewezen dat $B \notin A$. ∎

Uit deze stelling volgt meteen dat er geen universele verzameling bestaat (anders gezegd: dat de *klasse* van alle verzamelingen zelf geen verzameling is).

We zijn nu toe aan de moraal van dit alles voor ons verhaal over gelijkmachtige verzamelingen. De moraal is deze: er is geen garantie dat de equivalentieklassen die door de relatie $=_1$ worden geïnduceerd op de klasse van alle verzamelingen zelf bona fide verzamelingen zijn. Sterker nog, het zijn *geen* bona fide verzamelingen, getuige de volgende stelling:

Stelling 3.12 *Er bestaat geen verzameling A zodanig dat A alle verzamelingen V bevat die precies één element hebben. Met andere woorden: de equivalentieklasse $[\{\emptyset\}]_{=_1}$ is geen verzameling.*

Bewijs: Stel dat de equivalentieklasse $[\{\emptyset\}]_{=_1}$ wel een verzameling is. Noem die verzameling A. We hebben nu:

$$V \in A \iff V =_1 \{\emptyset\}.$$

We hebben aangenomen dat het verenigen van verzamelingen een bona fide operatie is. Dus: als A een verzameling is, dan is de verzameling die ontstaat door de vereniging te nemen van alle elementen van A, dat wil zeggen de verzameling $\bigcup A$, ook een verzameling. De elementen van A zijn precies alle singletons. Dus voor elke verzameling V geldt dat $\{V\} \in A$. Hieruit volgt dat $\bigcup A$ elke verzameling bevat, ofwel: $\bigcup A$ is de verzameling van alle verzamelingen. Zo'n verzameling bestaat niet, dus de aanname dat A een verzameling is, is onjuist. $A = [\{\emptyset\}]_{=_1}$ is geen verzameling. ∎

Dit is vervelend, want het betekent dat de truc om via het overstappen van verzamelingen op hun equivalentieklassen modulo $=_1$ te abstraheren van alle verschillen tussen verzamelingen behalve hun aantallen elementen, strikt genomen niet geoorloofd is. De manier van Cantor, Frege en Russell om het begrip *kardinaalgetal* in te voeren is te simpel. Kardinaalgetallen in deze zin bestaan niet.

In de axiomatische verzamelingenleer worden kardinaalgetallen via een omweg langs zogenaamde *ordinaalgetallen* ingevoerd. Het behandelen van deze definitie zou hier te ver voeren; we volstaan daarom met een verwijzing naar het leerboek [Van Dalen e.a. 1975]. Er is nog een andere, eenvoudiger manier om problemen met de Cantor-Frege-Russell definitie van *kardinaalgetal* te vermijden. We kunnen afspreken dat we ons bij het vormen van equivalentieklassen zullen beperken tot een verzameling die bona fide is en tevens 'groot genoeg'.

Neem bij voorbeeld aan dat we een verzameling $\mathcal{A}$ hebben waar de lege verzameling $\emptyset$ in zit, plus elk natuurlijk getal, plus de hele verzameling $\mathbf{N}$, plus alles wat je kunt krijgen door een eindig aantal malen de operaties *doorsnijden*, *verenigen* en *machtsverzamelingen vormen* toe te passen op elementen van $\mathcal{A}$. Zo'n verzameling is bona fide, en voor ons in eerste instantie groot genoeg. Zij bevat bijvoorbeeld $\mathbf{N}$, $\mathcal{P}(\mathbf{N})$, $\mathcal{PP}(\mathbf{N})$, $\mathcal{PPP}(\mathbf{N})$ als elementen.

De verzameling bevat ook functies met deze elementen als domein en als co-domein: functies zijn niets anders dan verzamelingen geordende paren, en geordende paren kunnen zelf weer als verzamelingen worden gecodeerd. Voor dit laatste hebben we een trucje nodig dat door de wiskundige Von Neumann is voorgesteld: we spreken af dat $\langle a, b\rangle$ (het geordende paar van a en b) een afkorting is voor $\{\{a\}, \{a, b\}\}$.

Opdracht 3.24 *Laat zien dat geordende paren, als ze met behulp van deze*

definitie worden ingevoerd, hun essentiële eigenschap

$$\langle a, b\rangle = \langle c, d\rangle \iff a = c \text{ en } b = d$$

blijven houden.

Opdracht 3.25 *Bedenk een andere codering voor het geordende paar $\langle a, b\rangle$ en bewijs hiervoor de bovenstaande essentiële eigenschap.*

Als we nu kardinaalgetallen definiëren als equivalentieklassen modulo $=_1$ op de verzameling $\mathcal{A}$ is het gevaar van genoemde contradicties op een eenvoudige manier bezworen. Dat we bij onze definitie de moederverzameling $\mathcal{A}$ gebruiken vermelden we nu eens en voor al; we zullen het in het vervolg niet steeds in herinnering roepen.

Dit alles neemt niet weg dat de verzamelingenleer, nog wel het fundament van de moderne wiskunde, een enorme 'bug' bleek te bevatten. Die is er nu uit verwijderd, maar wie weet zijn er andere tegenspraken die we nog niet ontdekt hebben. Het zou prettig zijn als we voor eens en altijd konden *bewijzen* dat de (ingeperkte) verzamelingenleer geheel vrij van contradicties, ofwel *consistent*, is. Om dit te onderzoeken moet de verzamelingenleer exacter gedefinieerd worden. Inderdaad hebben de verzamelingsparadoxen geleid tot logische preciseringen zoals de al genoemde Zermelo-Fraenkel axioma's. Om deze predikatenlogische formules en het onderzoek naar hun onderlinge consistentie te begrijpen is het zaak wat meer over logica te weten te komen.

Hoofdstuk 4

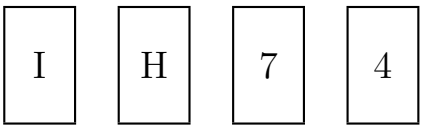
Logica, de leer van het correct redeneren

4.1 Wat is logica, en wat heb je eraan?

De gangbare omschrijving van 'logica' is: leer van het correct redeneren. Theorievorming over de manier waarop mensen *zouden moeten* redeneren heeft al vanaf de Klassieke Oudheid plaatsgehad. Er is veel voor te zeggen om de logica te beschouwen als het verst ontwikkelde onderdeel van de filosofie. Logica is een onderdeel van de filosofie (of: 'wijsbegeerte') omdat het een vak is waar mensen die zich 'filosoof' noemden zich al vanaf de Oudheid hebben beziggehouden. Er heeft sinds de syllogismen-leer van Aristoteles (384–322 voor Christus) een duidelijke vooruitgang plaatsgehad in dit vak. Die ontwikkeling is weliswaar met horten en stoten verlopen, maar zij is spectaculairder dan die in welk ander onderdeel van de filosofie ook.

Logica onderzoekt niet het feitelijk redeneren van mensen. Het is geen *empirische* wetenschap die bij voorbeeld probeert uit te zoeken waarom mensen vaak een bepaald soort (feitelijk constateerbare) vergissingen maken. Het vak logica is *normatief*: het onderzoekt de normen voor het correct redeneren. We geven een voorbeeld om het verschil aan te geven tussen een empirische en een normatieve aanpak van een redeneerprobleem.

Stelt u zich de situatie voor dat er vier kaarten op tafel liggen, en u weet dat elke kaart aan de ene kant een letter vertoont, en aan de andere kant een cijfer. De kaarten liggen als volgt:



De vraag is nu: welke twee kaarten moet ik omdraaien om de volgende bewering op juistheid te controleren:

> Van de kaarten die op tafel liggen hebben die met een *klinker* op de voorzijde een *even getal* op de achterzijde?

Uit tests is gebleken dat veel mensen (ook mensen die bij voorbeeld taalkunde, informatica of wiskunde hebben gestudeerd) bij het beantwoorden van deze vraag een bepaalde elementaire redeneerfout maken. Wie dit soort tests doet houdt zich bezig met een vorm van empirische wetenschap (bij voorbeeld: experimentele psychologie), niet met logica. De logicus is alleen geïnteresseerd in het blootleggen van het patroon dat achter de *correcte* redenering ligt.

Tot ver na de Middeleeuwen is gedacht dat het vak logica in handen van Aristoteles zijn definitieve vorm had gekregen. Die opvatting wordt zelfs nog verkondigd in Kant's *Kritik der reinen Vernunft*: zie het voorwoord in [Kant 1787]. Het zal dan nog bijna een eeuw duren voor het vak weer een grote sprong voorwaarts maakt. In 1879 verschijnt Gottlob Frege's *Begriffsschrift*, waarin in 88 bladzijden een symbooltaal wordt geïntroduceerd waarmee je alle redeneringen die Aristoteles had behandeld kunt analyseren, plus nog veel meer (zie [Frege 1879]). De door Frege geïntroduceerde taal is in feite die van de predikatenlogica, waarmee we in hoofdstuk 6 zullen kennismaken.

Frege's impuls heeft ervoor gezorgd dat logica niet alleen voor (taal)filosofen maar ook voor wiskundigen interessant werd. Het logische onderzoek van wiskundige redeneersystemen is intussen uitgegroeid tot een aparte tak van wiskunde, de zogenaamde meta-mathematica. Zie voor meer informatie: [Van Dalen 1978].

Wij blijven voorlopig wat dichter bij huis, door de vraag te stellen hoe de symbooltalen die door logici zijn voorgesteld zich verhouden tot de natuurlijke taal. In vergelijking met natuurlijke taal hebben formele talen het voordeel van een absolute precisie. Ze hebben echter ook een groot nadeel: hun gebrek aan flexibiliteit. Frege vergeleek in zijn *Begriffsschrift* de verhouding tussen natuurlijke taal en formele taal met die tussen het blote oog en de microscoop, elk met hun eigen toepassingsgebied.

De formele talen die door logici zijn ontwikkeld verschillen onderling sterk in uitdrukkingskracht. De uitdrukkingskracht van een logische taal hangt af van het scala van logische *operatoren* dat in die taal voorkomt.

De eenvoudigste (en dus minst krachtige) logische taal is die van de propositielogica. Deze taal kent alleen *logische voegwoorden* of *connectieven*, waarmee zinnen onderling verbonden kunnen zijn, zoals *en* en *of*. De eenvoudigste elementen die worden onderscheiden zijn de *basiszinnen*. De interne structuur van zo'n basiszin wordt niet nader beschouwd; daar is het analyse-apparaat niet op berekend. Een zin als "Iedereen slaapt"moet dus worden beschouwd als een ongeanalyseerd geheel. Zodra we ook zogenaamde *kwantoren* toevoegen aan ons logische arsenaal—hetgeen gebeurt in de predikatenlogica—kunnen we de structuur van deze zin verder blootleggen. De vertaling in de predikatenlogica is: $\forall x Sx$.

De predikatenlogica kan nog weer verder worden uitgebreid, zodat een nog krachtiger analysemiddel ontstaat. In de *modale* predikatenlogica kun je ook de logische functie van *modale hulpwerkwoorden* en *modale adverbia* uitdrukken, doordat zogenaamde modale operatoren $\Diamond$ ("het is mogelijk dat") en $\Box$ ("het is noodzakelijk dat") aan de taal zijn toegevoegd. De vertaling van "Niemand kan slapen" kan nu worden: $\forall x \neg \Diamond Sx$.

De meest krachtige formele taal is echter niet zonder meer de beste. Formalisering is geen doel op zichzelf, maar een *middel* om een bepaald doel te bereiken: de verheldering van de rol die de structuur van een bewering heeft in verband met de geldigheid of ongeldigheid van redeneringen waarin die bewering voorkomt. Waar het gaat om simpele redeneringen heeft het opstellen van zwaar geschut geen zin. Op de manier waarop in de logica redeneringen worden geanalyseerd gaan we in § 4.2 nader in.

Tot besluit van deze inleidende paragraaf wat relevante literatuur. Wie het vak logica geplaatst wil zien tegen de achtergrond van andere onderdelen van de filosofie, leze [Van Eijck 1982]. De rol van de logica in het alledaagse denken wordt belicht in [Emmet 1969]. Er bestaan vele inleidingen in de logica. Ze variëren qua oriëntatie van algemeen-filosofisch tot mathematisch. Het meest toegespitst op verbanden met de taalkunde is [Gamut 1982, twee delen], van harte aanbevolen. Wiskundig getinte inleidingen in de logica kunt u vinden in [Van Dalen 1983] en [Enderton 1972].

4.2 Redeneringen en redeneerschema's

Een redenering—mits voldoende gestroomlijnd—is op te vatten als een rijtje van *uitgangspunten* of *premissen*, gevolgd door een *conclusie*. Als we de conclusie scheiden van de premissen door een streep, dan kunnen we een redenering zo weergeven:

Als Jan boos is komt hij niet.
Jan is gekomen.

Jan is niet boos.

Boven de streep staan de premissen, eronder staat de conclusie. Soms is er maar één premisse, bij voorbeeld in de redenering die je nodig hebt om het klinker-getallen-probleem uit § 4.1 op te lossen:

Voor elke kaart geldt:
als er een klinker op staat, dan ook een even getal.

Voor elke kaart geldt:
als er een oneven getal op staat, dan ook een medeklinker.

We zullen hier niet ingaan op de functies van redeneringen in het dagelijks leven (waar de redeneringen vaak achteraf worden toegevoegd aan wensen of standpunten die al vast stonden, bij wijze van ideologische rechtvaardiging). Het zij slechts opgemerkt dat de redeneringen in systematische denkdisciplines—zoals daar zijn de filosofie, de theologie, de wiskunde, de informatica en de taalwetenschap—een dankbaarder object van studie vormen dan die uit de wereld van alledag. Een conclusie over het nut van logica bij het vinden van je weg in dit leven trekke u zelf...

In de logica wordt bij het bestuderen van redeneringen geabstraheerd van de inhoud. De volgende definitie speelt in dat abstractieproces de hoofdrol:

Definitie 4.1 *Een redenering heet* **geldig** *wanneer de waarheid van de premissen de waarheid van de conclusie afdwingt.*

Let op: de definitie zegt niets over de feitelijke waarheid van de premissen en de conclusie. Voor de geldigheid van een redenering is voldoende dat aan de volgende eis voldaan is: *als* de premissen waar zijn, *dan ook* de conclusie. De premissen kunnen dus best onwaar zijn, terwijl de redering geldig blijft. Aan de andere kant kunnen we ook de situatie hebben dat de premissen en de conclusie allebei waar zijn, terwijl we niettemin met een ongeldige redenering van doen hebben. De premissen en de conclusie zijn dan alleen 'toevallig' waar, om zo te zeggen. Hier is een voorbeeld van een ongeldige redenering met ware premissen en een ware conclusie:

Alle muizen hebben staarten.
Alle muizen zijn knaagdieren.

Alle knaagdieren hebben staarten.

De redenering is niet geldig, want andere redeneringen volgens ditzelfde stramien kunnen de combinatie vertonen van ware premissen met een onware conclusie. In de volgende variant op bovenstaande redenering hebben we de *inhoudswoorden* vervangen en de *functiewoorden* gehandhaafd. Deze procedure levert een redenering op waarvan de premissen waar zijn maar de conclusie niet.

Alle feministes zijn vrouwen.
Alle feministes zijn mensen.

Alle mensen zijn vrouwen.

Bij geldige redeneringen zal zoiets u niet overkomen, mits u bij het vervangen van onderdelen in de premissen en de conclusie het stramien van de redenering maar intact laat.

Blijkbaar doet het er voor de geldigheid van een redenering helemaal niet toe waar de redenering over gaat, maar alleen welk stramien hij heeft. We stappen daarom over van concrete redeneringen naar *redeneerschema's*, door stelselmatig te abstraheren van de elementen die er niet toe doen. Zo zien we dat de eerste voorbeeldredenering van deze paragraaf verloopt volgens het schema:

Als A, dan niet B.
B.

Niet A.

De letters A en B staan voor willekeurige beweringen (of: proposities). Door voor A "Jan is boos" en voor B "Jan komt" in te vullen krijgen we de bovenstaande redenering weer terug. Door voor A "Marie is verontwaardigd" en voor B "Marie groet Piet" in te vullen krijgen we een andere geldige redenering, enzovoorts.

Het schema van de redenering uit het kaarten-voorbeeld is:

Alle K zijn E.

Alle niet-E zijn niet-K.

Nu staan K en E voor uitdrukkingen die eigenschappen uitdrukken (in het jargon dat we in § 1.4 hebben ingevoerd: "die een eigenschap als denotatie hebben"). We kunnen ook andere eigenschapswoorden invullen, en we krijgen dan een andere geldige redenering (vul bij voorbeeld "kinderen" en "engelen" in).

Het schema van het muizen-voorbeeld is:

Alle P zijn Q.
Alle P zijn R.

Alle Q zijn R.

Merk op dat we tussen neus en lippen door een beetje hebben gestroomlijnd: R kan worden vervangen door "staartdier", en we krijgen "zijn staartdieren" in plaats van "hebben staarten". U ziet dat het analyseren soms enige souplesse vraagt: de analyses zijn steeds *ad hoc*, en er wordt geen *systematisch* verband gelegd tussen natuurlijke-taal zinnen en logische vormen.

In systemen die wel zo'n systematisch verband pretenderen te geven, zoals de Logische Vorm theorie binnen de Transformationele taalkunde of de Montague grammatica zijn zulke losse-pols-manoeuvres verboden.

Een stel beweringen die een bepaald schema vertonen, maar zo dat de premissen waar zijn en de conclusie niet, heet een *tegenvoorbeeld* tegen het schema. Het feministes-voorbeeld hierboven is een tegenvoorbeeld tegen het laatstgenoemde schema. We kunnen nu ook zeggen: een redeneerschema is geldig desda er geen tegenvoorbeelden tegen bestaan.

Enkele redeneerschema's zijn zo beroemd dat ze speciale namen hebben gekregen. De volgende schema's heten respectievelijk *Modus Ponens* en *Modus Tollens*.

Als A dan B.	Als A dan B.
A.	Niet B.
B.	Niet A.

U kunt proberen wat u wilt om tegenvoorbeelden tegen deze redeneerschema's te geven, het zal u niet lukken: de schema's zijn geldig.

Opdracht 4.1 *Geef een tegenvoorbeeld tegen het volgende schema:*

Als A dan niet B.
Als niet A dan B.

Opdracht 4.2 *Geef een tegenvoorbeeld tegen het volgende schema:*

Als niet A dan B.
B.
Niet A.

Hoofdstuk 5

Propositielogica

5.1 Voegwoorden en hun betekenis

"Jan is boos en hij is verdrietig" bestaat uit twee zinnen, te weten de zin "Jan is boos" en de zin "hij is verdrietig". Deze twee zinnen zijn door middel van het voegwoord *en* verbonden tot een samengestelde zin. Wanneer we even wat stroomlijnen door "hij" te vervangen door "Jan" krijgen we twee zinnen, "Jan is boos" en "Jan is verdrietig", waarvoor geldt dat informatie over het waar of onwaar zijn ervan (logisch jargon: informatie over hun *waarheidswaarde*) voldoende is om vast te stellen of de samengestelde zin al dan niet waar is. Anders gezegd: de waarheidswaarde van "Jan is boos en Jan is verdrietig" hangt af van de waarheidswaarden van "Jan is boos" en "Jan is verdrietig". En in logisch jargon heet het: het voegwoord *en* is waarheidsfunctioneel. Zoals we in § 2.8 hebben gezien betekent "een functie zijn van" hetzelfde als "afhangen van". Vergelijk ook de volgende tweetalige mededeling in een Belgisch restaurant:

> Les prix des plats sont en fonction des prix au marché.
> De prijzen der schotels zijn in functie van de marktprijzen.

Lang niet alle voegwoorden uit het Nederlands zijn waarheidsfunctioneel. Kijk bij voorbeeld naar de volgende zin: "Jan is verdrietig omdat Marie Piet heeft gezoend". Stel dat je weet dat "Jan is verdrietig" en "Marie heeft Piet gezoend" allebei waar zijn. Dan valt nog steeds niet uit de maken of ook de samengestelde zin waar is.

In de propositielogica, afgekort pL, kijken we wat de betekenis van zinnen betreft niet verder dan alleen naar het waar of onwaar zijn van die zinnen. Dit houdt in dat we alleen waarheidsfunctionele voegwoorden kunnen

behandelen. Alleen bij die voegwoorden kan de betekenis worden uitgelegd in termen van de betekenissen van de zinnen die worden samengevoegd tot een nieuwe zin. De behandelde voegwoorden zijn:

- en
- of
- als dan
- desda
- niet.

Het laatste voegwoord uit het rijtje is er een dat gecombineerd met een enkele zin al een nieuwe zin oplevert. Onze waarheidsfunctionele kijk betekent dat we afzien van allerlei aspecten in de betekenis van deze woorden. Toch blijken er ondanks deze beperkingen nog aardig wat redeneringen met behulp van propositielogica te analyseren.

5.2 Waarheidstafels

Alles wat we in de propositielogica nodig hebben zijn letters voor beweringen (*propositieletters*; ook wel: *propositionele variabelen* of *propositievariabelen*) en symbolen voor de voegwoorden. De voegwoorden noemen we voortaan *logische connectieven*. Het connectief voor negatie, $\neg$, combineert met één bewering, en we noemen het daarom *eenplaatsig*. Stel dat p een propositieletter is. Dan is $\neg p$ een samengestelde bewering, met als betekenis: "het is niet zo dat p". Het doet er niet toe voor welke bewering p staat: daarvan hadden we nu juist geabstraheerd. Het symbool voor *en* is $\wedge$ (ook veel gebruikt wordt &). We noemen $\wedge$ het conjunctieteken. $\vee$ staat voor *of* (Latijn: *v*el); we noemen dit het disjunctieteken. $\rightarrow$ staat voor *als dan*: het implicatieteken. Tenslotte hebben we $\leftrightarrow$ voor *desda*: het equivalentieteken. De connectieven $\wedge$, $\vee$, $\rightarrow$, en $\leftrightarrow$ combineren twee zinnen tot een nieuwe zin: ze heten daarom *tweeplaatsige connectieven*.

Zoals gezegd gaan we de connectieven waarheidsfunctioneel opvatten. Een mooie manier om de betekenis van de connectieven weer te geven is in de vorm van een zogenaamde *waarheidstafel*: een tabel waarin is uitgespeld welke waarheidswaarde het connectief oplevert voor alle mogelijke combinaties van waarheidswaarden van de zinnen die met behulp van dat connectief zijn samengevoegd.

We gebruiken φ en ψ voor willekeurige propositielogische beweringen. In jargon: deze symbolen zijn *metavariabelen* die staan voor propositielogische

beweringen. Een precieze definitie van 'propositielogische beweringen' volgt in § 5.4; voorlopig: alle beweringen die met behulp van de bovengenoemde connectieven kunnen worden gevormd. Een waarheidstafel voor $\wedge$ ziet er nu als volgt uit:

φ	ψ	$(\varphi \wedge \psi)$
waar	waar	waar
onwaar	waar	onwaar
waar	onwaar	onwaar
onwaar	onwaar	onwaar

Een formule van de vorm $(\varphi \wedge \psi)$ heet een *conjunctie*. De twee onderdelen φ en ψ noemen we de *conjuncten*.

Korter opschrijven van de waarheidstafel voor $\wedge$ kan ook. We gebruiken 1 voor *waar* en 0 voor *onwaar*. Het volgende staatje geeft dezelfde waarheidstafel in een andere vorm:

$\wedge$	1	0
1	1	0
0	0	0

Vanaf nu zullen we steeds 1 en 0 gebruiken voor respectievelijk *waar* en *onwaar*. We zeggen: de verzameling van waarheidswaarden is de verzameling $\{0, 1\}$.

Even terzijde: het *en* uit de omgangstaal kan natuurlijk niet helemaal in zo'n tabel gevangen worden. Immers, indien dat zo zou zijn, dan zouden de volgende twee zinnen hetzelfde moeten betekenen, *quod non*:

(1) *Hij slaakte een diepe zucht en hij overleed.*

(2) *Hij overleed en hij slaakte een diepe zucht.*

In (1) vindt de zucht voor het sterven plaats. In de propositielogica hebben we daarentegen: $(\varphi \wedge \psi)$ is waar desda $(\psi \wedge \varphi)$ waar is. In jargon: het waarheidsfunctionele connectief $\wedge$ is *commutatief*.

Een waarheidstafel voor negatie ziet er zo uit (we geven beide notatiewijzen naast elkaar):

φ	$\neg\varphi$
1	0
0	1

	$\neg$
1	0
0	1

U ziet het: wat $\neg$ doet is de twee waarheidswaarden omdraaien.

Bij *of* moeten we oppassen: de natuurlijke taal kent zowel het inclusieve als het exclusieve *of*. Exclusief *of* wordt meestal aangegeven door

of... of...: "Luister eens Jantje: of je krijgt een ijsje, of een glas limonade (maar niet zeuren om allebei)". Het disjunctieteken $\vee$ staat voor het inclusieve *of*. Hier is de waarheidstafel:

$\vee$	1	0
1	1	1
0	1	0

Opdracht 5.1 *Geef zelf een waarheidstafel voor het exclusieve* of *(gebruik het symbool a afgeleid van het Latijnse* aut*).*

Een formule van de vorm $(\varphi \vee \psi)$ heet een *disjunctie*, en φ en ψ zijn hierin de *disjuncten*.

Een formule van de vorm $(\varphi \rightarrow \psi)$ heet een *materiële implicatie* of kortweg *implicatie*; φ is hierin de *voorzin* en ψ de *nazin*. De voorzin in een implicatie wordt ook wel de *antecedent* genoemd, en de nazin de *consequent*.

De waarheidstafel voor $\rightarrow$ is vaak een didactisch struikelblok:

$\rightarrow$	1	0
1	1	0
0	1	1

Het begripsprobleem hier is dat het *als dan* uit de natuurlijke taal niet waarheidsfunctioneel is. We verwachten in een *als dan* zin dat er een of ander inhoudelijk verband is tussen de voor- en de nazin. Om een voorbeeld te noemen: "Als Marie Piet zoent, dan wordt Jan boos" wordt meestal zo opgevat dat Jan's boosheid iets met dat gezoen te maken heeft. Jan wordt boos *omdat* er gezoend wordt. Dit verband is echter niet waarheidsfunctioneel, dus in de propositielogica kan het niet worden uitgedrukt.

De achtergrond van de waarheidstafel voor $\rightarrow$ wordt gevormd door de wens om $\rightarrow$ te gebruiken voor de analyse van *als ... dan ...* in wiskundig spraakgebruik. Zie de volgende voorbeeld-bewering.

(3) *Als een verzameling A eindig is, dan is A hoogstens aftelbaar.*

Bewering (3) is van de vorm $(\varphi \rightarrow \psi)$, en de bewering is waar. Omdat de bewering van toepassing is op verzamelingen van willekeurige grootte kunnen we het volgende zeggen:

1. Wanneer de antecedent en de consequent allebei waar zijn (het geval waar A een eindige verzameling is) is implicatie (3) waar.

2. Wanneer de antecedent en de consequent allebei onwaar zijn (het geval waar A een overaftelbare verzameling is) is implicatie (3) waar.

3. Wanneer de antecedent onwaar is en de consequent waar (het geval waar A een aftelbare verzameling is) is implicatie (3) waar.

4. De enige manier om (3) onwaar te maken is door een verzameling A te vinden die eindig is zonder hoogstens aftelbaar te zijn, dat wil zeggen zonder dat er een bijectie tussen A en (een beginstuk van) **N** bestaat. Tegelijk voldoen aan beide eisen is per definitie onmogelijk. Dit vierde geval illustreert dat een implicatie onwaar is wanneer de antecedent waar is maar de consequent onwaar.

Deze overwegingen geven aanleiding tot de bovenstaande waarheidstabel. De waarheidsfunctionele manier om de knoop door te hakken wat betreft de betekenis van *als ... dan ...* heeft echter tot gevolg dat we de volgende twee zinnen—als we ze propositielogisch analyseren—waar moeten noemen:

(4) *Als 5 even is, dan is Beatrix koningin van Nederland.*

(5) *Als 5 even is, dan is Beatrix geen koningin van Nederland.*

De voorbeeldzin die nu volgt, daarentegen, is onwaar:

(6) *Als 5 oneven is, dan was Beatrix in 1989 geen koningin van Nederland.*

Ten overvloede wellicht (maar het is wel belangrijk): voor de analyse van implicatie in natuurlijke taal zitten we niet aan de propositielogica vastgebakken; er bestaat een keur van voorstellen om zwaarder logisch geschut in stelling te brengen. Wij houden ons echter hier nog even bezig met het *aap noot mies* van de logica: het waarheidsfunctioneel analyseren van beweringen.

Tenslotte een tafel voor de equivalentie:

$\leftrightarrow$	1	0
1	1	0
0	0	1

Deze waarheidsfunctionele equivalentie wordt ook wel *materiële equivalentie* genoemd.
We hebben al gezien dat het tweeplaatsige connectief $\wedge$ commutatief is.

Opdracht 5.2 *Ga aan de hand van de waarheidstafels na welke tweeplaatsige connectieven commutatief zijn en welke niet.*

Met behulp van de waarheidstafels die we hierboven gegeven hebben kunnen we nu waarheidstafels gaan maken voor willekeurig complexe formules. De precieze regels voor het construeren van complexe formules vindt u in

§ 5.4, maar we lopen alvast even vooruit: $((p \rightarrow q) \wedge (q \rightarrow p))$ is een complexe formule. We zullen voor deze formule een waarheidstafel gaan construeren. De constructie bestaat uit het stap voor stap uitrekenen van de waarheidswaarden voor de deelformules, met behulp van de informatie uit de waarheidstafels voor de verschillende connectieven:

p	q	$(p \rightarrow q)$	$(q \rightarrow p)$	$((p \rightarrow q) \wedge (q \rightarrow p))$
1	1	1	1	1
0	1	1	0	0
1	0	0	1	0
0	0	1	1	1

Deze waarheidstafel toont aan dat de volgende twee formules 'op hetzelfde neerkomen' (vergelijk de tafel voor $\leftrightarrow$):

- $((p \rightarrow q) \wedge (q \rightarrow p))$
- $(p \leftrightarrow q)$.

In termen die we in § 5.5 nader zullen toelichten: deze twee formules zijn logisch equivalent.

Opdracht 5.3 *Geef een waarheidstafel voor $\neg(p \leftrightarrow q)$. Geef vervolgens een andere formule die dezelfde waarheidstafel heeft (dat wil zeggen: die logisch equivalent is).*

Propositielogica speelt in vele programmeertalen een grote rol. Propositielogische uitdrukkingen heten daar veelal *Boolese uitdrukkingen* (Engels: *Booleans*), naar de logicus George Boole (1815–1864). Boolese uitdrukkingen kunnen worden gecombineerd met behulp van propositielogische voegwoorden `AND`, `OR`, `NOT`. Het is verleidelijk ook de bekende `IF THEN` en `IF THEN ELSE` constructies op te vatten als propositielogische voegwoorden, maar dit is strikt genomen niet correct, want het volgende stukje **Pascal** is geen Boolese uitdrukking maar een **Pascal** opdracht:

```
IF geslaagd THEN write('ok') ELSE write('fout')
```

In deze **Pascal** opdracht is `geslaagd` een Boolese variabele, dat wil zeggen een atomaire Boolese uitdrukking; `write('ok')` en `write('fout')` zijn **Pascal** opdrachten. Boolese variabelen zijn de **Pascal** tegenhangers van propositieletters; het zijn atomaire uitdrukkingen die de twee waarden *waar* en *onwaar* kunnen aannemen. De bovenstaande **Pascal** opdracht is equivalent met het volgende tweetal opdrachten:

```
IF geslaagd THEN write('ok');
IF NOT geslaagd THEN write('fout')
```

In de tweede opdracht van dit paar is `NOT geslaagd` een voorbeeld van een samengestelde Boolese uitdrukking.

5.3 BNF regels

De formele talen waarmee we ons in de logica en informatica bezighouden kunnen worden beschreven met behulp van zogenaamde *contextvrije herschrijfregels* of *contextvrije productieregels*. We zullen hier de notatie voor contextvrije herschrijfregels volgen die in informatica-kringen gebruikelijk is. Dergelijke herschrijfregels worden door informatici *BNF regels* genoemd. BNF staat voor *Backus-Naur Form*; Backus en Naur zijn twee informatici die het gebruik van dit soort herschrijfregels voor de formele definitie van programmeertalen hebben gepropageerd.

Hier is een voorbeeld van een omschrijving van een heel eenvoudig taalfragment met behulp van contextvrije herschrijfregels.

Z ::= A B **.**
A ::= C D
B ::= E A
C ::= **elke** | **een** | **geen**
D ::= **man** | **vrouw**
E ::= **bemint** | **haat**

Een verzameling herschrijfregels zoals deze noemen we een *herschrijfgrammatica*. Elke herschrijfregel heeft een linkerkant en een rechterkant, die van elkaar worden gescheiden door het speciale symbool *::=*. De linkerkant bestaat steeds uit een enkel symbool. De rechterkant bestaat uit een rijtje van symbolen; we vatten daarbij elk van de vetgedrukte woorden als een enkel symbool op. In sommige van de rijtjes aan de rechterkant van het herschrijfsymbool *::=* komt het speciale symbool | voor: dit symbool dient om alternatieven aan te geven. Beschouw de volgende regel:

D ::= **man** | **vrouw**

Dit betekent dat symbool D kan worden herschreven als het woord *man* of als het woord *vrouw*. De vetgedrukte woorden in de laatste drie regels van de voorbeeldgrammatica en de vetgedrukte punt in de eerste regel zijn *eindsymbolen*. De symbolen Z, A, B, C, D en E zijn *hulpsymbolen*.

Bij elke herschrijfgrammatica hoort een afspraak over het symbool waarmee het herschrijven begint. De bovenstaande voorbeeldgrammatica is bedoeld om Nederlandse zinnen op te leveren. Om zo'n Nederlandse zin te krijgen moeten we beginnen met het herschrijven van het symbool Z. Dit symbool heet het *beginsymbool* of het *startsymbool*.

Om na te gaan welke rijtjes van eindsymbolen zinnen van de door de herschrijfgrammatica beschreven taal zijn, starten we met het beginsymbool, en we gaan dat door toepassing van een productieregel *herschrijven* tot een

nieuw symboolrijtje; op het resultaat passen we opnieuw een productieregel toe, enzovoort: we vervangen steeds een symbool dat in de grammatica links van een pijl voorkomt door wat er rechts van die pijl staat. Hier is een voorbeeld van dit proces:

> $Z \Rightarrow$ *A B* **.** $\Rightarrow$ *C D B* **.** $\Rightarrow$ **elke** *D B* **.** $\Rightarrow$ **elke man** *B* **.** $\Rightarrow$
> **elke man** *E A* **.** $\Rightarrow$ **elke man bemint** *A* **.** $\Rightarrow$
> **elke man bemint** *C D* **.** $\Rightarrow$ **elke man bemint een** *D* **.** $\Rightarrow$
> **elke man bemint een vrouw .**

Een keten van herschrijvingen als die uit het voorbeeld noemen we een *afleiding*. Het voorbeeld toont dus aan dat in onze grammatica het startsymbool *Z* herschreven kan worden tot *elke man bemint een vrouw.*. Anders gezegd: er is een afleiding van *elke man bemint een vrouw.* uit *Z*. Een afleiding uit *Z* van een rijtje dat alleen bestaat uit eindsymbolen noemen we een *zin*. In plaats van

> Er bestaat een afleiding van *elke man bemint een vrouw.* uit *Z*

zegt men ook wel:

> *Z* brengt *elke man bemint een vrouw.* voort,

of:

> *Z* genereert *elke man bemint een vrouw.*.

Dus: *elke man bemint een vrouw.* is een *zin* die door onze voorbeeldgrammatica wordt voortgebracht.

De pijl $\Rightarrow$ geeft aan dat het symboolrijtje dat rechts van de pijl staat *direct* kan worden afgeleid van het symboolrijtje links ervan. Aan een directe afleiding komt slechts één productieregel te pas. Een afleiding waar meer dan een productieregel aan te pas komt heet *indirect*. We kunnen dus zeggen: er bestaat een indirecte afleiding van *elke man bemint een vrouw.* uit *Z*. Onze voorbeeldgrammatica genereert de volgende verzameling zinnen:

- een man bemint een man.
- elke man bemint een man.
- geen man bemint een man.
- een vrouw bemint een man.
- elke vrouw bemint een man.
- geen vrouw bemint een man.

- een man bemint een vrouw.
- ...

Opdracht 5.4 *Hoeveel verschillende zinnen genereert deze grammatica?*

Opdracht 5.5 *Ga na waarom de volgende voorbeeldgrammatica oneindig veel zinnen genereert:*

$Z ::= \mathbf{a}\ Z \mid \mathbf{b}$

Probeer een algemeen antwoord te geven op de vraag onder elke voorwaarde een eindige verzameling BNF regels oneindig veel zinnen genereert.

We resumeren nog even de verschillende soorten van symbolen die in de BNF herschrijfregels te onderscheiden vallen:

1. **metasymbolen:** Dit zijn de symbolen die worden gebruikt om aan te geven *hoe* de BNF regel moet worden gelezen. Ze hebben niets te maken met de taal die wordt gedefinieerd. Het zijn de symbolen *::=* en | . We beschouwen *::=* als een enkel symbool.
2. **hulpsymbolen:** Dit zijn symbolen die met behulp van BNF regels worden herschreven, maar die zelf niet in de de gedefinieerde taal voorkomen. Wij zullen hulpsymbolen steeds *cursief* schrijven. Een van de hulpsymbolen neemt een speciale plaats in: het startsymbool.
3. **eindsymbolen:** Dit zijn de symbolen die tot de gedefinieerde taal behoren. Eindsymbolen worden zelf nooit herschreven; ze kunnen alleen aan de rechterkant van BNF regels voor. Wij zullen eindsymbolen steeds **vet** schrijven.

Hulpsymbolen en eindsymbolen kunnen uit meerdere tekens bestaan; we beschouwen het woord *man* als een enkel eindsymbool. Wanneer we ook voor de hulpsymbolen woorden gebruiken kunnen we de structuur van het fragment verduidelijken:

zin ::= *nominale-constituent verbale-constituent* **.**
nominale-constituent ::= *determinator nomen*
verbale-constituent ::= *werkwoord nominale-constituent*
determinator ::= **elke** | **een** | **geen**
nomen ::= **man** | **vrouw**
werkwoord ::= **bemint** | **haat**

Soms worden er naast *::=* en | nog andere metasymbolen gebruikt. We gaan daar hier niet op in, want voor onze eerste toepassingen bij het definiëren van logische talen weet u nu genoeg.

5.4 De syntaxis van de propositielogica

In wiskunde en logica wordt veel gewerkt met een manier van definiëren die we *inductief* of *recursief* noemen. Het recursief definiëren van een verzameling A gaat zo.

- We noemen eerst een eindig aantal dingen waarvan we meedelen dat ze elementen zijn van de verzameling A. Dit is de *basisclausule* van de definitie.
- Vervolgens zeggen we: als we een ding hebben dat in verzameling A zit, en we voeren daar een bepaalde bewerking op uit, dan is het resultaat weer een element van A. Dit is de *recursie-clausule* van de definitie.
- Tenslotte zeggen we: behalve de elementen die je op de bovengenoemde manieren in een eindig aantal stappen kunt vormen heeft A geen elementen. Dit heet: de *afsluitingsclausule* van de recursieve definitie.

Een verzameling A kan ook recursief worden gedefinieerd zonder afsluitingsclausule. De formulering wordt dan:

- A is de *kleinste* verzameling zo dat
  1. de basisclausule opgaat,
  2. de recursieve clausule opgaat.

Deze formulering komt precies op hetzelfde neer.

We geven een aantal voorbeelden van recursieve definities: Eerst een recursieve definitie van *mens*. Daar gaat ie.

- Ten eerste: Adam en Eva zijn mensen.
- Ten tweede: kinderen van mensen zijn mensen.
- Tenslotte: verder zijn er geen mensen.

Gegeven deze definitie kun je nu gaan controleren of een bepaald object een mens is. Laten we prins Bernhard nemen. Is prins Bernhard een mens? Die vraag kunnen we herleiden tot de vraag of de ouders van de prins mensen waren. Nu, dat is niet meteen duidelijk. We moeten kijken naar *hun* ouders. Enzovoorts. Hetzij we komen bij Adam en Eva terecht (let wel: voor de prins zelf en voor alle voorouders van de prins), en Z.K.H. blijkt een mens, hetzij dat gebeurt niet, en de prins valt door de mand (volgens deze definitie).

De definitie van *natuurlijk getal* in de wiskunde gaat net zo. Ten eerste: 0 is een natuurlijk getal. Ten tweede: als iets een natuurlijk getal is, dan is

het getal dat je krijgt door 1 bij dat getal op te tellen (of met andere woorden: door de *opvolger*-functie op dat getal los te laten) ook een natuurlijk getal. Tenslotte: niets anders is een natuurlijk getal. Dit levert de bekende verzameling $\mathbf{N} = \{0, 1, 2, 3, 4, \ldots\}$.

In § 2.6 hebben we het gehad over geordende rijtjes van elementen. We hebben toen uitgelegd wat een geordend paar is, en vervolgens gezegd: geordende rijtjes kunnen ook meer dan twee elementen hebben. Naar formeel-wiskundige maatstaven gemeten is dit eigenlijk te vaag. Formeel gesproken dient het begrip 'geordend rijtje' expliciet te worden gedefinieerd, en dat kan gebeuren met een recursieve definitie. Als voorbeeld zullen we hier het begrip *geordend rijtje schrijftekens* of kortweg *tekenrijtje* recursief definiëren. In programmeertalen spelen tekenrijtjes (Engels: *strings*) een grote rol.

Eerst definiëren we de verzameling schrijftekens (Engels: *characters*). Dat gaat door opsomming, en we kunnen er een BNF regel voor gebruiken:

schrijfteken ::= **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** | **L** | **M** | **N** |
O | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** |
a | **b** | **c** | **d** | **e** | **f** | **g** | **h** | **i** | **j** | **k** | **l** | **m** | **n** |
o | **p** | **q** | **r** | **s** | **t** | **u** | **v** | **w** | **x** | **y** | **z** |
␣ | **,** | **.** | **?** | **!** | **:** | **;**

Hier staat ␣ voor het spatieteken. De recursieve definitie van *tekenrijtje* gaat nu gewoon met een tweetal BNF regels:

tekenrijtje ::= *leeg* | *schrijfteken tekenrijtje*
leeg ::=

Dat we te maken hebben met een recursieve definitie zien we aan het feit dat het hulpsymbool voor *tekenrijtje* zowel aan de linkerkant als de rechterkant in de regel voorkomt. U gelieve na te gaan dat de twee BNF regels samen neerkomen op het volgende:

Definitie 5.1 Tekenrijtjes*:*

- *Een rijtje dat helemaal niets bevat is een tekenrijtje.*
- *Een schrijfteken gevolgd door een tekenrijtje is een tekenrijtje.*
- *Niets anders is een tekenrijtje.*

U kunt zelf controleren dat *prins Bernhard* een tekenrijtje is.

Nu gaan we de verzameling formules van de propositielogica recursief definiëren. Iets preciezer: we gaan definiëren hoe een propositielogische taal $\mathcal{T}$ eruit ziet. Elke keuze van de propositieletters bepaalt een andere taal.

We zouden de volgende keuze voor de verzameling propositieletters kunnen maken: $\{p, q, r\}$. Ander keuzes zijn ook mogelijk. Zolang de verzameling eindig is kunnen de elementen gewoon worden opgesomd. Het is echter ook geen probleem om oneindig veel propositieletters in te voeren. Dat gaat natuurlijk weer gewoon met recursie. Bij voorbeeld: we willen dat de letter p gevolgd door een willekeurig aantal accenttekens een propositieletter is. Dat wil zeggen: p, p''', p'''''''' zijn voorbeelden van propositieletters. De recursieve definitie van de (oneindige) verzameling $\{p, p', p'', p''', p'''', \ldots\}$ van propositieletters gaat als volgt:

propositieletter ::= **p** | *propositieletter* **'**

Dit komt neer op het volgende:

Definitie 5.2 Propositieletters *van* $\mathcal{T}$*:*

- p *is een propositieletter;*
- *als* A *een propositieletter is, dan* A' *ook;*
- *niets anders is een propositieletter.*

In deze definitie is A weer een *metavariabele*: A staat voor een willekeurige propositieletter die al eerder is geconstrueerd. Propositieletters die op deze manier zijn geconstrueerd zijn voor ons mensen soms moeilijk uit elkaar te houden, maar een computer draait er zijn hand niet voor om. In gevallen waar je maar een paar propositieletters nodig hebt is het handiger om gewoon verschillende letters van het alfabet te nemen. Vaak worden dan de letters p, q, r en s gebruikt.

Elke verzameling propositieletters bepaalt een propositielogische taal $\mathcal{T}$. Een propositielogische taal $\mathcal{T}$ is niets anders dan een verzameling *welgevormde formules* (Engels: *wellformed formulas—wffs*). We definiëren met behulp van het begrip *propositieletter* het begrip *welgevormde formule* (voor zekere taal $\mathcal{T}$, waarbij de verzameling propositieletters de taal $\mathcal{T}$ bepaalt). Weer geven we twee versies: eerst met behulp van BNF regels, daarna met een expliciete definitie. Hier is de BNF definitie:

formule ::= *propositieletter* | $\neg$ *formule* | (*formule* $\wedge$ *formule*) |
(*formule* $\vee$ *formule*) | (*formule* $\rightarrow$ *formule*) |
(*formule* $\leftrightarrow$ *formule*)

Welke taal wordt gedefinieerd hangt af van de BNF regel voor *propositieletter*.

We kunnen ditzelfde ook expliciet in een recursieve definitie verwoorden. Daarbij gebruiken we φ en ψ als meta-variabelen voor propositielogische formules.

Definitie 5.3 Welgevormde formules *van $\mathcal{T}$:*

- *Elke propositieletter van $\mathcal{T}$ is een welgevormde formule van $\mathcal{T}$;*
- *als φ een welgevormde formule van $\mathcal{T}$ is, dan $\neg\varphi$ ook; als φ en ψ welgevormde formules van $\mathcal{T}$ zijn, dan $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$ en $(\varphi \leftrightarrow \psi)$ ook;*
- *niets anders is een welgevormde formule van $\mathcal{T}$.*

De propositieletters van $\mathcal{T}$ worden ook wel de *atomaire formules van $\mathcal{T}$* of de *atomen van $\mathcal{T}$* genoemd. φ en ψ—de metavariabelen uit de definitie—staan voor willekeurige formules van de taal $\mathcal{T}$ die al geconstrueerd zijn.

Bij recursieve definities horen *inductieve bewijzen*. Om te bewijzen dat alle welgevormde formules van $\mathcal{T}$ zekere eigenschap E hebben lopen we de recursieve definitie langs, en we controleren twee dingen:

1. dat in het basisgeval E aanwezig is;
2. dat de eigenschap E bij het toepassen van de recursieve clausule bewaard blijft.

De eerste controlestap wordt de *basisstap* genoemd; de tweede controlestap heet de *inductiestap*.

Een kinderachtig voorbeeld: u kunt inductief bewijzen dat elke propositieletter uit de verzameling $\{p, p', p'', p''', \ldots\}$, gedefinieerd als boven, begint met de letter p.

Opdracht 5.6 *Schrijf dit inductieve bewijs op.*

Iets minder kinderachtig, maar nog steeds flauw: u kunt inductief bewijzen dat de welgevormde formules van $\mathcal{T}$ evenveel linker- als rechterhaakjes hebben. Het bewijs gaat zo.

Stelling 5.1 *De welgevormde formules van een propositielogische taal $\mathcal{T}$ hebben evenveel linker- als rechterhaakjes.*

Bewijs:

- Basisstap: propositieletters hebben geen haakjes, dus zeker evenveel linker- als rechter.
- Inductiestap: als φ evenveel linker- als rechterhaakjes heeft, dan $\neg\varphi$ ook: er komen immers geen haakjes bij; als zowel φ als ψ evenveel linker- als rechterhaakjes heeft, dan $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$ en $(\varphi \leftrightarrow \psi)$ ook: er komt steeds precies één linker- en één rechterhaakje bij. ∎

Het is van belang grondig vertrouwd te raken met inductieve bewijzen. De volgende opdracht laat een toepassing zien in het redeneren over verzamelingen.

Opdracht 5.7 *In § 2.5 hebben we zonder bewijs vermeld dat voor eindige verzamelingen A geldt: als A n elementen heeft dan heeft $\mathcal{P}(A)$ 2^n elementen. Ga na waarom dit zo is. Aanwijzing: u kunt dit als volgt nagaan met behulp van inductie:*

- *Basisstap: Een verzameling van 0 elementen heeft een machtsverzameling van $1 = 2^0$ element.*
- *Inductiestap: Neem aan dat je al weet dat verzameling A met $n - 1$ elementen een machtsverzameling van 2^{n-1} elementen heeft. Laat met behulp van dit gegeven zien dat een verzameling van n elementen tweemaal 2^{n-1} elementen heeft (dat wil zeggen: $2 \times 2^{n-1} = 2^n$ elementen).*

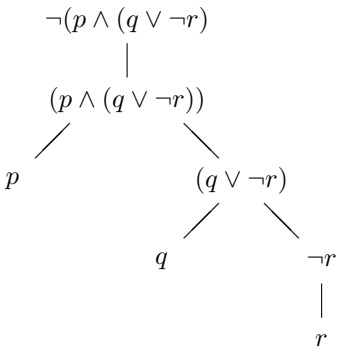
Opdracht 5.8 *Laat $\mathcal{T}$ de propositielogische taal zijn met als propositieletters de verzameling $\{p, q, r\}$. Scheid de welgevormde formules van $\mathcal{T}$ van de rest:*

1. pq
2. $p \wedge q$
3. $(p \wedge q)$
4. $(p \wedge (q \vee r))$
5. (p'')
6. $\neg\neg p$
7. $(\neg\neg p)$
8. $(\neg(\neg p))$
9. $p \vee p$
10. $(p \vee p)$
11. $(p \vee q \vee r)$
12. $(p \rightarrow (p \rightarrow p))$
13. $(p \rightarrow p) \rightarrow p$
14. $\neg\neg\neg\neg\neg\neg\neg q$.

Opdracht 5.9 *Hoeveel formules heeft de taal $\mathcal{T}$ uit de vorige opdracht?*

Opdracht 5.10 *Als de propositielogische taal $\mathcal{T}$ aftelbaar veel propositieletters heeft, hoeveel formules heeft $\mathcal{T}$ dan?*

U zult bij het maken van opdracht 5.8 hebben gemerkt dat de definitie van *welgevormde formule* heel precies vastlegt waar de haakjes horen. De haakjes zijn van groot belang voor het vermijden van dubbelzinnigheden: ze zorgen ervoor dat elke formule op precies één manier kan worden 'ontleed'. Ontleden van een propositielogische formule is niets anders dan nagaan op welke manier die formule volgens de constructieregels uit de recursieve definitie van *welgevormde formule* is opgebouwd. De opbouw van een formule kunnen we weergeven in een zogenaamde *constructieboom* (Engels: construction tree). Hier is een voorbeeld, voor de formule $\neg(p \wedge (q \vee \neg r))$:



De *knopen* van de constructieboom worden gevormd door welgevormde formules; de welgevormde formule die de topknoop-positie inneemt is geconstrueerd met behulp van de welgevormde formules op de lager gelegen knopen. Als u wilt kunt u voor elke knoop nagaan volgens welke clausule in de recursieve definitie de formule op die knoop is gevormd.

Uit de constructieboom voor een formule kan het zogenaamde *bereik* (Engels: *scope*) van de verschillende connectieven die in de formule voorkomen worden afgelezen. Een connectief heeft bereik over de formule of formules op de knoop of knopen waar u terecht komt door vanaf de knoop waar het desbetreffende connectief wordt geïntroduceerd langs een tak van de constructieboom precies één stapje naar beneden te doen. In boomjargon: een connectief heeft bereik over de formules op de knopen die *onmiddellijk worden gedomineerd* door de knoop waar het connectief wordt geïntroduceerd.

Opdracht 5.11 *Geef van beide negatietekens in de formule $\neg(p \wedge (q \vee \neg r))$ het bereik aan.*

Opdracht 5.12 *Maak een constructieboom voor de formule*

$$(\neg p \rightarrow (\neg q \wedge r))$$

en geef het bereik van de beide negatietekens aan.

Wanneer we in de formule $(p \wedge (q \vee r))$ de haakjes zouden weglaten, zouden we niet meer kunnen nagaan dat deze formule ontstaan is door de conjunctie te nemen van de atomaire formule p en de formule $(q \vee r)$; immers, we zouden dan evengoed te maken kunnen hebben met de disjunctie van de conjunctie van p en q aan de ene kant, en de atomaire formule r aan de andere kant. Door weglating van de haakjes zou de formule $(p \wedge (q \vee r))$ op meerdere manieren kunnen worden gelezen. Omdat deze verschillende lezingen niet op hetzelfde neerkomen zou dit betekenen dat sommige formules van de propositielogica dubbelzinnig zouden zijn geworden. De haakjes dienen om dit te voorkomen. Die haakjes zorgen ervoor dat iedere formule precies één constructieboom heeft.

Er zijn echter ook gevallen waar we het ons kunnen permitteren wat slordiger om te springen met de de haakjes. In elke formule die begint met een linkerhaakje en eindigt met een rechterhaakje kunnen die buitenste haakjes worden weggelaten zonder dat dit dubbelzinnigheden oplevert. Dus: in plaats van $(p \wedge (q \vee r))$ kunnen we zonder bezwaar schrijven: $p \wedge (q \vee r)$. Dit weglaten van buitenste haakjes maakt de formules in het algemeen wat leesbaarder: het wordt dan ook veel gedaan.

Opdracht 5.13 *Geef een stel BNF regels die precies de formules van een propositielogische taal met een gegeven verzameling propositieletters opleveren, maar nu volgens het recept dat buitenhaakjes niet worden geschreven. Aanwijzing: u heeft een extra hulpsymbool* naaktloper *nodig, voor formules zonder buitenhaakjes.*

Een andere situatie waarin haakjes zonder gevaar van dubbelzinnigheid kunnen worden weggelaten is in een formule als $(p \wedge (q \wedge (r \wedge s)))$. De volgorde waarin de conjuncties worden genomen doet er voor de betekenis van het geheel niet toe, dus we kunnen net zo goed schrijven: $p \wedge q \wedge r \wedge s$. In het vervolg zullen we ons zo nu en dan dit soort vrijheden veroorloven. Dit betekent niet dat we de definitie van *welgevormde formule* hebben aangepast, maar alleen dat we er een beetje mee sjoemelen waar het geen kwaad kan.

De constructiebomen die we hierboven gegeven hebben zijn familie van de syntactische structuurbomen die in de moderne taalkunde worden gebruikt. Om de verbanden precies te kunnen bespreken hebben we een formele definitie nodig van het begrip *boom*. Wij geven er echter de voorkeur

aan om u eerst intuïtief vertrouwd te laten raken met bomen, en u op die manier te prepareren voor een echt formele behandeling.

We hebben hierboven gezien dat haakjes worden gebruikt om dubbelzinnigheden in propositielogische formules te vermijden. Er bestaat overigens ook een manier om dergelijke dubbelzinnigheden te vermijden zonder gebruik te maken van haakjes, de zogenaamde *prefix notatie* of *Poolse notatie*. De laatste benaming herinnert aan het feit dat deze notatie door Poolse logici is voorgesteld. In de prefix notatie worden de tweeplaatsige connectieven *voor* de formules die ze samenvoegen geplaatst in plaats van *ertussen* zoals bij de gewone notatie. Nog wat jargon: de gewone notatie wordt ook wel *infix* notatie genoemd. Haakjes blijken in de prefix notatie overbodig te zijn. Hier zijn een aantal voorbeelden van Pools genoteerde propositielogische formules:

- p
- $\wedge pq$
- $\wedge \vee pqr$
- $\wedge p \vee qr$
- $\wedge \vee pq \vee r\neg r$.

Opdracht 5.14 *Geef constructiebomen voor deze formules. Geef ook bij elke formule de corresponderende infix-versie.*

De verzameling van welgevormde formules in Poolse notatie (van een taal $\mathcal{T}$ met verzameling propositieletters P) kan weer recursief worden gedefinieerd.

Opdracht 5.15 *Geef deze definitie.*

Opdracht 5.16 *Geef een stel BNF regels die de verzameling van welgevormde formules van de propositielogica in Poolse notatie oplevert.*

Behalve een infix en een prefix notatie voor formules van de propositielogica bestaat er ook een zogenaamde *postfix notatie*, ook wel *omgekeerd Poolse notatie* (Engels: reverse Polish notation) genoemd. Omgekeerd Poolse notatie is een zeer geschikt voor computerverwerking; een programmeertaal die gebruik maakt van de voordelen van deze notatie is de taal FORTH (zie [Winfield 1983]).

Opdracht 5.17 *Geef de postfix-versies van de volgende formules in prefix notatie:*

1. p
2. $\wedge pq$
3. $\wedge \vee pqr$
4. $\wedge p \vee qr$
5. $\wedge \vee pq \vee r\neg r$
6. $\neg\neg p$.

Opdracht 5.18 *Geef een recursieve definitie van de verzameling welgevormde formules van een propositielogische taal in postfix notatie (ga uit van een taal $\mathcal{T}$ met P als verzameling propositieletters).*

Een laatste notatievariant voor propositielogische formules is de zogenaamde *lijst-notatie* (soms ook Cambridge Pools genoemd, omdat deze notatie gebruikt wordt in LISP, een programmeertaal die is ontwikkeld aan het MIT in Cambridge, Massachusetts). Hier zijn de formules uit opdracht 5.17 in lijstnotatie:

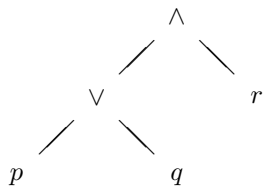
- p
- $(\wedge pq)$
- $(\wedge(\vee pq)r)$
- $(\wedge p(\vee qr))$
- $(\wedge(\vee pq)(\vee r(\neg r)))$
- $(\neg(\neg p))$.

U ziet het: Pools, maar met kwistig gebruik van extra haakjes.

Opdracht 5.19 *Geef een recursieve definitie van de verzameling welgevormde formules in lijst-notatie van de propositielogische taal $\mathcal{T}$ met verzameling propositieletters P.*

In [Van Eijck 1987, hoofdstuk 6] kunt u voorbeelden vinden van **Pascal** programma's voor het herkennen en manipuleren van propositielogische formules in een aantal verschillende notaties.

Nauw verwant aan constructiebomen voor formules zijn de zogenaamde *structuurbomen*. Hier is een voorbeeld van een structuurboom voor een formule (let op; we schakelen weer over op de standaardnotatie). De structuurboom voor de formule $((p \vee q) \wedge r)$ is:



Zoals u aan dit voorbeeld kunt zien wordt het bereik van de verschillende connectieven in een structuurboom aangegeven door de positie die die connectieven in de boom innemen. Dus: de rol van de haakjes in een propositielogische formule wordt in een structuurboom voor die formule gespeeld door de hiërarchische verhoudingen tussen de knopen in de boom.

Het is niet zo moeilijk om in te zien dat structuurbomen kunnen dienen als de grondstof waaruit de verschillende notaties voor formules (standaard, Pools, omgekeerd Pools, lijst-notatie) kunnen worden toebereid. Al deze notaties kunnen worden verkregen door mechanische bewerkingen op structuurbomen waarbij de knopen uit die bomen op een bepaalde systematische manier worden verwerkt.

Opdracht 5.20 *Geef structuurbomen voor de volgende formules:*

1. $\neg\neg\neg p$
2. $\neg(p \vee \neg q)$
3. $\neg(\neg p \wedge \neg q)$
4. $((p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p))$.

5.5 De semantiek van de propositielogica

De semantiek van de propositielogica is in principe niets anders dan een systematische verhandeling over hoe je kunt uitmaken of bepaalde propositielogische formules waar zijn of niet. In die systematische verhandeling wordt royaal gebruik gemaakt van begrippen uit de verzamelingenleer.

Laat P een verzameling propositieletters zijn, en $\mathcal{T}$ de bijbehorende propositielogische taal. De verzameling welgevormde formules van $\mathcal{T}$ noemen we $\mathrm{WF}_{\mathcal{T}}$. We zijn nu geïnteresseerd in zogenaamde *waarderingen* of *valuaties* of *valuatie-functies* (Engels: valuations) voor $\mathrm{WF}_{\mathcal{T}}$. Een waardering voor de propositielogische taal $\mathcal{T}$ is een functie V die aan alle formules van $\mathcal{T}$ een waarheidswaarde toekent, en die dat doet op een waarheidsfunctionele manier. Iets preciezer: een *waardering* V voor propositielogische taal $\mathcal{T}$ is een functie met domein $\mathrm{WF}_{\mathcal{T}}$ en codomein $\{0, 1\}$. V levert voor elke formule

van $\mathcal{T}$ een waarheidswaarde op. Wat wil het nu zeggen dat V waarheidsfunctioneel is? Ruwweg het volgende: V houdt zich aan de betekenissen van de connectieven.

We hebben in § 5.1 en § 5.2 al gezien dat de propositielogische connectieven *waarheidsfunctioneel* zijn. Dit houdt in dat twee waarderingen V en V' voor een taal $\mathcal{T}$ alleen verschillend kunnen zijn—dat wil zeggen: voor sommige welgevormde formules verschillende waarden kunnen opleveren—wanneer ze verschillende waarheidswaarden toekennen aan minstens één propositieletter. Immers, gesteld dat V en V' aan elk van de propositieletters van $\mathcal{T}$ dezelfde waarde toekennen, dan worden de toekenningen voor alle andere formules *afgedwongen* door de waarheidstafels. Met andere woorden: V en V' kennen dan aan elke formule dezelfde waarde toe. Daarmee bedoelen we dat voor elke formule φ geldt:

$$V(\varphi) = V'(\varphi).$$

Gevolg van dit alles is: als we geïnteresseerd zijn in valuatie-functies hoeven we alleen te kijken naar de toekenningen van waarheidswaarden aan de propositieletters van onze taal. Met die toekenning ligt de waardering voor *alle* formules van de taal vast. Een toekenning van waarheidswaarden aan de propositieletters van de taal $\mathcal{T}$ heet een *interpretatie* voor $\mathcal{T}$. Formeel:

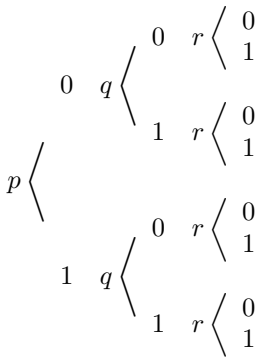
Definitie 5.4 *Als $\mathcal{T}$ de propositielogische taal is met P als verzameling propositieletters, dan is een functie $I : P \to \{0, 1\}$ een* **interpretatie** *voor $\mathcal{T}$.*

Zoals we in § 2.9 hebben gezien heet een functie met de verzameling $\{0, 1\}$ als codomein een karakteristieke functie. Karakteristieke functies corresponderen met deelverzamelingen, dus we kunnen zeggen: een interpretatie voor $\mathcal{T}$ karakteriseert een deelverzameling van de verzameling P van propositieletters van $\mathcal{T}$.

Laten we als voorbeeld de taal $\mathcal{T}$ nemen die gebaseerd is op de verzameling $\{p, q, r\}$ van propositieletters. Nu kun je een interpretatie I_1 voor $\mathcal{T}$ geven door te zeggen: $I_1(p) = 1$, $I_1(q) = 1$, $I_1(r) = 0$. Interpretatie I_2 kan er zo uitzien: $I_2(p) = 1$, $I_2(q) = 0$, $I_2(r) = 0$. I_2 verschilt nu van I_1 in de toekenning van een waarde aan de propositieletter q.

Voor een taal met drie propositieletters zijn er $2^3 = 8$ mogelijke interpretaties. Immers, bij elke extra propositieletter verdubbelt het aantal mogelijkheden, zoals uit het volgende plaatje blijkt (stel dat p, q en r de

propositieletters zijn):



Uit het plaatje blijkt dat er in totaal 8 mogelijkheden zijn. Dit klopt ook met het deelverzamelingen-perspectief: een verzameling met 3 elementen heeft 8 onderling verschillende deelverzamelingen, een verzameling met n elementen heeft 2^n onderling verschillende deelverzamelingen.

Waarom heet een functie van propositieletters naar waarheidswaarden nu een *interpretatie*? Omdat zo'n functie de rol vervult van 'een specifieke betekenis geven' aan de propositieletters. Wat een interpretatie-functie doet komt op hetzelfde neer als het vervangen van de verzameling propositieletters door een verzameling direct controleerbare beweringen over de werkelijkheid.

Een interpretatie in de huis-, tuin- en keukenzin van een propositielogische taal met als propositieletters de verzameling {p, q, r} zou bij voorbeeld kunnen zijn:

$p \longrightarrow$ 'Het regent op koninginnedag '88 in Soestdijk.'
$q \longrightarrow$ 'Het sneeuwt op koninginnedag '88 in Soestdijk.'
$r \longrightarrow$ 'Het hagelt op koninginnedag '88 in Soestdijk.'

Helaas, dit is nogal een mond vol. En we hebben gezien dat het enige onderscheid dat de propositielogica kan maken is: dat tussen ware beweringen en onware. Elke ware bewering is even goed als een andere ware bewering. Dit betekent dat we—in plaats van het bovenstaande—net zo goed kunnen zeggen:

$p \longrightarrow$ een willekeurige ware bewering
$q \longrightarrow$ een willekeurige onware bewering
$r \longrightarrow$ een willekeurige onware bewering

Maar dat is weer niets anders dan de volgende interpretatiefunctie met domein $\{p, q, r\}$:

$$\begin{array}{lcl} p & \longrightarrow & 1 \\ q & \longrightarrow & 0 \\ r & \longrightarrow & 0 \end{array}$$

Vandaar dat de definitie van 'interpretatie' is zoals hij is.

We gaan het nu hebben over de manier waarop een interpretatiefunctie I kan worden uitgebreid tot een waardering V_I voor alle welgevormde formules van de taal. Let op: het uitbreidingsvoorschrift volgt de inductieve definitie van de welgevormde formules. We nemen als voorbeeld de taal $\mathcal{T}$ met de verzameling propositieletters $\{p, q, r\}$. Laat een interpretatiefunctie I voor deze verzameling gegeven zijn. We definiëren nu eerst V_I voor atomaire formules, daarna voor willekeurige formules.

- $V_I(p) = I(p)$; $V_I(q) = I(q)$; $V_I(r) = I(r)$.

In het algemeen: als φ een atomaire formule is, dan stellen we:

- $V_I(\varphi) = I(\varphi)$.

De functie I was gedefinieerd voor alle atomaire formules. V_I is voor atomaire formules gewoon gelijk aan I.

- $V_I(\neg\varphi) = 1$ desda $V_I(\varphi) = 0$.

Toelichting: Gesteld dat we al weten welke waarde V_I toekent aan φ, dan kunnen we met behulp van deze clausule de waarde voor $\neg\varphi$ afleiden.

- $V_I((\varphi \wedge \psi)) = 1$ desda $V_I(\varphi) = 1$ en $V_I(\psi) = 1$.

Toelichting: Gesteld dat we de waarheidswaarden van φ en ψ al weten, dan kunnen we die van $(\varphi \wedge \psi)$ afleiden. Hetzelfde geldt voor de samenstellingen die met behulp van de andere connectieven kunnen worden gevormd, zoals blijkt uit de volgende clausules.

- $V_I((\varphi \vee \psi)) = 1$ desda $V_I(\varphi) = 1$ of $V_I(\psi) = 1$.
- $V_I((\varphi \rightarrow \psi)) = 0$ desda $V_I(\varphi) = 1$ en $V_I(\psi) = 0$.
- $V_I((\varphi \leftrightarrow \psi)) = 1$ desda $V_I(\varphi) = V_I(\psi)$.

Hiermee hebben we alle gevallen gehad, en dus hebben we V_I nu inductief gedefinieerd voor alle formules van $\mathcal{T}$. Aan de hand van de waarheidstafels voor de verschillende connectieven kunt u nagaan dat de bovenstaande clausules juist zijn.

Nu we beschikken over valuaties kunnen we waarheidstafels in een nieuw licht gaan bekijken. Een waarheidstafel is niets anders dan een constructie

van alle mogelijke valuaties voor een formule, uitgaande van de verschillende interpretatiemogelijkheden van de propositieletters die erin voorkomen.

Als we een tafel willen maken voor de formule $\neg(\neg p \vee \neg q)$, dan beginnen we met alle mogelijke interpretaties voor de twee propositieletters p en q. Dat zijn er vier:

	p	q
I_1	1	1
I_2	0	1
I_3	1	0
I_4	0	0

Het opbouwen van de waarderingen V_1, V_2, V_3 en V_4 gaat nu als volgt:

	p	q	$\neg p$	$\neg q$	$(\neg p \vee \neg q)$	$\neg(\neg p \vee \neg q)$
V_1	1	1	0	0	0	1
V_2	0	1	1	0	1	0
V_3	1	0	0	1	1	0
V_4	0	0	1	1	1	0

Merk op dat uit de waarheidstafel blijkt dat $\neg(\neg p \vee \neg q)$ op hetzelfde neerkomt als $(p \wedge q)$.

We zijn weer toe aan het invoeren van wat nieuw logisch jargon.

Definitie 5.5 *Twee propositielogische formules φ en ψ heten* **logisch equivalent** *als voor elke mogelijke waardering V geldt: $V(\varphi) = V(\psi)$.*

In waarheidstafel-termen betekent dit: φ en ψ zijn logisch equivalent als ze dezelfde waarheidstafel opleveren. We kunnen nu dus zeggen: de formules $\neg(\neg p \vee \neg q)$ en $(p \wedge q)$ zijn logisch equivalent.

Opdracht 5.21 *Ga na dat $\neg(p \wedge q)$ en $(\neg p \vee \neg q)$ logisch equivalent zijn.*

Definitie 5.6 *Een formule φ heet een* **tautologie** *(of: een* **logische waarheid***, of ook:* **geldig***) als voor elke mogelijke waardering V geldt: $V(\varphi) = 1$.*

In waarheidstafel-termen: φ is een tautologie als de waarheidstafel voor φ louter enen heeft.

Opdracht 5.22 *Welke van de volgende formules zijn tautologieën?*

1. $(p \rightarrow p)$
2. $\neg(p \rightarrow p)$
3. $(p \rightarrow (\neg p \rightarrow p))$

4. $((p \wedge q) \leftrightarrow \neg(\neg p \vee \neg q))$

5. $(\neg p \rightarrow (p \rightarrow (q \rightarrow r)))$

6. $(\neg(p \leftrightarrow q) \rightarrow ((p \leftrightarrow r) \vee (q \leftrightarrow r)))$.

Opdracht 5.23 *Ga met behulp van waarheidstafels na dat de volgende formules tautologieën zijn (voor elke keuze van φ, ψ en χ ; we hebben hier dus eigenlijk te maken met tautologie-*schema's *of tautologie-sjablonen):*

1. $\varphi \vee \neg\varphi$ ['wet van de uitgesloten derde']

2. $(\varphi \wedge \psi) \rightarrow \varphi$

3. $\varphi \rightarrow (\varphi \vee \psi)$

4. $\neg\varphi \rightarrow (\varphi \rightarrow \psi)$ ['ex falso sequitur quodlibet']

5. $((\varphi \rightarrow \psi) \rightarrow \varphi) \rightarrow \varphi$ ['wet van Peirce']

6. $(\varphi \rightarrow (\psi \rightarrow \chi)) \rightarrow ((\varphi \rightarrow \psi) \rightarrow (\varphi \rightarrow \chi))$.

Definitie 5.7 *Een formule φ heet een* **contradictie** *(of: een* **logische onwaarheid***) als voor elke mogelijke waardering V geldt: $V(\varphi) = 0$.*

In waarheidstafel-termen: φ is een contradictie als de waarheidstafel voor φ louter nullen heeft.

Opdracht 5.24 *Welke van de formules uit de vorige twee opdrachten zijn contradicties?*

Opdracht 5.25 *Laat zien dat de ontkenning van een tautologie altijd een contradictie is en omgekeerd.*

Opdracht 5.26 *Geef een voorbeeld van een formule die noch een tautologie, noch een contradictie is.*

De laatste opdracht was gemakkelijk: u zult eruit begrepen hebben dat lang niet alle formules tautologieën of contradicties zijn.

Definitie 5.8 *Een formule die noch een tautologie, noch een contradictie is noemen we een* **contingente** *formule.*

We vertalen het bovenstaande jargon weer even in huis-, tuin- en keukentermen. De waarheid van een tautologie hangt niet af van de interpretatie van de propositieletters die erin voorkomen. Dat maakt hem nu juist *logisch* waar: toetsing aan de werkelijkheid, kennis van de wereld, of iets dergelijks,

is *niet* nodig om de waarheid van een tautologie vast te stellen. Nog anders gezegd: ik hoef niet uit het raam te kijken om vast te stellen dat de bewering 'het regent of het regent niet' waar is. Een parafrase van 'contingente formule': contingente formules zijn formules waarvan de waarheidswaarde afhangt van de interpretatie. Nog anders: contingente formules hebben een waarheidstafel met *niet* alleen enen en *niet* alleen nullen.

We zijn nu toe aan de definitie van *logisch gevolg*, het kernbegrip uit de propositielogica. Dit is het belangrijkste stukje logisch jargon uit de propositielogica omdat propositielogica uiteindelijk de leer van het correct propositielogisch redeneren is (vergelijk ook hoofdstuk 4).

Definitie 5.9 *Een formule φ heet een* **logisch gevolg** *van formule ψ als voor elke valuatie V met $V(\psi) = 1$ geldt dat $V(\varphi) = 1$.*

Net zo voor het geval waar er meerdere premissen zijn, zeg $\psi_1, \ldots, \psi_n$. We zeggen dan: de premissenverzameling is de verzameling $\{\psi_1, \ldots, \psi_n\}$. Noem de premissenverzameling Σ. Dan kunnen we de definitie van *logisch gevolg* generaliseren:

Definitie 5.10 *Formule φ is een* **logisch gevolg** *van de premissenverzameling Σ als voor elke valuatie V die voor elke formule in Σ de waarde 1 oplevert geldt: $V(\varphi) = 1$.*

We noteren dit als:

$$\psi_1, \ldots, \psi_n \models \varphi$$

of ook wel als:

$$\{\psi_1, \ldots, \psi_n\} \models \varphi$$

of gewoon:

$$\Sigma \models \varphi.$$

Hier zijn $\psi_1, ..., \psi_n$ de premissen, en is φ de conclusie. We zeggen ook wel: het redeneerpatroon 'concludeer uit premissenverzameling Σ tot conclusie φ' is *geldig*. We zijn hier in feite weer terug bij ons uitgangspunt uit hoofdstuk 4: we hebben nu een exacte definitie gegeven van het begrip *logische gevolgtrekking* (of: *geldige redenering*) voor de taal van de propositielogica.

De bovenstaande definitie van het begrip *logisch gevolg* maakt het mogelijk nog een iets andere parafrase te geven van het begrip *tautologie*. Een tautologie is een logisch gevolg van de lege premissenverzameling. Ga na dat deze parafrase klopt. Als φ een tautologie is, dan kunnen we dat zo opschrijven:

$$\models \varphi.$$

Immers, φ volgt dan logisch uit de lege premissenverzameling. 'φ is geen tautologie' noteren we wel als:

$$\not\models \varphi.$$

We zeggen in dit geval ook wel: formule φ is niet logisch geldig. Let op: $\not\models \varphi$ houdt nog niet in dat φ een contingente formule is. Dat moeten we uitdrukken door:

$$\not\models \varphi \text{ en } \not\models \neg\varphi.$$

We kunnen nu nog een keer met andere woorden zeggen wat het betekent dat twee formules φ en ψ logisch equivalent zijn: φ en ψ heten *logisch equivalent* als ze logisch gevolg zijn van elkaar, dat wil zeggen: als elke valuatie die φ waar maakt ook ψ waar maakt, en omgekeerd.

Gewoontegetrouw stappen we tot slot nog even over naar het waarheidstafelperspectief. Hoe kun je met behulp van waarheidstafels controleren of een bepaalde gevolgtrekking logisch geldig is? Heel eenvoudig (maar soms wel veel werk; in § 5.6 zullen we een andere methode presenteren): maak waarheidstafels voor alle premissen. Beschouw dan alleen die valuaties (rijen waarheidswaarden in de tafel) die overal een 1 hebben (dat wil zeggen: die alle premissen waar maken). Leveren al die valuaties ook voor de conclusie een 1 op, dan is de gevolgtrekking propositielogisch geldig, anders is ze dat niet.

We behandelen een voorbeeld. Is $((p \to q) \to (p \to r))$ een logisch gevolg van $(p \to (q \to r))$? Het antwoord is uit de volgende tafel af te lezen:

p	q	r	p	$\to$	$(q \to r)$	$(p \to q)$	$\to$	$(p \to r)$
1	1	1		1	1	1	1	1
0	1	1		1	1	1	1	1
1	0	1		1	1	0	1	1
0	0	1		1	1	1	1	1
1	1	0		0	0			
0	1	0		1	0	1	1	1
1	0	0		1	1	0	1	0
0	0	0		1	1	1	1	1

U ziet het: zelfs voor zo'n simpel geval als dit is het opstellen van de complete tafel een hele klus. De conclusie uit de tafel:

$$(p \to (q \to r)) \models ((p \to q) \to (p \to r)).$$

Dit wil zeggen: de redenering van $(p \to (q \to r))$ naar $((p \to q) \to (p \to r))$ is propositielogisch geldig. Alleen de valuaties die 'ertoe doen' zijn in de tafel helemaal uitgewerkt.

Opdracht 5.27 *Laat met behulp van de waarheidstafelmethode zien dat het redeneerpatroon Modus Ponens (dat wil zeggen: concludeer uit de premissen φ en $(\varphi \to \psi)$ tot ψ) propositielogisch geldig is.*

Opdracht 5.28 *Laat op dezelfde manier zien dat het patroon Modus Tollens (dat wil zeggen: concludeer uit de premissen $(\varphi \to \psi)$ en $\neg\psi$ tot $\neg\varphi$) propositielogisch geldig is.*

Opdracht 5.29 *Idem voor de volgende redeneerpatronen:*

1. *Concludeer uit $(\varphi \to \psi)$ en $(\psi \to \chi)$ tot $(\varphi \to \chi)$.*
2. *Concludeer uit $(\varphi \vee \psi)$ en $(\varphi \to \chi)$ tot $(\chi \vee \psi)$.*

Tot slot de waarheidstafelparafrase van *logische equivalentie* tussen formules: twee formules zijn logisch equivalent desda ze dezelfde waarheidstafel hebben.

Opdracht 5.30 *Laat zien met behulp van waarheidstafels dat de volgende formules per regel logisch equivalent zijn (weer: voor elke keuze van φ en ψ).*

1. *φ; $\neg\neg\varphi$*
2. *$\neg\varphi$; $\varphi \to (\psi \wedge \neg\psi)$*
3. *$\neg(\varphi \wedge \psi)$; $\neg\varphi \vee \neg\psi$*
4. *$\neg(\varphi \vee \psi)$; $\neg\varphi \wedge \neg\psi$.*

Drie van deze logische equivalenties hebben speciale namen: 1. heet de wet van de dubbele negatie; 3. en 4. zijn de zogenaamde wetten van De Morgan.

Deze laatste opdracht was bedoeld om u een zeker 'gevoel' bij te brengen voor de logische equivalentie van formules uit de propositielogica. Maar zoals gezegd: de waarheidstafelmethode is vaak nogal omslachtig als test voor logische equivalentie en logische geldigheid. In de volgende paragraaf zullen we kennis gaan maken met een testmethode die eleganter is.

5.6 Semantische tableaus voor de propositielogica

De semantische tableaumethode voor het testen van rederingen, die we nu gaan behandelen, is eleganter dan de methode van het controleren met behulp van waarheidstafels waarmee we hierboven hebben kennisgemaakt

omdat in het algemeen niet alle mogelijke valuaties hoeven te worden uitgewerkt. Deze methode heeft bovendien het voordeel dat zij kan worden uitgebreid tot een geldigheidstest voor predikatenlogische redeneringen (zie § 6.6). De waarheidstafelmethode voor het controleren van geldigheid is rechttoe, rechtaan: onderzoek gewoon alle mogelijkheden. Bij het maken van een waarheidstafel ter controle van een redenering

$$\varphi_1, \ldots, \varphi_n \models \psi$$

moeten we, als er m verschillende propositieletters voorkomen in de premissen en de conclusie, 2^m verschillende waarderingen (rijen in de waarheidstafel) gaan onderzoeken. Bij de tableaumethode kunnen we hier in veel gevallen op bezuinigen.

De semantische tableaumethode werkt niet direct maar *indirect*: wat er gebeurt is dat er systematisch wordt gezocht naar een *tegenvoorbeeld* tegen de redenering. De methode werkt als volgt: veronderstel dat de conclusie *onwaar* is, en dat alle premissen waar zijn. Wat moet er dan aan de hand zijn...? Aldus terugredenerend worden alle wegen die mogelijkerwijs leiden naar een tegenvoorbeeld uitgeplozen. Lukt het om zo'n tegenvoorbeeld te construeren (dat wil zeggen een valuatie te vinden die de premissen waar maakt en de conclusie onwaar), dan is de redenering kennelijk ongeldig. Lukt het niet dan mogen we, dankzij het feit dat het zoeken systematisch geschiedde, concluderen dat de redenering geldig is. Op dezelfde manier kunnen we tautologieën op het spoor komen, door het systematisch zoeken naar een valuatie die de kandidaat-tautologie *onwaar* maakt (een tautologie is immers een formule die logisch volgt uit de lege premissenverzameling).

We zullen de semantische tableaumethode introduceren aan de hand van voorbeelden. Om de voorbeelden te begrijpen moet u weten wat een semantisch tableau is: een plaatje dat is onderverdeeld in een linker- en een rechterkolom, met links en rechts verzamelingen formules. Links op elke regel staan de formules die waar moeten worden, rechts de formules die onwaar moeten worden (in het tegenvoorbeeld dat we proberen te construeren). Bij het testen van een redenering beginnen we met de premissen links te zetten en de conclusie rechts. Immers: in het tegenvoorbeeld waarnaar we op zoek zijn moeten de premissen waar worden en de conclusie onwaar. Hoe de verdere constructie van het tableau plaatsvindt moge blijken uit de voorbeelden die we nu gaan behandelen.

Voorbeeld 5.1 Is $p \wedge q \models p$?

Neem aan van niet:

waar	onwaar
$p \wedge q$	p

Het feit dat de formule $p \wedge q$ links in het tableau staat betekent dat die formule waar is, het feit dat p rechts staat betekent dat p onwaar is. Als $p \wedge q$ waar is, dan betekent dat dat zowel p als q waar moet zijn:

waar	onwaar
$p \wedge q$	p
p, q	

Dit kan echter niet: p staat nu zowel links als rechts in het tableau, en zou dus zowel waar als onwaar moeten zijn. Dit noteren we met een streep onderaan. We zeggen: het tableau *sluit*. De conclusie die we mogen trekken is dat het construeren van een tegenvoorbeeld tegen $p \wedge q \models p$ is mislukt. Met andere woorden: de redenering is geldig.

Voorbeeld 5.2 Is $p \vee q \models p$?

We beginnen het tableau weer als volgt:

waar	onwaar
$p \vee q$	p

Dat $p \vee q$ waar is betekent dat minstens een van de twee volgende mogelijkheden moet zijn gerealiseerd: hetzij p is waar, hetzij q. Het feit dat er twee mogelijkheden zijn geven we in het tableau aan door middel van splitsing:

waar $p \vee q$		onwaar p	
waar	onwaar	waar	onwaar
p		q	

Een van de twee voortzettingen leidt tot een onmogelijke situatie. Die voortzetting leidt tot sluiting. Dit geven we weer aan door middel van een streep onderaan. De andere voortzetting sluit niet. Die voortzetting levert een tegenvoorbeeld: q is waar, p is niet waar. Uit het bestaan van dit tegenvoorbeeld volgt dat de redenering ongeldig is.

We moeten er aan denken dat we, om een tableau te controleren op sluiting, steeds het hele tableau van boven naar beneden moeten doorzoeken. Bij een opsplitsing van een tableau in tweeën loopt het oorspronkelijke tableau door in twee 'takken': het gedeelte voor de splitsing plus de ene voortzetting, en het gedeelte voor de splitsing plus de tweede voortzetting. Wanneer een van de twee tableauontwikkelingen zich wederom splitst hebben we in totaal drie takken, enzovoorts. Met een *subtableau* bedoelen we vanaf nu een volledige 'tak' in het tableau. Een subtableau *sluit* als er een formule φ zowel ergens onder 'waar' als ergens onder 'onwaar' voorkomt in dat subtableau. Het hele tableau sluit als elk subtableau sluit.

Voorbeeld 5.3 Is $p \models p \wedge q$?

Stel van niet:

waar	onwaar
p	$p \wedge q$

Als $p \wedge q$ onwaar is, dan moet minstens een van de twee volgende voorwaarden vervuld zijn: p onwaar, of q onwaar. Voor het tableau levert dit op:

waar		onwaar	
p		$p \wedge q$	
waar	onwaar	waar	onwaar
	p		q

Tegenvoorbeeld: p waar, q onwaar. De redenering is niet geldig.

Voorbeeld 5.4 Is $p \models p \vee q$?

Stel van niet:

waar	onwaar
p	$p \vee q$

Dat $p \vee q$ onwaar is, wil zeggen dat p en q beide onwaar moeten zijn:

waar	onwaar
p	$p \vee q$
	p, q

Het tableau sluit, de redenering is geldig.

Voorbeeld 5.5 Is $\neg\neg p \models p$?

Stel van niet:

waar	onwaar
$\neg\neg p$	p

De negatie van een bewering is waar wanneer die bewering zelf onwaar is, en de negatie van een bewering is onwaar wanneer de bewering zelf waar is. Dit geeft de volgende ontwikkeling van het tableau:

waar	onwaar
$\neg\neg p$	p
	$\neg p$
p	

Het tableau sluit; de redenering is geldig.

In de voorbeelden zijn tot nu toe de volgende regels voor de behandeling van operatoren geïntroduceerd:

- $\wedge$-links:

waar	onwaar
$\varphi \wedge \psi$	
φ, ψ	

- $\wedge$-rechts:

waar		onwaar $\varphi \wedge \psi$	
waar	onwaar	waar	onwaar
	φ		ψ

- $\vee$-links:

waar $\varphi \vee \psi$		onwaar	
waar	onwaar	waar	onwaar
φ		ψ	

- $\vee$-rechts:

waar	onwaar
	$\varphi \vee \psi$
	φ, ψ

- $\neg$-links:

waar	onwaar
$\neg\varphi$	
	φ

- $\neg$-rechts:

waar	onwaar
	$\neg\varphi$
φ	

- Het recept voor het afsluiten van een subtableau:

waar	onwaar
⋮	
φ	⋮
⋮	φ

of

waar	onwaar
	⋮
⋮	φ
φ	⋮

  Dit wil zeggen: een *subtableau sluit* wanneer een formule φ zowel links als rechts in dat subtableau voorkomt. Let wel: een subtableau begint altijd helemaal bovenaan. Denk eraan dat φ geen atomaire formule hoeft te zijn. Een *tableau sluit* wanneer elk subtableau van dat tableau sluit.

We kunnen de bovenstaande regels gebruiken voor het construeren van tableaus voor redeneringen met $\wedge$, $\vee$ en $\neg$ als voegwoorden.

Voorbeeld 5.6 Is $\neg\varphi \vee \neg\psi \models \neg(\varphi \wedge \psi)$?

Stel van niet:

waar	onwaar
$\neg\varphi \vee \neg\psi$	$\neg(\varphi \wedge \psi)$

Het volledige tableau voor het testen van deze redenering ziet er als volgt uit:

<table>
<tr><td colspan="2">waar
$\neg\varphi \vee \neg\psi$
$\varphi \wedge \psi$
φ, ψ</td><td colspan="2">onwaar
$\neg(\varphi \wedge \psi)$</td></tr>
<tr><td>waar
$\neg\varphi$</td><td>onwaar

φ</td><td>waar
$\neg\psi$</td><td>onwaar

ψ</td></tr>
</table>

Alle subtableaus van het tableau sluiten: de redenering is geldig. Merk op dat we bij het construeren van dit semantisch tableau zelf in de hand hebben in welke *volgorde* we bepaalde regels willen toepassen. We hebben eerst $\neg$-rechts toegepast, vervolgens $\wedge$-links, daarna $\vee$-links, en tenslotte in beide subtableaus $\neg$-links. We hadden ook kunnen beginnen met $\vee$-links. Voor het eindresultaat maakt dat niet uit, maar in de praktijk is het handig om het toepassen van regels die tableau-splitsing tot gevolg hebben zo lang mogelijk uit te stellen.

Opdracht 5.31 *Laat met behulp van de semantische tableau methode zien dat de volgende formule-schema's regel voor regel logisch equivalent zijn:*

1. $\varphi \wedge (\psi \vee \chi)$; $(\varphi \wedge \psi) \vee (\varphi \wedge \chi)$
2. $\varphi \vee (\psi \wedge \chi)$; $(\varphi \vee \psi) \wedge (\varphi \vee \chi)$.

Dit zijn de zogenaamde wetten van de distributie*. Aanwijzing: om te laten zien dat twee formules φ en ψ logisch equivalent zijn moet u laten zien dat $\varphi \models \psi$* en *dat $\psi \models \varphi$.*

Opdracht 5.32 *Geef de regels voor a-links en a-rechts (a staat voor de exclusieve disjunctie).*

Wanneer we nu nog regels voor de behandeling van $\rightarrow$ en $\leftrightarrow$ invoeren is onze tableaumethode compleet. Hoe $\rightarrow$-links behandeld moet worden volgt direct uit de waarheidstafel-definitie: $\varphi \rightarrow \psi$ is waar wanneer hetzij φ onwaar is, hetzij ψ waar. Dus:

- $\rightarrow$-links:

waar $\varphi \rightarrow \psi$		onwaar	
waar	onwaar	waar	onwaar
	φ	ψ	

De motivering voor $\rightarrow$-rechts moet u nu zelf kunnen bedenken. Hier is de regel:

- $\rightarrow$-rechts:

waar	onwaar
	$\varphi \rightarrow \psi$
φ	ψ

Opdracht 5.33 *Ga met behulp van een tableau waarin de regel a-rechts wordt gebruikt die u in de opdracht 5.32 heeft geformuleerd na of een redenering volgens het volgende schema geldig is:*

$\varphi \vee \psi \models \varphi \ a \ \psi$.

Opdracht 5.34 *Geef de regels voor $\leftrightarrow$-links en $\leftrightarrow$-rechts.*

Opdracht 5.35 *Laat met behulp van de semantische tableaumethode zien dat de volgende formule-schema's regel voor regel logisch equivalent zijn:*

1. $\varphi \to (\psi \to \chi)$; $(\varphi \wedge \psi) \to \chi$
2. $\varphi \leftrightarrow \psi$; $(\varphi \to \psi) \wedge (\psi \to \varphi)$
3. $\varphi \to \psi$; $\neg\psi \to \neg\varphi$
4. $\varphi \to \neg\psi$; $\psi \to \neg\varphi$.

3. en 4. zijn de zogenaamde wetten van de contrapositie.

Dat de principes uit opdracht 5.35 in het redeneren een rol spelen blijkt bij voorbeeld uit het feit dat bij het *bewijzen* van een stelling van de vorm "Zus is het geval desda zo het geval is" er vaak wordt *uitgesplitst*. Er worden dan twee onderdelen bewezen:

- Ten eerste: als zus het geval is, dan is zo het geval.
- Ten tweede: als zo het geval is, dan is zus het geval.

Hier ligt natuurlijk de equivalentie uit onderdeel 2. van 5.35 aan ten grondslag.

Een principe dat bij het bewijzen van stellingen van de vorm "Als zus het geval is, dan is zo het geval" vaak wordt gebruikt is principe 3. uit 5.35. We kunnen dan contrapositie toepassen, en in plaats van de oorspronkelijke stelling bewijzen we: "Als zo *niet* het geval is, dan is zus *niet* het geval".

Opdracht 5.36 *Laat met behulp van de semantische tableaumethode zien dat de volgende formules tautologieën zijn:*

1. $(p \wedge (p \to q)) \to q$
2. $(\neg q \wedge (p \to q)) \to \neg p$
3. $((p \to q) \wedge (q \to r)) \to (p \to r)$
4. $p \to (p \to p)$
5. $(p \to (q \to r)) \to ((p \to q) \to (p \to r))$.

Aanwijzing: om met de tableaumethode te laten zien dat een formule φ een tautologie is dienen we te controleren of de aanname dat φ onwaar is tot een tegenspraak leidt. Dit houdt in dat we het tableau moeten beginnen door φ rechts in het tableau te zetten.

5.7 Axiomatiek

In de paragrafen hierboven hebben we het zogenaamde *semantische perspectief* op propositielogica uiteengezet. De semantische kijk op tautologieën karakteriseert een tautologie als een formule die waar is voor alle interpretaties van de propositieletters. In dezelfde geest geeft het semantische perspectief een karakterisering van logisch geldige redeneringen. In § 5.6 hebben we een gestroomlijnde semantische testmethode voor geldigheid en tautologie-zijn gepresenteerd. Naast het semantische perspectief op redeneren bestaat er nog een heel andere kijk, met een veel langere traditie, het zogenaamde *afleidingsperspectief*.

In het afleidingsperspectief wordt een aantal *axioma's* als uitgangspunt gekozen, en uit die axioma's worden met behulp van zogenaamde *afleidingsregels* nieuwe formules afgeleid, de zogenaamde *stellingen*. Bij het afleiden van een conclusie φ uit een verzameling premissen $\{\psi_1, \ldots, \psi_n\}$ is het nu de kunst om, met behulp van de afleidingsregels, φ af te leiden uit de axioma's plus de premissenverzameling.

Een verzameling axioma's plus een verzameling afleidingsregels heet een *deductief systeem* (of: een *axiomatiek*, een *axiomatische calculus*, of kortweg een *calculus*). We geven een deductief systeem S voor de propositielogische taal $\mathcal{T}$.

Definitie 5.11 *Formules van de volgende vormen zijn* **axioma's** *van de propositielogica.*

1. $\varphi \to (\psi \to \varphi)$
2. $(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$
3. $(\neg\varphi \to \neg\psi) \to (\psi \to \varphi)$.

Niets anders is een axioma.

Merk op dat er oneindig veel axioma's zijn.

Opdracht 5.37 *Ga na waarom de volgende formules axioma's zijn:*

1. $p \to (p \to p)$
2. $(p \to q) \to (p \to (p \to q))$
3. $(p \to (q \to p)) \to ((p \to q) \to (p \to p))$
4. $(p \to (q \to (p \to q))) \to ((p \to q) \to (p \to (p \to q)))$
5. $(\neg\neg p \to \neg(q \to r)) \to ((q \to r) \to \neg p)$.

We nemen aan dat het deductieve systeem slechts één enkele afleidingsregel kent. Die regel is *Modus Ponens*:

> *Als φ is afgeleid en $(\varphi \to \psi)$ is afgeleid, dan mag ook ψ worden afgeleid.*

We gaan nu een formele definitie geven van het begrip *afleiding* (Engels: *derivation*) in een deductief systeem. Iets preciezer gezegd: we definiëren het begrip "afleiding in deductief systeem S uit formuleverzameling Σ". Informeel komt de definitie neer op het volgende: een afleiding uit een formuleverzameling Σ is een eindig geordend rijtje formules, waarbij geldt: elke formule in het rijtje is een axioma, of een formule uit Σ, of een formule die met behulp van een afleidingsregel is verkregen uit voorafgaande formules in het rijtje. Een afleiding binnen S waarbij geen extra premissen worden gebruikt noemen we een *stelling* van S.

We zullen straks gaan demonstreren hoe je *in* en *over* het deductieve systeem S kunt redeneren, maar eerst voeren we wat nieuwe notatie in. De notatie voor "φ is afleidbaar uit formuleverzameling Σ" wordt:

$$\Sigma \vdash \varphi.$$

Wanneer we te maken hebben met een eindige verzameling Σ die bestaat uit formules $\{\psi_1, \ldots, \psi_n\}$ wordt ook wel de volgende notatie gebruikt

$$\psi_1, \ldots, \psi_n \vdash \varphi$$

in plaats van:

$$\{\psi_1, \ldots, \psi_n\} \vdash \varphi.$$

De notatie voor "φ is een stelling van het systeem" wordt:

$$\vdash \varphi.$$

Hier is een voorbeeld van een heel simpele afleiding binnen het zojuist gepresenteerde deductieve systeem S; de afleiding laat zien dat de formule r afleidbaar is uit de twee premissen $q \to r$ en $(p \to (p \to p)) \to q$.

Stelling 5.2 *Er geldt: $q \to r, (p \to (p \to p)) \to q \vdash r$.*

Bewijs:

1	$p \to (p \to p)$	[ax 1]
2	$(p \to (p \to p)) \to q$	[premisse]
3	q	[MP uit 1 en 2]
4	$q \to r$	[premisse]
5	r	[MP uit 3 en 4].

Bij elke formule uit het rijtje is aangegeven hoe we eraan komen: de eerste formule is een axioma volgens schema 1; de tweede formule is element van de premissen-verzameling; de derde is door Modus Ponens verkregen uit de eerste en de tweede; de vierde is weer een premisse; en de vijfde, de uiteindelijke conclusie, is verkregen door Modus Ponens uit de derde en de vierde. ∎

Nog een voorbeeld, om te laten zien dat p afleidbaar is uit de twee premissen $\neg p \rightarrow \neg q$ en q.

Stelling 5.3 *Er geldt:* $\neg p \rightarrow \neg q, q \vdash p$.

Bewijs:

1	$\neg p \rightarrow \neg q$	[premisse]
2	$(\neg p \rightarrow \neg q) \rightarrow (q \rightarrow p)$	[ax 3]
3	$q \rightarrow p$	[MP uit 1 en 2]
4	q	[premisse]
5	p	[MP uit 3 en 4].

Hiermee is de stelling bewezen. ∎

Het begrip *afleiding binnen een deductief systeem* schreeuwt natuurlijk om een recursieve definitie. We definiëren recursief de verzameling $\mathrm{AFL}_{S,\Sigma}$ van alle afleidingen binnen deductief systeem S uit formule-verzameling Σ.

Definitie 5.12 *De verzameling $AFL_{S,\Sigma}$ van alle afleidingen binnen deductief systeem S uit formule-verzameling Σ:*

- *Als φ een axioma is van S of een lid van Σ, dan is $\langle\varphi\rangle$ een lid van $AFL_{S,\Sigma}$.*
- *Als $\langle\psi_1, \ldots, \psi_n\rangle$ een lid is van $AFL_{S,\Sigma}$, en φ is een axioma van S, een lid van Σ, of een formule die met behulp van een afleidingsregel uit S verkregen is uit formules in $\{\psi_1, \ldots, \psi_n\}$, dan is $\langle\psi_1, \ldots, \psi_n, \varphi\rangle$ ook een lid van $AFL_{S,\Sigma}$.*
- *Niets anders is een lid van $AFL_{S,\Sigma}$.*

De formules die voorkomen als laatste element van een lid van $\mathrm{AFL}_{S,\Sigma}$ heten *afleidbaar* in S uit Σ. We kunnen Σ hier zien als een verzameling waaruit de premissen voor de afleiding mogen worden gekozen. Als Σ eindig is, is Σ dus een verzameling $\{\psi_1, \ldots, \psi_n\}$ van formules . In het algemeen hoeft Σ echter geen eindige verzameling te zijn.

Een speciaal geval hebben we wanneer Σ gelijk is aan de lege verzameling.

Definitie 5.13 *De formules die in een deductief systeem S afleidbaar zijn uit de lege verzameling heten de* **stellingen** *(Engels:* theorems*) van S.*

Met andere woorden: de stellingen van een deductief systeem S zijn alle formules die voorkomen als laatste element van zeker lid van $\mathrm{AFL}_{S,\emptyset}$. Merk op dat elk axioma van systeem S een stelling is van het systeem.

Definitie 5.14 *Een afleiding in S uit de lege verzameling heet een* **bewijs** *in S (Engels:* proof *in S).*

De bovenstaande definities waren voor willekeurige deductieve systemen. Vanaf nu concentreren we ons weer op het deductieve systeem voor de propositielogica waarvoor we hierboven de axioma's en de redeneerregel hebben gegeven. Overigens zijn er meerdere deductieve systemen voor de propositielogica mogelijk die equivalent zijn met het systeem dat wij hier hebben gepresenteerd, in de zin dat ze precies dezelfde verzameling stellingen opleveren.

Hier is een heel simpel voorbeeld van een bewijs van een stelling binnen ons deductieve systeem voor de propositielogica:

Stelling 5.4 *Er geldt:* $\vdash p \to (p \to p)$.

Bewijs:

$$1 \quad p \to (p \to p) \quad [\text{ax } 1].$$

Deze afleiding toont aan dat $p \to (p \to p)$ een stelling is in het deductieve systeem. ∎

Het systeem dat hierboven gegeven is, gebruikt een minimum aan axioma's en redeneerregels. Het geven van een zo zuinig systeem heeft als voordeel dat redeneren *over* het systeem gemakkelijk is. Het nadeel is dat het leveren van afleidingen *binnen* het systeem in eerste instantie wat lastiger is dan bij een systeem met meerdere afleidingsregels. Je moet nu eerst extra redeneerhulpmiddelen gaan fabriceren: iedere stelling die je hebt bewezen mag vanaf dat moment worden gebruikt als was het een axioma.

Opdracht 5.38 *Zij gegeven de premissenverzameling* $\{\neg\neg q \to \neg\neg p, q\}$. *Leid hieruit de formule p af. Met andere woorden: laat zien dat*

$$\neg\neg q \to \neg\neg p, q \vdash p.$$

Opdracht 5.39 *Toon aan:* $p, q, r, p \to (q \to (r \to s)) \vdash s$.

Opdracht 5.40 *Toon aan:* $\vdash (p \to q) \to (p \to p)$.

Het zo precies uitspellen van de definitie van *afleiding* (en daarmee samenhangend: die van *stelling*) als we hierboven gedaan hebben lijkt misschien pedant of anderszins overdreven, maar een recursieve formulering van het begrip *afleiding* is van belang wanneer we gaan redeneren *over* ons deductieve systeem: de recursieve definitie van *afleiding* maakt het mogelijk inductieve bewijzen te gaan leveren van zogenaamde *metastellingen* (Engels: metatheorems), stellingen *over* het deductieve systeem, die moeten worden onderscheiden van de stellingen *van* het systeem.

We geven nu eerst nog een voorbeeld van een redenering *binnen* het systeem. Een bewijs voor $\varphi \to \varphi$ gaat als volgt (schrikt u vooral niet):

Stelling 5.5 *In het systeem S voor de propositielogica geldt voor elke formule φ het volgende:* $\vdash \varphi \to \varphi$.

Bewijs:

1	$\varphi \to ((\varphi \to \varphi) \to \varphi)$	[ax 1, met $(\varphi \to \varphi)$ voor ψ]
2	$(\varphi \to ((\varphi \to \varphi) \to \varphi))$ $\to ((\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi))$	[ax 2, met $(\varphi \to \varphi)$ voor ψ]
3	$(\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi)$	[MP uit 1 en 2]
4	$\varphi \to (\varphi \to \varphi)$	[ax 1, met φ voor ψ]
5	$\varphi \to \varphi$	[MP uit 3 en 4].

Hiermee is het bewijs geleverd. ∎

Opmerking: dit is eigenlijk geen bewijs in het systeem S, maar een bewijs*stramien*; elke invulling van een formule voor φ levert een bewijs op (vandaar dat het verhaal voor elke formule φ opgaat). φ is weer een metavariabele.

Als u toch geschrokken bent van dit bewijs: zo'n afleiding construeer je natuurlijk niet van voor naar achter, maar *andersom*. Je constateert eerst dat $\varphi \to \varphi$ zelf geen axioma is, maar $\varphi \to (\varphi \to \varphi)$ wel. Dus zijn we klaar als we $(\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi))$ kunnen aantonen. Maar dat volgt weer uit het axioma

$$(\varphi \to ((\varphi \to \varphi) \to \varphi)) \to ((\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi))$$

plus het axioma

$$\varphi \to ((\varphi \to \varphi) \to \varphi)$$

en klaar.

Goed, dit was een laatste voorbeeld van een afleiding *binnen* het systeem. Nu een voorbeeld van redeneren *over* het systeem. We zullen een metastelling bewijzen, de zogenaamde *deductiestelling*. In het bewijs van

deze stelling wordt gebruik gemaakt van *inductie*, en wel als volgt. We laten eerst zien dat een bepaalde eigenschap geldt voor afleidingen van lengte 1 (dat wil zeggen afleidingen die uit een enkele formule bestaan; de afleiding van $p \to (p \to p)$ die we boven hebben gegeven is een voorbeeld). Dit is de basisstap in het inductie-bewijs. Daarna laten we zien dat *als* de eigenschap geldt voor afleidingen waarvan de lengte niet groter is dan n, *dan* geldt hij ook voor afleidingen van lengte $n+1$. Dat is de inductie-stap. Omdat de inductie-redenering betrekking heeft op de lengte van de afleiding noemen we dit: *inductie naar de lengte van een afleiding*. In § 5.4 hebben we kennisgemaakt met *inductie naar de complexiteit van een formule*, in het bewijs dat de welgevormde formules van een propositielogische taal in standaardnotatie evenveel linker- als rechter-haakjes hebben.

Stelling 5.6 (Deductiestelling) *Als* $\varphi \vdash \psi$*, dan* $\vdash \varphi \to \psi$*.*

Bewijs: We gebruiken inductie *naar de lengte van de afleiding* van ψ uit φ, dat wil zeggen van het bewijs van $\varphi \vdash \psi$.

- **basisstap:** De lengte van de afleiding is 1, dus: ofwel ψ is een axioma, of $\psi = \varphi$. In het eerste geval is

1	ψ	[axioma volgens gegeven]
2	$\psi \to (\varphi \to \psi)$	[ax 1]
3	$\varphi \to \psi$	[MP uit 1 en 2].

  de gevraagde afleiding van $\varphi \to \psi$, in het tweede geval moeten we $\varphi \to \varphi$ bewijzen. Dat hebben we in stelling 5.5 al gedaan, dus: klaar. Dit was het basisgeval.

- **inductiestap:** De inductiehypothese luidt: de bewering geldt voor alle paren φ, ψ waarbij de lengte van de afleiding van ψ uit φ niet groter is dan n. We bekijken nu een paar φ, ψ waarbij ψ in $n+1$ stappen uit φ is afgeleid. Er zijn drie mogelijkheden:

  1. ψ is een axioma. Bewijs nu $\varphi \to \psi$ als in de basisstap.
  2. $\psi = \varphi$. Bewijs $\varphi \to \psi$ als in de basisstap.
  3. ψ komt via MP uit eerdere χ, $\chi \to \psi$ in de afleiding.
     In dit laatste geval is het recept voor de afleiding van $\varphi \to \psi$ als volgt. Merk eerst op dat op χ en $\chi \to \psi$ de inductiehypothese van toepassing is. We hebben dus: $\vdash \varphi \to \chi$ en $\vdash \varphi \to (\chi \to \psi)$. De afleiding voor $\vdash \varphi \to \psi$ is nu:

$\vdots$	
$\varphi \to \chi$	[de afleiding van $\varphi \to \chi$]
$\vdots$	
$\varphi \to (\chi \to \psi)$	[de afleiding van $\varphi \to (\chi \to \psi)$]
$(\varphi \to (\chi \to \psi)) \to$	
$((\varphi \to \chi) \to (\varphi \to \psi))$	[ax 2]
$(\varphi \to \chi) \to (\varphi \to \psi)$	[MP uit voorafgaande formules]
$\varphi \to \psi$.	[nogmaals MP]

Dit is de gevraagde afleiding, en hiermee is de deductiestelling rond. ∎

Opdracht 5.41 *Lever een bewijs van het omgekeerde van de deductiestelling: "*als $\vdash \varphi \to \psi$ dan $\varphi \vdash \psi$*".*

De deductiestelling kan het redeneren binnen het systeem gemakkelijker maken omdat we haar mogen gebruiken als extra afleidingsregel. Voordat we dat gaan demonstreren roepen we u op om zelf een versterking van de deductiestelling te bewijzen:

Opdracht 5.42 *Werk het bewijs van de deductiestelling uit tot een bewijs voor een meer algemene variant van deze stelling die het volgende zegt:*

Als $\varphi_1, \ldots, \varphi_n, \varphi_{n+1} \vdash \psi$ *dan* $\varphi_1, \ldots, \varphi_n \vdash (\varphi_{n+1} \to \psi)$.

We zullen de versterkte versie van de deductiestelling gebruiken om de volgende stelling te bewijzen.

Stelling 5.7 $\varphi \to \psi, \psi \to \chi \vdash \varphi \to \chi$.

Bewijs: Om dit te laten zien hoeven we volgens de deductiestelling (versie uit opdracht 5.42) alleen het volgende aan te tonen:

$$\varphi \to \psi, \psi \to \chi, \varphi \vdash \chi.$$

Dit is simpel:

1	φ	[gegeven]
2	$\varphi \to \psi$	[gegeven]
3	ψ	[MP uit 1 en 2]
4	$\psi \to \chi$	[gegeven]
5	χ	[MP uit 3 en 4].

Hiermee is de stelling bewezen. ∎

5.8 Beslisbaarheid, correctheid, volledigheid

Het zogenaamde *beslisbaarheidsprobleem* voor de propositielogica is de volgende vraag: bestaat er een mechanische procedure die, voor een willekeurige formule φ van de propositielogica, kan uitmaken of die formule een tautologie is? We hebben niet precies omschreven wat een mechanische procedure is, maar we mogen het beslisbaarheidsprobleem hier wel even als volgt parafraseren: valt er een **Pascal**-computerprogramma te schrijven dat, nadat ik een willekeurige formule heb ingetikt, in een eindige hoeveelheid tijd kan uitmaken of die formule een tautologie is? Het antwoord is: ja. De waarheidstafelmethode en de semantische tableau-methode zijn methoden die met niet al te veel moeite zijn om te zetten in computerprogramma's: voor allebei deze werkwijzen geldt dat ze bestaan uit domweg regeltjes toepassen, en het gestelde probleem wordt beantwoord. Dus: de propositielogica is beslisbaar.

We komen nu toe aan de hamvraag die kan worden gesteld met betrekking tot het semantische perspectief en het afleidingsperspectief op propositielogisch redeneren: dekken de centrale begrippen uit het eerste perspectief (logisch gevolg) en die uit het tweede perspectief (afleidbaar uit) elkaar? Ofwel: dekken $\models$ en $\vdash$ elkaar? Deze vraag valt in twee onderdelen uiteen:

- **Ten eerste:** geldt *als* $\psi_1, \ldots, \psi_n \vdash \varphi$ *dan* $\psi_1, \ldots, \psi_n \models \varphi$?

Dit is de vraag naar de *correctheid* van het deductieve systeem. Een systeem is correct als geen enkele gevolgtrekking waarvoor in het systeem een afleiding kan worden gevonden logisch ongeldig blijkt te zijn.

Dat het in § 5.7 gegeven deductieve systeem voor de propositielogica correct is, is niet zo moeilijk te bewijzen. Gebruik waarheidstafels of semantische tableaus om te laten zien dat alle axioma's tautologieën zijn, en dus ook: logische gevolgen van elke premissenverzameling. Toon vervolgens aan dat Modus Ponens de eigenschap van logisch-gevolg-zijn van een gegeven premissenverzameling bewaart.

Opdracht 5.43 *Werk deze schets van het bewijs van de correctheid van ons deductief systeem voor de propositielogica uit.*

Correctheid is heel handig wanneer we willen aantonen dat $\psi_1, \ldots, \psi_n \nvdash \varphi$. Het feit dat u of wij er niet in slagen om een afleiding van φ uit $\psi_1, \ldots, \psi_n$ te vinden bewijst immers niet er niet zo'n afleiding bestaat ... Maar een tegenvoorbeeld voor $\psi_1, \ldots, \psi_n \models \varphi$ is in het algemeen snel gevonden. En *correctheid* zegt dan dat uit $\psi_1, \ldots, \psi_n \not\models \varphi$ volgt dat $\psi_1, \ldots, \psi_n \nvdash \varphi$.

Er is echter nog een tweede onderdeel van de vraag naar de verhouding tussen $\vdash$ en $\models$.

- **Ten tweede:** geldt *als* $\psi_1, \ldots, \psi_n \models \varphi$, *dan* $\psi_1, \ldots, \psi_n \vdash \varphi$?

Dit is de vraag naar de *volledigheid* van het deductieve systeem: is het deductieve systeem rijk genoeg om voor elke willekeurige logisch geldige gevolgtrekking een afleiding te kunnen produceren?

Een eerste kleine hobbel die we moeten nemen heeft te maken met het feit dat de axioma's in de boven gepresenteerde propositielogische calculus alleen gebruik maken van de voegwoorden $\neg$ en $\rightarrow$, terwijl de afleidingsregel Modus Ponens bij de axioma's aansluit door alleen gebruik te maken van het voegwoord $\rightarrow$. Een en ander betekent dat alleen formules kunnen worden afgeleid die alleen de voegwoorden $\neg$ en $\rightarrow$ bevatten. Het gebruik van alleen $\neg$ en $\rightarrow$ heeft voordelen voor het redeneren over de calculus: bij inductiebewijzen naar de complexiteit van formules hoeven we in de inductiestap maar twee gevallen te bekijken. Aan de andere kant is het handig om toch de beschikking te hebben over de andere voegwoorden, met name als we de brug tussen axiomatiek en semantiek willen slaan. De formules waarvan in de volledigheidsstelling sprake is kunnen immers de voegwoorden $\wedge$, $\vee$ en $\leftrightarrow$ bevatten.

Om hier een mouw aan te passen kunnen we formules die voorkomens van $\wedge$, $\vee$ en $\leftrightarrow$ bevatten beschouwen als *afkortingen* voor formules met alleen $\neg$ en $\rightarrow$. We spreken af:

- $\varphi \wedge \psi$ is een afkorting voor $\neg(\varphi \rightarrow \neg\psi)$.
- $\varphi \vee \psi$ is een afkorting voor $\neg\varphi \rightarrow \psi$.
- $\varphi \leftrightarrow \psi$ is een afkorting voor $\neg((\varphi \rightarrow \psi) \rightarrow \neg(\psi \rightarrow \varphi))$.

U kunt gemakkelijk met behulp van waarheidstafels nagaan dat deze afkortingen de juiste betekenissen toekennen.

Ook nadat deze eerste hobbel is weggewerkt is de volledigheidsvraag veel moeilijker te beantwoorden dan de correctheidsvraag. De rest van deze paragraaf is eraan gewijd, maar wie de details bij eerste lezing wil overslaan heeft onze zegen. Wie tot volledig begrip van wat nu volgt wil geraken raden wij aan om de rest van de paragraaf niet alleen van voren naar achteren maar ook van achteren naar voren te lezen. Deze werkwijze maakt duidelijk wat de motivering is van de hulpstellingen die tot het uiteindelijke resultaat leiden.

We beginnen met het toepassen van *contrapositie*. Dit wil zeggen, we zetten de volledigheidsvraag om in:

- Geldt *als* $\psi_1, \ldots, \psi_n \nvdash \varphi$, *dan* $\psi_1, \ldots, \psi_n \not\models \varphi$?

Om conclusies te kunnen trekken uit het feit dat een bepaalde formule *niet* afleidbaar is uit een gegeven stel premissen hebben we een flinke omweg

nodig. Noem de premissenverzameling Γ. Wat we willen is een waardering vinden voor de formules van de taal die alle premissen waar maakt maar de conclusie φ onwaar. De weg die we zullen bewandelen om zo'n waardering te vinden is als volgt. We breiden de premissenverzameling eerst uit met de negatie van de conclusie. Dit geeft de verzameling $\Gamma \cup \{\neg\varphi\}$. Deze verzameling breiden we vervolgens verder uit tot een verzameling Γ^* van formules die de volgende eigenschap heeft: $\psi \in \Gamma^*$ desda $\Gamma^* \vdash \psi$. Wanneer dan ook nog uit de constructie van Γ^* volgt dat er een valuatie V is die precies de formules uit Γ^* waar maakt zijn we klaar. Die V zal de premissen uit de oorspronkelijke premissenverzameling waar maken en de conclusie onwaar, waarmee is aangetoond dat $\psi_1, \ldots, \psi_n \not\models \varphi$.

We voeren een paar nieuwe begrippen in. Allereerst het begrip *consistentie*.

Definitie 5.15 *Een verzameling formules* Γ *heet* **consistent** *wanneer er geen formule* φ *is waarvoor zowel* $\Gamma \vdash \varphi$ *als* $\Gamma \vdash \neg\varphi$.

Een voorbeeld van een consistente verzameling formules is $\{p, p \vee q, \neg q\}$. Een formuleverzameling die niet consistent is heet *inconsistent*; $\{p, \neg p\}$ is een voorbeeld van een inconsistente formuleverzameling.

Opdracht 5.44 *Laat zien: als een formuleverzameling* Γ *inconsistent is, dan is elke* Γ' *zo dat* $\Gamma \subseteq \Gamma'$ *eveneens inconsistent.*

We bewijzen nu een paar stellingen die ons iets meer vertellen over het zojuist gedefinieerde begrip *consistentie*.

Stelling 5.8 *Een formuleverzameling* Γ *is consistent desda er een formule* φ *te vinden is zodat* $\Gamma \nvdash \varphi$.

Bewijs: De stelling bestaat uit twee onderdelen. Ten eerste: als Γ consistent is, dan is er een φ zodat $\Gamma \nvdash \varphi$. Ten tweede: als $\Gamma \nvdash \varphi$ dan is Γ consistent.

Het eerste is gemakkelijk. Als Γ consistent is, dan is er per definitie geen φ te vinden zo dat zowel φ zelf als zijn negatie afleidbaar is uit Γ. Neem dus een willekeurige formule φ. Ofwel $\Gamma \nvdash \varphi$, en klaar, ofwel: $\Gamma \vdash \varphi$ en daaruit volgt dat $\Gamma \nvdash \neg\varphi$, en ook klaar.

Nu de andere richting. Stel er is een φ zodat $\Gamma \nvdash \varphi$. We moeten nu laten zien dat Γ consistent is, dat wil zeggen dat er geen ψ is waarvoor $\Gamma \vdash \psi$ en $\Gamma \vdash \neg\psi$. Veronderstel even dat er wel zo'n ψ is: we hebben dan $\Gamma \vdash \psi$ en $\Gamma \vdash \neg\psi$. Als we nu kunnen laten zien dat $\Gamma \vdash \neg\psi \rightarrow (\psi \rightarrow \varphi)$, dan hebben we ook $\Gamma \vdash \varphi$ (tweemaal Modus Ponens), en dus een tegenspraak met het gegeven dat $\Gamma \nvdash \varphi$. We mogen dan concluderen dat de aanname dat Γ inconsistent is niet klopt, en we zijn klaar. (Dit is weer een *reductio ad absurdum*.)

We laten nu zien dat

$$\vdash \neg\psi \to (\psi \to \varphi)$$

en dus zeker:

$$\Gamma \vdash \neg\psi \to (\psi \to \varphi).$$

In stelling 5.7 hebben we bewezen dat

$$\varphi \to \psi, \psi \to \chi \vdash \varphi \to \chi.$$

We gebruiken dit hulpresultaat in de verlangde afleiding:

1	$\neg\psi \to (\neg\varphi \to \neg\psi)$	[ax 1]
2	$(\neg\varphi \to \neg\psi) \to (\psi \to \varphi)$	[ax 2]
3	$\neg\psi \to (\psi \to \varphi)$	[hulpresultaat toegepast op 1 en 2].

En klaar. Hiermee is de stelling bewezen. ∎

Opdracht 5.45 *Laat met behulp van stelling 5.8 zien dat de verzameling van alle propositieletters van een propositielogische taal $\mathcal{T}$ consistent is.*

Stelling 5.9 *Als $\Gamma \nvdash \varphi$ dan is $\Gamma \cup \{\neg\varphi\}$ consistent.*

Bewijs: Neem aan dat $\Gamma \nvdash \varphi$. Dan is Γ kennelijk consistent (stelling 5.8). We doen nu weer een *reductio ad absurdum*. Veronderstel dat $\Gamma \cup \{\neg\phi\}$ inconsistent is. Dan volgt uit stelling 5.8 dat

$$\Gamma \cup \{\neg\varphi\} \vdash \neg\alpha$$

voor een willekeurig propositielogisch axioma α. Toepassen van de deductiestelling levert nu: $\Gamma \vdash \neg\varphi \to \neg\alpha$. Hieruit volgt, met behulp van axiomaschema 3: $\Gamma \vdash \alpha \to \varphi$. Maar α was een axioma, dus we hebben zeker $\Gamma \vdash \alpha$, zodat we met Modus Ponens kunnen concluderen: $\Gamma \vdash \varphi$. Dit is in tegenspraak met het gegeven $\Gamma \nvdash \varphi$. Hiermee is de veronderstelling dat $\Gamma \cup \{\neg\varphi\}$ inconsistent is weerlegd. ∎

Definitie 5.16 *We noemen een formuleverzameling* Γ **maximaal consistent** *wanneer* Γ *de volgende twee eigenschappen heeft:*

- Γ *is consistent.*
- *Als* $\Gamma \subset \Gamma'$*, dan is* Γ' *inconsistent.*

Een maximaal consistente verzameling formules is een consistente verzameling formules waaraan geen formules meer kunnen worden toegevoegd zonder de verzameling inconsistent te maken.

Opdracht 5.46 *Laat zien: als V een waardering is voor de formules uit $\mathcal{T}$, dan is $\{\varphi \in \mathcal{T} \mid V(\varphi) = 1\}$ een maximaal consistente verzameling.*

Stelling 5.10 (Lemma van Lindenbaum) *Elke consistente verzameling Γ van propositielogische formules is bevat in een maximaal consistente formuleverzameling Γ^*.*

Bewijs: Elke propositielogische taal $\mathcal{T}$ heeft aftelbaar veel formules. We mogen daarom aannemen dat we beschikken over een aftelling

$$\varphi_0, \varphi_1, \varphi_2, \varphi_3, \ldots$$

van alle formules van de taal. We definiëren nu, uitgaande van Γ, een rij van verzamelingen Γ_i met de eigenschap dat voor alle i: $\Gamma_i \subseteq \Gamma_{i+1}$. De definitie gaat als volgt:

- $\Gamma_0 = \Gamma$.
- $\Gamma_{i+1} = \begin{cases} \Gamma_i \cup \{\varphi_i\} & \text{als } \Gamma_i \cup \{\varphi_i\} \text{ consistent is.} \\ \Gamma_i & \text{anders.} \end{cases}$

Tenslotte nemen we de limiet van deze rij door al deze formuleverzamelingen op één hoop te vegen:

- $\Gamma^* = \bigcup_i \Gamma_i$.

Door inductie naar i is nu gemakkelijk in te zien dat elk van de verzamelingen Γ_i consistent is. Maar dan is ook Γ^* consistent. Stel immers van niet. Dan is er een formule ψ zo dat zowel $\Gamma^* \vdash \psi$ als $\Gamma^* \vdash \neg\psi$. Elk van deze afleidingen gebruikt slechts eindig veel premissen. Dus is er een getal m zo dat alle premissen die in een van beide afleidingen gebruikt worden in Γ_m zitten. Maar dan hebben we: $\Gamma_m \vdash \psi$ en $\Gamma_m \vdash \neg\psi$, met andere woorden Γ_m is inconsistent: een tegenspraak met wat we reeds hadden aangetoond.

Tenslotte geldt dat Γ^* maximaal consistent is. Neem een willekeurige formule ψ zo dat $\psi \notin \Gamma^*$ en beschouw de verzameling $\Gamma^* \cup \{\psi\}$. Formule ψ zit in onze aftelling van de formules van de taal, dus er is een k zo dat $\psi = \varphi_k$. Beschouw nu de stap in het constructieproces van Γ^* waar Γ_{k+1} werd gevormd. We weten dat $\varphi_k \notin \Gamma^*$, en dat kan alleen betekenen dat toevoeging van φ_k aan Γ_k de verzameling inconsistent zou hebben gemaakt. Maar dan is $\Gamma^* \cup \{\varphi_k\}$ eveneens inconsistent, want $\Gamma_k \subseteq \Gamma^*$. ∎

We bewijzen nu dat maximaal consistente verzamelingen de eigenschappen hebben die nodig zijn voor het volledigheidsresultaat.

Stelling 5.11 *Als Γ een maximaal consistente formuleverzameling is dan geldt voor elke formule φ: $\varphi \in \Gamma$ desda $\Gamma \vdash \varphi$.*

Bewijs: Uiteraard geldt dat als $\varphi \in \Gamma$ dan $\Gamma \vdash \varphi$. Om het omgekeerde te bewijzen nemen we aan dat $\varphi \notin \Gamma$. Uit de maximaliteit van Γ volgt dat $\Gamma \cup \{\varphi\}$ inconsistent is. Als nu $\Gamma \vdash \varphi$ dan volgt dat ook Γ inconsistent is, in tegenspraak met het gegeven omtrent Γ. Dus $\Gamma \nvdash \varphi$. ∎

Stelling 5.12 *Als Γ een maximaal consistente formuleverzameling is dan geldt voor elke formule φ: $\varphi \in \Gamma$ of $\neg\varphi \in \Gamma$.*

Bewijs: Stel voor een willekeurige formule φ dat $\neg\varphi \notin \Gamma$. Omdat Γ *maximaal* consistent is moet dus $\Gamma \cup \{\neg\varphi\}$ inconsistent zijn. Vanwege stelling 5.9 is derhalve $\Gamma \vdash \varphi$. Met stelling 5.11 volgt dat dan $\varphi \in \Gamma$. ∎

Merk op dat we deze stelling ook anders mogen formuleren: als Γ maximaal consistent is geldt dat $\varphi \in \Gamma$ desda $\neg\varphi \notin \Gamma$.

Stelling 5.13 *Als Γ een maximaal consistente formuleverzameling is, dan geldt: $\varphi \to \psi \in \Gamma$ desda (als $\varphi \in \Gamma$ dan $\psi \in \Gamma$).*

Bewijs: Neem aan dat Γ maximaal consistent is en dat $\varphi \to \psi \in \Gamma$ en $\varphi \in \Gamma$. Eenmaal toepassen van Modus Ponens geeft: $\Gamma \vdash \psi$, en met stelling 5.11 kunnen we concluderen dat $\psi \in \Gamma$.

Omgekeerd, neem aan dat Γ maximaal consistent is, en dat we weten dat voor een tweetal formules φ, ψ geldt: als $\varphi \in \Gamma$ dan $\psi \in \Gamma$. Twee mogelijkheden: ofwel ψ zit in Γ, en dan hebben we $\Gamma \vdash \varphi \to \psi$ (gebruik axioma $\psi \to (\varphi \to \psi)$ plus Modus Ponens), en dus $\varphi \to \psi \in \Gamma$ met stelling 5.11, ofwel φ, ψ geen van beide in Γ, en dan hebben we $\neg\varphi \in \Gamma$ met stelling 5.12, en opnieuw: $\Gamma \vdash \varphi \to \psi$ (gebruik axioma $\neg\varphi \to (\neg\psi \to \neg\varphi)$, pas Modus Ponens toe, en gebruik axiomaschema 3 plus nogmaals Modus Ponens om $\varphi \to \psi$ af te leiden), zodat ook in dit geval $\varphi \to \psi \in \Gamma$. ∎

We weten nu genoeg om te kunnen laten zien dat maximaal consistente verzamelingen geschikt zijn om de brug te slaan tussen axiomatiek en semantiek.

Stelling 5.14 *Als Γ een consistente verzameling formules is, dan is er een waardering V zo dat voor elke $\varphi \in \Gamma$: $V(\varphi) = 1$.*

Bewijs: Als Γ consistent is dan is er, zoals we hierboven hebben bewezen, een maximaal consistente Γ^* zo dat $\Gamma \subseteq \Gamma^*$. Neem nu de waardering V die gebaseerd is op de volgende interpretatie I voor de propositieletters van de taal $\mathcal{T}$: $I(p) = 1$ desda $p \in \Gamma^*$.

We laten zien dat voor de aldus gedefinieerde V geldt: $V(\varphi) = 1$ desda $\varphi \in \Gamma^*$. We gebruiken inductie naar de complexiteit van de formule φ.

- Basisstap: wanneer φ atomair geldt de bewering per definitie.

- Inductiestap:
  - Stel dat φ de vorm $\neg\psi$ heeft. Dan hebben we: $V(\varphi) = 1$ desda $V(\psi) = 0$ desda (inductiehypothese) $\psi \notin \Gamma^*$ desda (stelling 5.12) $\varphi \in \Gamma$.
  - Als φ de vorm $\psi \rightarrow \chi$ heeft hebben we: $V(\varphi) = 0$ desda ($V(\psi) = 1$ en $V(\chi) = 0$) desda (inductiehypothese) ($\psi \in \Gamma^*$ en $\chi \notin \Gamma^*$) desda (stelling 5.13) $\varphi \notin \Gamma^*$.

Omdat $\Gamma \subseteq \Gamma^*$ hebben we: als $\varphi \in \Gamma$ dan $V(\varphi) = 1$. ∎

De volledigheidsstelling volgt nu onmiddellijk:

Stelling 5.15 (Volledigheid van de propositielogica) *Als* $\Gamma \nvdash \varphi$, *dan* $\Gamma \not\models \varphi$.

Bewijs: Neem aan dat $\Gamma \nvdash \varphi$. Uit het feit dat formule φ niet uit Γ afleidbaar is volgt dat Γ consistent is, en $\Gamma \cup \{\neg\varphi\}$ ook. Volgens stelling 5.14 is er nu een waardering V die alle formules in $\Gamma \cup \{\neg\varphi\}$ waar maakt. Deze waardering V levert een tegenvoorbeeld waarin alle formules uit Γ waar zijn terwijl φ niet waar is. ∎

De volledigheidsstelling voor de propositielogica werd in 1920 bewezen door de logicus E. Post. De methode die wij hebben gevolgd is van L. Henkin.

Dankzij de volledigheidsstelling weten we dat het moeizame zoeken van afleidingen in de propositielogische calculus kan worden ingeruild voor het construeren van waarheidstafels of semantische tableaus.

5.9 Functionele volledigheid

Een leuke vraag die je over de voegwoorden van de propositielogica kunt stellen is deze. Welke connectieven kunnen in termen van welke andere worden gedefinieerd? Een voorbeeld: het invoeren van een speciaal teken a voor het exclusieve of is niet nodig, omdat $\varphi\ a\ \psi$ kan worden weergegeven als $\neg(\varphi \leftrightarrow \psi)$. In § 5.8 hebben we gezien hoe $\wedge$, $\vee$ en $\leftrightarrow$ kunnen worden uitgedrukt in termen van $\rightarrow$ en $\neg$ alleen.

Op deze manier kun je nog meer connectieven wegwerken: $\leftrightarrow$ is definieerbaar in termen van $\rightarrow$ en $\wedge$; $\rightarrow$ is definieerbaar met behulp van $\neg$ en $\vee$. Een verzameling connectieven in termen waarvan je alle waarheidstafelpatronen kunt uitdrukken heet *functioneel volledig*. Iets formeler:

Definitie 5.17 *Een verzameling connectieven C heet* **functioneel volledig** *desda er voor elk natuurlijk getal $n > 0$ en voor elke functie*

$$f : \{0,1\}^n \rightarrow \{0,1\}$$

een formule φ is met n propositieletters die alleen de connectieven uit C bevat en die het waarheidstafelpatroon van f heeft.

Stelling 5.16 *De verzameling $\{\wedge, \vee, \neg\}$ is functioneel volledig.*

Bewijs: We leveren het bewijs aan de hand van een voorbeeld. Laat f een functie van $\{0,1\}^2$ naar $\{0,1\}$ zijn met bij voorbeeld als functievoorschrift:

$$\begin{aligned} f(1,1) &= 1 \\ f(0,1) &= 1 \\ f(1,0) &= 0 \\ f(0,0) &= 0. \end{aligned}$$

We construeren nu een formule φ die twee propositieletters p en q bevat en als waarheidstafel het patroon van f heeft, dat wil zeggen:

	p	q	φ
I_1	1	1	1
I_2	0	1	1
I_3	1	0	0
I_4	0	0	0

De formule φ mag volgens de spelregels alleen de voegwoorden $\wedge$, $\vee$ en $\neg$ bevatten. Merk op dat een argumentenpaar van f overeenkomt met een *interpretatie* van p en q. Zo'n interpretatie kan worden omschreven met behulp van een simpele conjunctie: voor de eerste interpretatie, I_1, is de omschrijving $p \wedge q$; I_2 kan worden omschreven als $\neg p \wedge q$; I_3 als $p \wedge \neg q$; I_4 als $\neg p \wedge \neg q$. Alleen I_1 en I_2 maken φ waar, dus we kunnen voor φ de disjunctie nemen van de corresponderende conjuncties: $(p \wedge q) \vee (\neg p \wedge q)$. U gelieve zelf na te gaan dat deze formule het gewenste waarheidstafelpatroon heeft.

Het is nu duidelijk hoe we voor elke gegeven tweeplaatsige waarheidsfunctie f een geschikte φ vinden: neem de disjunctie van de formules die horen bij de interpretaties die 1 moeten opleveren. In één geval werkt dit procédé niet: het geval dat f louter nullen als waarden heeft. In dat geval voldoet de formule $(p \wedge \neg p) \vee (q \wedge \neg q)$ (ga dit na).

Het niet moeilijk om in te zien hoe de geschetste methode kan worden toegepast voor willekeurige waarden van n. Als $n = 1$ dan hebben we één propositieletter p nodig, en kunnen we de interpretaties beschrijven met de formules p en $\neg p$. Als $n = 3$ dan zijn er drie propositieletters p, q en r nodig, en acht conjuncties om de acht mogelijke interpretaties te beschrijven: $p \wedge q \wedge r$, $\neg p \wedge q \wedge r$, $p \wedge \neg q \wedge r$, enzovoort. In het algemeen: gebruik n propositieletters en 2^n formules die elk een conjunctie zijn van (negaties van) elk van die n propositieletters, en neem voor φ de disjunctie

over alle conjuncten die waar opleveren; als zulke conjuncten er niet zijn, neem dan de disjunctie van $p \wedge \neg p$, voor alle propositieletters p. ∎

Opdracht 5.47 *Laat met behulp van stelling 5.16 zien dat* $\{\wedge, \neg\}$, $\{\rightarrow, \neg\}$, $\{\vee, \neg\}$ *functioneel volledig zijn.*

Opdracht 5.48 *Laat zien dat* $\{\wedge, \vee\}$ niet *functioneel volledig is. Aanwijzing: u hoeft alleen te laten zien dat negatie niet uit te drukken valt.*

Opdracht 5.49 *Laat zien dat* $\{\neg, \leftrightarrow\}$ *niet functioneel volledig is.*

Een voordeel van $\vee$ boven a is nu dat het stel $\{\vee, \neg\}$ functioneel volledig is, terwijl $\{a, \neg\}$ dat niet is.

Leuk is dat het zelfs met een enkel connectief kan. De volgende connectieven kunnen het allebei (ze heten respectievelijk—naar hun uitvinders—de Quine dolk '†' en de Sheffer streep '|'):

†	1	0
1	0	0
0	0	1

|	1	0
1	0	1
0	1	1

Zoals u uit deze waarheidstafels kunt aflezen betekent $\varphi \dagger \psi$ 'noch φ noch ψ', terwijl $\varphi | \psi$ staat voor 'niet zowel φ als ψ'. Een voorbeeld van omschrijving: $\neg\varphi$ wordt met behulp van de Quine dolk $\varphi \dagger \varphi$, en met behulp van de Sheffer streep: $\varphi \mid \varphi$.

Opdracht 5.50 *Omschrijf de connectieven* $\wedge$, $\vee$, $\rightarrow$ *en* $\leftrightarrow$ *met behulp van alleen* † *en met behulp van alleen* | .

Opdracht 5.51 *Geef de semantische tableauregels voor de Quine dolk* † *en de Sheffer streep* | .

5.10 Variaties en uitbreidingen

We besluiten dit hoofdstuk met een paar opmerkingen over variaties op en uitbreidingsmogelijkheden van de propositielogica. In zekere zin is de predikatenlogica (zie volgend hoofdstuk) een uitbreiding van de propositielogica. De predikatenlogica gebruikt dezelfde waarheidsfunctionele connectieven als de propositielogica, en kan dus alle werk aan waar de propositielogica voor geschikt is, en met gemak.

In plaats van de voegwoord-analyse uit de propositielogica over te nemen (wat in de predikatenlogica gebeurt) zouden we ook kunnen overwegen die voegwoord-analyse te *herzien*. We stippen alleen een paar mogelijkheden aan.

We zouden naast de waarheidswaarden 1 en 0 ook een waarde # voor 'onbepaald' kunnen toelaten. Als we # lezen als 'nog niet bekend', dan liggen de volgende waarheidstafels voor de hand:

	$\neg$
1	0
0	1
#	#

$\wedge$	1	0	#
1	1	0	#
0	0	0	0
#	#	0	#

$\vee$	1	0	#
1	1	1	1
0	1	0	#
#	1	#	#

$\rightarrow$	1	0	#
1	1	0	#
0	1	1	1
#	1	#	#

$\leftrightarrow$	1	0	#
1	1	0	#
0	0	1	#
#	#	#	#

Dit is echter niet de enige interpretatiemogelijkheid voor #. We zouden # ook kunnen opvatten als 'onzinnig'. Dit levert de volgende tafels (een onzinnig onderdeel maakt de hele formule onzinnig):

	$\neg$
1	0
0	1
#	#

$\wedge$	1	0	#
1	1	0	#
0	0	0	#
#	#	#	#

$\vee$	1	0	#
1	1	1	#
0	1	0	#
#	#	#	#

$\rightarrow$	1	0	#
1	1	0	#
0	1	1	#
#	#	#	#

$\leftrightarrow$	1	0	#
1	1	0	#
0	0	1	#
#	#	#	#

Een propositielogisch systeem dat naast 1 en 0 ook nog een derde waarde # toelaat heet een *driewaardige propositielogica*. Driewaardige logica's worden wel gebruikt om het zogenaamde *presuppositie*-begrip—waar linguïsten in geïnteresseerd zijn—te behandelen. Zie voor meer informatie: [Gamut 1982], deel 1, hoofdstuk 5.

Bij de overstap van een tweewaardig naar een driewaardig systeem wordt de eis van functionele volledigheid voor stellen voegwoorden zwaarder. Immers, we moeten nu alle mogelijke functies van $\{0, 1, \#\}^n$ naar $\{0, 1, \#\}$ kunnen uitdrukken. Als u wilt kunt u zelf nagaan dat het stel $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ onder geen van beide driewaardige lezingen die we zojuist hebben gegeven functioneel volledig is.

De overgang van een tweewaardig naar een meerwaardig propositielogisch systeem komt neer op een variatie waarbij gesleuteld wordt aan de manier waarop de zinsoperatoren worden opgevat. We kunnen ook besluiten nieuwe zinsoperatoren toe te voegen. Zo kan de stap worden gezet van

gewone propositielogica naar propositionele *tijds*logica, naar *modale* propositielogica, of naar een combinatie van beide.

Modale propositielogica ontstaat door aan de gewone propositielogica operatoren $\Box$ ('het is noodzakelijk dat') en $\Diamond$ ('het is mogelijk dat') toe te voegen. We kunnen nu formules maken als de volgende:

- $\Box p$
- $\neg\Diamond\neg p$
- $\Box p \rightarrow p$
- $p \rightarrow \Diamond p$.

De intuïtie achter de nieuwe operatoren is: we willen *noodzakelijkheid* uitleggen op de manier van de filosoof Georg Wilhelm Leibniz (1646–1716). Leibniz legde uit: iets is noodzakelijk waar als het waar is *in alle mogelijke werelden*. In termen van de propositielogische semantiek:

Definitie 5.18 $\Box\varphi$ *is* **waar** *wanneer* φ *waar is in alle werelden* w*, waarbij een wereld* w *niets anders is dan een klassieke interpretatie voor de propositielogische variabelen.*

Een interpretatie voor modale predikatenlogica heeft dus een verzameling W van mogelijke werelden of contexten van node, en dient in *elke* wereld een waarheidswaarde toe te kennen aan alle propositieletters. Een *paar* bestaande uit een werelden-verzameling W en een interpretatiefunctie I die voor elke wereld een waarheidswaarden-toekenning voor de variabelen oplevert heet een *Kripke-model* (naar de logicus Saul Kripke die zulke modellen in de jaren 50 voor het eerst voorstelde). Zie [Gamut 1982], deel 2, voor meer informatie.

De uitbreiding naar propositionele *tijds*logica gaat geheel analoog. Voeg tijdsoperatoren P en F toe aan de verzameling gewone propositielogische connectieven. $P\varphi$ staat nu voor: 'het is het geval geweest—minstens eens in het verleden—dat φ'. $F\varphi$ staat voor 'φ zal minstens eens vanaf nu het geval zijn'. Voor het interpreteren van formules uit de propositionele tijdslogica heb je een Kripke-model nodig waarbij de werelden gevormd worden door tijdstippen. Die tijdstippen kunnen *lineair* ten opzichte van elkaar geordend zijn, dat wil zeggen: ze kunnen als volgt op een as liggen:

$\leftarrow$*vroeger* *later* $\rightarrow$

De relatie tussen de tijdstippen is uiteraard de *eerder dan* relatie. Voor elk tijdstip t is er nu een waarheidswaarden-toekenning aan de propositionele variabelen. Weer verwijzen we naar [Gamut 1982], deel 2, en de daar genoemde literatuur voor meer informatie.

Hoofdstuk 6

Predikatenlogica

6.1 Kwantoren en variabelen

De *predikatenlogica* of PL—ook wel *predikatencalculus*, *elementaire logica*, *klassieke logica* of *eerste orde logica* genoemd—is ontwikkeld om naast de analyse van voegwoorden en de negatie-operator (alles op zinsniveau) ook de analyse van de *interne structuur* van zinnen ter hand te kunnen nemen. Logisch gezien is dat van belang, want we willen graag iets kunnen zeggen over de geldigheid van redeneringen als de volgende:

Alle informatica-studenten zijn tuk op geld.
Piet is een informatica-student.

Piet is tuk op geld.

Met de propositielogica kunnen we hier niet volstaan, want propositielogisch kunnen we deze redenering alleen analyseren als:

$$\frac{\begin{array}{c}p\\q\end{array}}{r}$$

en dat voldoet niet.

Nee, de logische geldigheid van deze redenering is wezenlijk gebaseerd op de rol van de operator *alle*. Een woord als *alle* noemen we een *kwantor*. Een zeer elegante theorie voor de behandeling van kwantoren is geleverd door de reeds eerder genoemde logicus Gottlob Frege: de moderne predikatenlogica is min of meer zijn geesteskind.

Frege merkte op dat bepaalde zinsdelen in een zin kunnen worden beschouwd als *functies*. Het functioneel karakter van die zinsdelen blijkt door

een bepaald woord of een woordgroep in de zin te variëren en de rest constant te houden. Neem de zin "Jan slaat Piet". Haal de eigennaam *Jan* weg, en je krijgt "— slaat Piet". Ervan uitgaande dat de interpretatie van een zin een waarheidswaarde is, is de interpretatie van het zinsdeel "— slaat Piet" een karakteristieke functie: afhankelijk van de eigennaam die je invult krijg je de waarde 0 of 1 als uitkomst. Bij voorbeeld: voor *Marie* wordt "Marie slaat Piet" onwaar, voor "Jaap"wordt "Jaap slaat Piet" waar, enzovoorts. Maar we hebben in § 2.9 gezien dat zo'n karakteristieke functie eigenlijk niets anders is dan een *deelverzameling* van een of andere verzameling, in dit geval: een deelverzameling van de verzameling individuen die in aanmerking komen om onderwerp van gesprek te zijn. In jargon heet deze verzameling: het *discussiedomein* (Engels: *domain of discourse*). Met andere woorden: "— slaat Piet" wordt geïnterpreteerd als de verzameling van alle individuen die Piet slaan. Net zo voor "Jan slaat —". Ook dit zinsdeel heeft een (karakteristieke functie van een) deelverzameling van het discussiedomein als interpretatie: de verzameling van alle individuen die door Jan geslagen worden.

Er is geen reden om hier op te houden: we kunnen ook nog de nog overblijvende eigennaam verwijderen uit het zinsdeel "Jan slaat —". Resultaat: "$—_1$ slaat $—_2$". De open plaatsen zijn genummerd om ze uit elkaar te kunnen houden. Dat dit nummeren nodig is blijkt bij voorbeeld wanneer we "Jan slaat zichzelf" en "Jan slaat Piet" vergelijken. De zinnen zijn verschillend van structuur, en het verschil kunnen we uitdrukken door te zeggen dat "Jan slaat zichzelf" gevormd is door de eigennaam *Jan* te combineren met "$—_1$ slaat $—_1$", terwijl "Jan slaat Piet" ontstaat door combinatie van *Jan* en "$—_1$ slaat Piet", een zinsdeel dat op zijn beurt is opgebouwd uit "$—_1$ slaat $—_2$", gecombineerd met *Piet*. Hieruit blijkt dat "$—_1$ slaat $—_1$" en "$—_1$ slaat $—_2$" uit elkaar moeten worden gehouden.

Hoe moeten we "$—_1$ slaat $—_2$" nu interpreteren? Wie zich § 2.7 nog herinnert zal niet verbaasd staan over het antwoord. De interpretatie van "$—_1$ slaat $—_2$" is een tweeplaatsige relatie op het discussiedomein. In 2.7 hebben we uitgelegd dat tweeplaatsige relaties op een verzameling A niets anders zijn dan deelverzamelingen van A^2, ofwel: verzamelingen van paren elementen uit A. In het geval van ons voorbeeld wordt dit een verzameling van paren elementen uit het discussiedomein zo dat telkens het eerste lid van het paar het tweede slaat, dus bij voorbeeld:

$$\{\langle \text{Jan}, \text{Piet}\rangle, \langle \text{Jan}, \text{Marie}\rangle, \langle \text{Annie}, \text{Wim}\rangle\}.$$

Weer is er geen reden om hier op te houden, bij zinsdelen die geïnterpreteerd worden als tweeplaatsige relaties op het discussiedomein. Uit "Haddock geeft Bobbie aan Kuifje" kunnen we, door verwijderen van eigennamen, de volgende functie verkrijgen:

$$—_1 \textit{ geeft } —_2 \textit{ aan } —_3$$

De interpretatie: een drieplaatsige relatie op het discussiedomein.

Merk nu op dat we het functionele zinsdeel "$—_1$ slaat Piet" niet alleen met een eigennaam kunnen combineren tot een zin, maar ook met behulp van een zogenaamde *kwantificerende uitdrukking*, zoals *iemand*, *iedereen* of *niemand*: "Iedereen slaat Piet", "Niemand slaat Piet".

In het voorstel van Frege werd "$—_1$ slaat Piet" geïnterpreteerd als de verzameling van individuen in het discussiedomein die Piet slaan. *Jan* wordt geïnterpreteerd als een element in het discussiedomein, en de zin die ontstaat door *Jan* te combineren met "$—_1$ slaat Piet" is waar desda het element uit het discussiedomein dat de interpretatie is van *Jan* (de *referent* van *Jan*, om weer wat jargon te laten vallen) een element is van de interpretatie van "$—_1$ slaat Piet".

Eigennamen worden geïnterpreteerd als elementen van het discussiedomein, maar kwantificerende uitdrukkingen gedragen zich anders dan eigennamen. Zo heeft "Niemand slaat Piet", anders dan bij voorbeeld "Jan slaat Piet", niet tot gevolg dat Jan geslagen wordt. Een ander logisch verschil tussen eigennamen en kwantificerende uitdrukkingen is dat

Jan slaat Piet.
Jan is iemand.

Iemand slaat Piet.

een geldige redering is, waar

Iemand slaat Piet.
Iemand is Jan.

Jan slaat Piet.

niet geldig is. Dus kunnen we kwantificerende uitdrukkingen niet interpreteren als elementen van het discussiedomein. Frege's oplossing maakt gebruik van variabelen, die we nu gaan introduceren.

De notatie met de genummerde open plaatsen is lastig. In plaats daarvan gebruiken we, in het voetspoor van Frege, uitdrukkingen die voor een willekeurig individu kunnen staan, de zogenaamde *individuele variabelen*. "$—_1$ slaat $—_1$" wordt nu "x slaat x", en "$—_1$ slaat$—_2$" wordt "x slaat y".

Zinnen als "Jan slaat Piet" en "Jan veracht zichzelf" kunnen nu wat betreft hun betekenis worden beschouwd als resultaten van de *uniforme substitutie* (*substitueren* is logisch jargon voor *vervangen*) van een naam voor een variabele, in een uitdrukking die die variabele bevat. Voorbeeld: uniforme substitutie van de naam *Jan* voor de variabele x in de uitdrukking "x slaat x" levert op: "Jan slaat Jan". "Uniforme substitutie van een naam

a voor een variabele v" wil dus zeggen dat overal waar v voorkwam a komt te staan.

Bij het behandelen van kwantificerende uitdrukkingen komen de variabelen ons goed van pas. We kunnen de betekenis van "Iedereen slaat Piet" als volgt uitleggen. Welke interpretatie je ook kiest voor x (dat wil zeggen: welke *waarde* je ook aan x toekent), "x slaat Piet" is waar. Net zo voor "Iemand slaat Piet". Dit is waar wanneer er *een* toekenning bestaat voor x zodanig dat "x slaat Piet" waar is.

We kunnen dit alles nog iets explicieter opschrijven. In plaats van "Iedereen slaat Piet" schrijven we: "Voor alle x geldt: x slaat Piet". In plaats van "Iemand slaat Piet": "Voor minstens één x geldt: x slaat Piet". "Iedereen scheert zichzelf" wordt nu: "Voor alle x geldt: x scheert x".

We maken een inventaris op van wat we nu hebben:

- *namen* van individuen;
- *variabelen*, die kunnen staan voor willekeurige individuen;
- uitdrukkingen die geïnterpreteerd worden als eigenschappen of relaties (voortaan: *predikaat-uitdrukkingen*);
- "voor alle v geldt: $\cdots$" en "voor minstens één v geldt: $\cdots$", waarbij v een variabele is; voortaan noemen we deze uitdrukkingen *kwantoren*;
- de waarheidsfunctionele *voegwoorden* nemen we gewoon over uit de propositielogica.

Namen, variabelen, predikaat-uitdrukkingen, kwantoren en voegwoorden, ziedaar de ingrediënten van de predikatenlogica. In de volgende paragraaf gaan we er systematisch op in.

6.2 De syntaxis van de predikatenlogica

De definitie van de welgevormde formules van een predikatenlogische taal gaat weer met *recursie*. De situatie is nu een tikje ingewikkelder dan bij de propositielogica, omdat de allersimpelste formules (de formules die niet uit kleinere formules zijn samengesteld, ofwel: de *atomaire* formules) zelf ook nog interne structuur hebben. In de propositielogica waren de atomaire formules simpelweg propositieletters. In de predikatenlogica zijn het combinaties van namen en/of variabelen met *predikaatletters*. In plaats van "— slaapt" gebruiken we de predikaatletter S. We moeten er dan wel bij vermelden dat S moet worden gecombineerd met één naam of variabele. In jargon: S is een eenplaatsige predikaatletter. Als a een naam is, dan is Sa

een atomaire formule. Net zo: als x een variabele is, dan is Sx een atomaire formule.

Nu in het algemeen. Stel, we hebben de beschikking over een verzameling $\{a, b, c, \ldots\}$ van namen (ook wel *individuele constanten* genaamd), een verzameling $\{A, B, \ldots, R, S, \ldots\}$ van predikaatletters, waarbij van elke predikaatletter de zogenaamde *plaatsigheid* (het aantal namen en/of variabelen waarmee hij combineert) (Engels: *arity*) is gegeven, en tenslotte een verzameling individuele variabelen $\{x, y, z, \ldots\}$. Dan is het recept voor atomaire formules dit.

- Laat P een predikaatletter zijn van plaatsigheid n, en laat $t_1, \ldots, t_n$ variabelen of constanten zijn (met een parapluwoord: *individuele termen*). Dan is $Pt_1 \ldots t_n$ een atomaire formule.

Een paar voorbeelden. A is een eenplaatsige predikaatletter, c is een constante. Nu is Ac een atomaire formule (de formule zou kunnen staan voor "*c*ornelis *A*pplaudiseert"). Zij R een tweeplaatsige predikaatletter, c weer een constante, en x en y variabelen. Dan zijn Rxy, Ryx, Rxx, Ryc, Rcc atomaire formules.

Opdracht 6.1 *Vertaal in atomaire formules:*

1. *Jan scheert Piet.*
2. *Piet scheert Jan.*
3. *Jan scheert zichzelf.*

Gebruik hierbij de volgende **vertaalsleutel***:*

S:	*scheert [tweeplaatsig]*
j:	*Jan*
p:	*Piet*

We kunnen nu al een soort propositielogica gaan doen met atomaire predikaatlogische formules, door op de gebruikelijke manier de waarheidsfunctionele voegwoorden in te voeren. Alles gaat dan als in de propositielogica, alleen de propositieletters zijn vervangen door atomaire formules.

Opdracht 6.2 *We geven weer een vertaalsleutel, maar voor het gemak gebruiken we nu variabelen om de plaatsigheid van de predikaatletters aan te geven:*

$Gxyz$:	*x geeft y aan z*	Mx:	*x is een mens*
Wx:	*x weent*	Kx:	*x kraait*
Ixy:	*x is intelligenter dan y*	s:	*Sokrates*
h:	*de haan*	p:	*Petrus.*

Vertaal nu:

1. *Sokrates is intelligenter dan de haan.*
2. *Als de haan kraait, dan weent Petrus.*
3. *Sokrates geeft de haan niet aan Petrus.*
4. *De haan is geen mens.*
5. *Als Sokrates een mens is, dan is hij intelligenter dan de haan.*
6. *Sokrates is intelligenter dan de haan of de haan is intelligenter dan Sokrates.*
7. *Sokrates geeft zichzelf de haan.*

Goed, dit was aardig, maar het wordt pas echt leuk wanneer we kwantoren gaan invoeren. De *universele kwantor* of *al-kwantor*, $\forall$, wordt gecombineerd met een variabele x. $\forall x$ betekent: "Voor alle x in het discussiedomein geldt dat…". We kunnen nu, gegeven de vertaalsleutel uit de laatste opdracht, "Iedereen weent" vertalen als: $\forall x W x$. "Niet iedereen is intelligenter dan Sokrates" wordt: $\neg\forall x I x s$. $\exists$ is de *existentiële kwantor*. $\exists x$ betekent: "Er is minstens één x waarvoor geldt dat…". Dus: "Sokrates geeft iets aan Petrus" wordt: $\exists x G s x p$. "Niemand is intelligenter dan de haan": $\neg\exists x I x h$.

Hoe moeten we nu "Alle mensen wenen" vertalen? We moeten dat in elk geval zo doen dat we er, ongeacht wie of wat Petrus is, het volgende uit kunnen concluderen: "Als Petrus een mens is, dan weent Petrus". De vertaling van deze laatste zin weten we al: $Mp \rightarrow Wp$. Nu is de vertaling van "Alle mensen wenen" niet moeilijk meer: $\forall x(Mx \rightarrow Wx)$. U gelieve zelf na te gaan waarom $\forall x(Mx \wedge Wx)$ *niet* goed is.

Nu de vertaling van "Minstens één mens weent". Even zijn we geneigd om deze zin, naar analogie van het *alle*-geval, als volgt te vertalen: $\exists x(Mx \rightarrow Wx)$. Dit is echter fout, want uit de propositielogica weten we dat deze formule logisch equivalent is met $\exists x(\neg Mx \vee Wx)$, en dit is al waar als er ook maar één niet-menselijk wezen in het discussiedomein te vinden is, terwijl geen mens weent. Wat we juist willen zeggen is dat er menselijke wezens zijn die wenen, dus: $\exists x(Mx \wedge Wx)$.

"Geen mens weent" wordt nu: $\neg\exists x(Mx \wedge Wx)$. "Geen mens weent" betekent hetzelfde als "Voor elk mens geldt dat hij niet weent". We kunnen dus de zin dus evengoed als volgt vertalen: $\forall x\neg(Mx \wedge Wx)$. Dit is weer equivalent met: $\forall x(Mx \rightarrow \neg Wx)$. Al deze vertalingen zijn goed, en wel: precies even goed. Ze komen namelijk op hetzelfde neer. Dit is een verschijnsel dat we al uit de propositielogica kennen. Ook daar heeft het geen

zin om te gaan bekvechten over de keuze tussen $p \rightarrow \neg q$ en $\neg(p \wedge q)$, als vertaling van "Als Piet boos is, dan vindt Marietje hem niet aardig".

We kunnen de zaak ook omgekeerd bekijken. Wanneer twee zinnen uit het Nederlands in de predikatenlogica dezelfde vertaling krijgen, dan betekent dat dat de predikatenlogica het eventuele betekenisverschil tussen die twee zinnen verwaarloost. Neem de volgende twee zinnen: "Minstens één mens weent" en "Er is een wenende die mens is". In beide gevallen wordt de vertaling: $\exists x(Mx \wedge Wx)$. Merk op dat je ook over de keuze tussen de vertalingen $\exists x(Mx \wedge Wx)$ en $\exists x(Wx \wedge Mx)$ niet hoeft te bekvechten: die formules zijn namelijk weer equivalent. U ziet uit het voorbeeld ook dat de predikatenlogica het onderscheid tussen enkelvoud en meervoud verwaarloost: er is geen verschil in vertaling tussen "Een mens weent" en "Sommige mensen wenen". Net zo min tussen "Ieder mens weent" en "Alle mensen wenen"; de vertaling is voor allebei: $\forall x(Mx \rightarrow Wx)$. "Alle wenenden zijn mensen" en "Iedere wenende is een mens" hebben weer elk dezelfde vertaling, maar dit is een andere dan die van "Alle mensen wenen", namelijk: $\forall x(Wx \rightarrow Mx)$. Logisch gezien *moet* er natuurlijk ook verschil in vertaling zijn, want je kunt redeneren:

Alle wenenden zijn mensen.
De haan weent.

De haan is een mens.

maar niet:

Alle mensen wenen.
De haan weent.

De haan is een mens.

Merk op dat er geen verschil in betekenis is tussen aan de ene kant de formule $\forall x(Mx \rightarrow Wx)$ en aan de andere kant $\forall y(My \rightarrow Wy)$. Het feit dat er een andere variabele is gebruikt maakt voor de interpretatie van de formule niets uit.

Tot nu toe hebben we steeds voorbeelden besproken van formules met maar één kwantor, maar er kunnen in een formule willekeurig veel kwantoren voorkomen. Voor de vertaling van "Elke jongen houdt van een meisje" hebben we een universele *en* een existentiële kwantor nodig:

$$\forall x(Jx \rightarrow \exists y(My \wedge Hxy)).$$

De formule drukt uit dat elke jongen een lief heeft (om met de Vlamingen te spreken). Als we willen zeggen dat er een meisje is dat zich in de liefde van alle jongens mag koesteren (bij voorbeeld Brigitte), dan kunnen we dit uitdrukken door een andere formule te gebruiken:

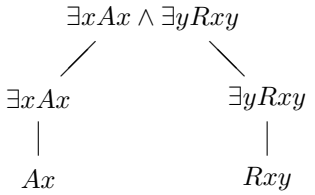
$$\exists x(Mx \wedge \forall y(Jy \rightarrow Hyx)).$$

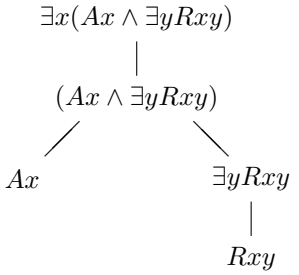
Het voorbeeld laat zien dat de predikatenlogica van pas kan komen om twee betekenissen van de zin "Elke jongen houdt van een meisje" te onderscheiden. Het verschil zit hem in de *bereik*-verhoudingen tussen de kwantoren $\forall$ en $\exists$ die respectievelijk corresponderen met *elke* in *elke jongen* en *een* in *een meisje*. We zeggen dat in de formule $\forall x(Jx \rightarrow \exists y(My \wedge Hxy))$ de universele kwantor groter bereik heeft dan de existentiële, terwijl dat voor de formule $\exists x(Mx \wedge \forall y(Jy \rightarrow Hyx))$ precies andersom ligt. We zeggen ook wel: de universele kwantor heeft in $\forall x(Jx \rightarrow \exists y(My \wedge Hxy))$ groot bereik (Engels: *wide scope*), de existentiële kwantor heeft in deze formule klein bereik (Engels: *narrow scope*).

Het *bereik* van een kwantor in een welgevormde formule van de predikatenlogica is, intuïtief gesproken, het gedeelte van die formule waarover de kwantor iets te zeggen heeft, dat wil zeggen het gedeelte waar hij variabelen kan *binden*. Een paar voorbeelden kunnen dit duidelijk maken. We beschouwen de volgende twee formules:

- $\exists xAx \wedge \exists yRxy$
- $\exists x(Ax \wedge \exists yRxy)$.

Het bereik van de kwantoren in deze formules valt af te leiden uit de samenstelling van de formules: een kwantor heeft bereik over de deelformule waarmee hij is samengesteld. Hoewel we de precieze recursieve definitie van van predikatenlogische formules nog moeten geven, zijn hier alvast een paar constructiebomen (vergelijk § 5.4 voor constructiebomen in de propositielogica; het begrip *bereik van een kwantor* is geheel analoog aan het begrip *bereik van een zinsoperator*):





Uit deze constructiebomen valt het bereik van $\exists x$ in de twee formules af te leiden. In de eerste boom vormt $\exists x$ samen met de deelformule Ax een formule $\exists xAx$, die vervolgens wordt geconjugeerd met een andere formule. Het bereik van de kwantor $\exists x$ is hier dus de deelformule Ax. In de tweede boom wordt $\exists x$ gecombineerd met $(Ax \wedge \exists yRxy)$ tot een nieuwe formule. Het bereik van de kwantor $\exists x$ is hier dus de formule $(Ax \wedge \exists yRxy)$.

In de twee formules die we zoëven bekeken hebben zouden we het bereik van de kwantor $\exists x$ met behulp van onderstrepingen als volgt kunnen aangeven:

- $\exists x\underline{Ax} \wedge \exists yRxy$
- $\exists x\underline{(Ax \wedge \exists yRxy)}$

Zulke onderstrepingen geven natuurlijk geen nieuwe informatie: omdat constructiebomen voor formules uniek zijn, ligt het bereik van alle kwantoren die in een formule voorkomen ondubbelzinnig vast.

We komen nog even terug op het voorbeeld "Elke jongen houdt van een meisje", met zijn twee predikatenlogische parafrases:

- $\forall x(Jx \rightarrow \exists y(My \wedge Hxy))$
- $\exists y(My \wedge \forall x(Jx \rightarrow Hxy))$

In de eerste parafrase is het bereik van de universele kwantor de deelformule $Jx \rightarrow \exists y(My \wedge Hxy)$, en dat van de existentiële kwantor de deelformule $(My \wedge Hxy)$. We zien dus dat het bereik van de existentiële kwantor een *echt deel* is van het bereik van de universele kwantor. In het geval van $\exists y(My \wedge \forall x(Jx \rightarrow Hxy))$ ligt dit precies andersom: hier is het bereik van de universele kwantor een *echt deel* van het bereik van de existentiële. We kunnen nu een heel precieze definitie geven van wat het betekent dat de ene kwantor groter bereik heeft dan de andere.

Definitie 6.1 *In formule φ heeft (een voorkomen van een) kwantor K_1* **groot bereik** *ten opzichte van (een voorkomen van een) kwantor K_2 (of, wat op hetzelfde neerkomt: heeft K_2* **klein bereik** *ten opzichte van K_1) desda het bereik van K_2 een deelformule is van het bereik van K_1.*

Merk op dat in de formule $\exists xAx \wedge \exists yRxy$ de variabele x in Rxy (preciezer gezegd: het *voorkomen* van x in Rxy; Engels: the *occurrence* of x in Rxy) door geen van beide kwantoren wordt gebonden. De kwantor $\exists y$ bindt x in Rxy niet, omdat $\exists y$ alleen voorkomens van y binnen zijn bereik bindt; de kwantor $\exists x$ bindt x in Rxy niet, omdat Rxy niet binnen het bereik van deze kwantor valt. Zulke niet-gebonden voorkomens van variabelen heten *vrij* (Engels: *free*), de andere voorkomens heten *gebonden* (Engels: *bound*).

We zeggen: de variabele x komt vrij voor in $\exists x Ax \wedge \exists y Rxy$. De variabele x komt overigens ook gebonden voor in $\exists x Ax \wedge \exists y Rxy$. Dus: dezelfde variabele kan in een formule zowel vrij als gebonden voorkomen. De variabele y komt alleen gebonden voor in $\exists x Ax \wedge \exists y Rxy$.

We kunnen ook spreken over het vrij/gebonden zijn van voorkomens van variabelen (Engels: *variable occurrences*) in een *deelformule* van een formule. Voorbeeld: de variabele x komt alleen gebonden voor in de formule $\exists x(Ax \wedge \exists y Rxy)$, maar ze is vrij in de deelformule $(Ax \wedge \exists y Rxy)$: *binnen die deelformule* komt geen kwantor voor die de beide voorkomens van x bindt.

Opdracht 6.3 *Geef het bereik aan van de kwantor $\forall x$ in de volgende formules:*

1. $\forall x Rxx$
2. $\forall x \exists y \forall z Sxyz$
3. $\exists z \forall x \forall y Sxyz$
4. $Ac \wedge \forall x \exists y Rxy$
5. $\exists z \forall x (Ax \wedge Bz)$
6. $\forall x (Ax \rightarrow Bx) \wedge Cx$
7. $\forall x ((Ax \rightarrow Bx) \wedge Cx)$
8. $(Ax \rightarrow \forall x Bx) \wedge Cx$.

We spreken af dat de combinatie van een kwantor $\forall x$ of $\exists x$ met een formule φ zo moet worden geïnterpreteerd dat de kwantor alle voorkomens van x die *vrij* zijn in φ bindt. Nu zijn er een paar speciale gevallen. Het kan gebeuren dat x helemaal niet voorkomt in φ. Voorbeeld: $\forall x \exists y Ryy$. De kwantor $\forall x$ bindt niets, want x komt niet voor in $\exists y Ryy$. We spreken in zo'n geval van *loze kwantificatie* (Engels: *vacuous quantification*). De reden waarom we loze kwantificatie toestaan is een hele goede: het vereenvoudigt de syntactische definitie van de predikatenlogica; je hoeft nu immers het loze geval niet meer uit te sluiten.

Een iets ander speciaal geval bij het combineren van een kwantor $\forall x$ en een formule φ hebben we—u had het waarschijnlijk al zelf bedacht—wanneer x wel voorkomt in φ, maar alleen *gebonden*. Voorbeeld: $\forall x \exists x Rxx$. Hier wordt $\forall x$ gecombineerd met $\exists x Rxx$, en in die deelformule komt x alleen gebonden voor. Volgens afspraak mag $\forall x$ alleen *vrije* voorkomens van x binden in de deelformule waarmee de kwantor combineert, dus ook hier is de kwantificatie *loos*. U gelieve te onthouden: een voorkomen van een

variabele wordt gebonden door de dichtstbijzijnde kwantor die *loopt* over die variabele en die dat voorkomen van de variabele in zijn bereik heeft. Dus: in de formule $\forall x(Ax \rightarrow (Bx \vee \exists x Cx))$ wordt de x in Cx gebonden door de existentiële kwantor.
We geven van enige belangrijke begrippen formele definities.

Definitie 6.2 *'φ is een deelformule van ψ' wordt recursief gedefinieerd door (φ en ψ zijn welgevormde formules):*

- *φ is een deelformule van φ.*
- *Als φ een deelformule van ψ is, dan is φ ook een deelformule van $\neg\psi$, $\forall v\psi$, $\exists v\psi$ (v is een variabele).*
- *Als φ een deelformule is van ψ of van χ, dan ook van $(\psi \wedge \chi)$, $(\psi \vee \chi)$, $(\psi \rightarrow \chi)$ of $(\psi \leftrightarrow \chi)$.*

Opdracht 6.4 *Geef zelf een recursieve definitie van 'v is een vrij voorkomen van een variabele in φ'.*

Definitie 6.3 *v is een* **gebonden** *voorkomen van een variabele in φ wanneer v niet vrij is in φ.*

Definitie 6.4 *Formules waarin alle voorkomende variabelen gebonden zijn door een of andere kwantor heten* **gesloten formules** *(Engels:* closed formulas*), of ook wel:* **zinnen** *(Engels:* sentences*).*

Definitie 6.5 *Formules waarin een of meer variabelen vrij voorkomen heten* **open formules** *(Engels:* open formulas*).*

Opdracht 6.5 *Geef aan welke van de formules uit opdracht 6.3 open zijn en welke gesloten.*

Het verschil tussen open en gesloten formules blijkt bij het interpreteren van die formules (zie § 6.3): het waar of onwaar zijn van gesloten formules hangt niet af van een keuze voor de interpretatie van de variabelen van de taal, het waar of onwaar zijn van open formules hangt daar wel van af. Voor we Ax kunnen interpreteren moeten we weten waar x op slaat, maar bij $\forall x Ax$ hoeven we dat niet te weten.

Een volgend punt is: hoe verhouden de existentiële en de universele kwantor zich tot elkaar? Die onderlinge verhoudingen kunnen we uitdrukken met behulp van de negatie-operator. $\neg\exists x Ax$ en $\forall x\neg Ax$ zijn logisch equivalent: ontkennen dat er dingen zijn met zekere eigenschap komt immers op hetzelfde neer als zeggen dat van alle dingen geldt dat ze die eigenschap niet hebben. Heel in het algemeen hebben we:

- $\forall v \neg\varphi \iff \neg\exists v\varphi$
- $\exists v \neg\varphi \iff \neg\forall v\varphi$.

Deze twee principes kunnen worden gebruikt om een negatie-teken "door een kwantor heen te trekken". Er volgt ook uit dat we in principe maar een van beide kwantoren nodig hebben. Wanneer we $\forall$ hebben kunnen we $\exists$ met behulp daarvan en van de negatie-operator definiëren, namelijk als $\neg\forall\neg$. Omgekeerd kunnen we ook $\forall$ definiëren in termen van negatie en de andere kwantor, namelijk als $\neg\exists\neg$.

We kunnen deze principes gebruiken om formules te vervangen door equivalente formules met alleen universele kwantoren, of met alleen existentiële kwantoren. Als voorbeeld schrijven we $\neg\exists x(Ax \wedge Bx)$ met alleen universele kwantoren:

$$\neg\exists x(Ax \wedge Bx) \iff \neg\neg\forall x\neg(Ax \wedge Bx) \iff \forall x\neg(Ax \wedge Bx).$$

In de laatste stap is gebruik gemaakt van de wet van de dubbele negatie.

Opdracht 6.6 *Geef voor de volgende formules equivalente formules met alleen universele kwantoren:*

1. $\forall x\exists yRxy$
2. $\exists xAx \vee \forall yBy$
3. $\exists x\forall y\exists zSxyz$
4. $\neg\exists xAx \vee \forall yBy$
5. $\neg(\exists xAx \vee \forall yBy)$
6. $\exists x\exists y\exists z(Ax \vee Ryz)$.

Opdracht 6.7 *Geef voor de formules uit de vorige opdracht ook equivalente formules met alleen existentiële kwantoren.*

Het wordt hoog tijd voor de recursieve definitie van de verzameling welgevormde formules van predikatenlogische talen. Elke verzameling individuele constanten en predikaatletters (waarbij van elke predikaatletter de plaatsigheid is gegeven) bepaalt een predikatenlogische taal $\mathcal{T}$. Verder dient een verzameling individuele variabelen gegeven te zijn. Hier is de recursieve definitie van de welgevormde formules van $\mathcal{T}$, gegeven een verzameling van constanten, een verzameling van predikaatletters, en een verzameling van individuele variabelen:

Definitie 6.6 Welgevormde formules *van predikatenlogische taal $\mathcal{T}$:*

- *Als A een n-plaatsige predikaatletter is en $t_1, \ldots, t_n$ zijn variabelen of constanten, dan is $At_1 \ldots t_n$ een welgevormde formule van $\mathcal{T}$.*
- *Als φ een welgevormde formule is van $\mathcal{T}$, dan $\neg\varphi$ ook.*
- *Als φ en ψ welgevormde formules zijn van $\mathcal{T}$, dan $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$ en $(\varphi \leftrightarrow \psi)$ ook.*
- *Als φ een welgevormde formule is van $\mathcal{T}$, en v is een variabele, dan zijn $\forall v\varphi$ en $\exists v\varphi$ ook welgevormde formules van $\mathcal{T}$.*
- *Niets anders is een welgevormde formule van $\mathcal{T}$.*

Merk op dat er in de bovenstaande definitie weer metavariabelen zijn gebruikt: A is een metavariabele over predikaatletters, t_1 tot en met t_n zijn metavariabelen over individuele termen (constanten of variabelen), v is een metavariabele over variabelen, en φ en ψ zijn metavariabelen over formules. De structuur van de definitie is geheel analoog aan die voor de welgevormde formules van propositielogische talen (vergelijk § 5.4).

Net als in de propositielogica sjoemelen we met de haakjes waar het geen kwaad kan. Net als in de propositielogica kunnen we alternatieve notaties gebruiken voor de formules van de predikatenlogica: Poolse notatie, omgekeerd Pools, en lijst-notatie (vergelijk § 5.4). We gaan er hier niet op in.

Het is bijzonder nuttig nog wat verder te oefenen in het vertalen van het Nederlands naar de predikatenlogica. Dergelijke vertalingen leggen logische overeenkomsten bloot tussen zinnen die er 'oppervlakkig' verschillend uitzien:

(1) *Op elk potje past een dekseltje.*

(2) *Ieder huisje heeft zijn kruisje.*

(3) *Alle gebeurtenissen hebben een oorzaak.*

(4) *Iedereen houdt van iemand.*

Deze zinnen vertonen overeenkomst in *logische vorm*, hetgeen blijkt uit het feit dat ze allemaal kunnen worden vertaald met behulp van de formule

(5) $\forall x(Px \rightarrow \exists y(Qy \wedge Rxy))$

waarbij P, Q en R dan natuurlijk steeds andere predikaten uitdrukken.

In het algemeen moet je, om van het Nederlands naar de predikatenlogica te kunnen vertalen, ten eerste aangeven wat het *discussiedomein* is, en ten tweede hoe de predikaatletters en constanten die je in je vertaling gebruikt, moeten worden gelezen (dat wil zeggen: je moet een *vertaalsleutel* geven).

Stel dat we willen vertalen: *Al het aardse is vergankelijk*. We kiezen als discussiedomein de verzameling van aardse en bovenaardse dingen. De vertaalsleutel wordt: Ax: x is aards; Vx: x is vergankelijk. Op grond hiervan wordt de vertaling: $\forall x(Ax \rightarrow Vx)$.

Merk op dat de keuze van het discussiedomein de vertaling kan beïnvloeden. Als bij de vertaling van "Al het aardse is vergankelijk" de verzameling van aardse dingen als discussiedomein wordt genomen, dan wordt de vertaling simpelweg: $\forall xVx$.

Verder kunt u natuurlijk sjoemelen bij het vertalen door het kiezen van een vertaalsleutel waarin een deel van de logische structuur van een zin is weggemoffeld. Stel bij voorbeeld dat u moet vertalen: "Niemand kijkt een (hem) gegeven paard in de bek". U kunt zich hier vanaf maken door de verzameling van mensen als discussiedomein te kiezen, en als vertaalsleutel te nemen:

- Kx: x kijkt een hem gegeven paard in de bek.

De vertaling wordt nu simpelweg: $\neg\exists xKx$. Helaas, eenvoud is hier niet het kenmerk van het ware, want het kiezen van de bovenstaande vertaalsleutel betekent dat de kans verkeken is om de geldigheid van de volgende redenering predikatenlogisch te verantwoorden:

> Niemand kijkt een (hem) gegeven paard in de bek.
> Elias geeft Blessie aan Jan.
> Blessie is een paard.
> ---
> Jan kijkt Blessie niet in de bek.

Nee, we moeten juist in de predikatenlogische vertaling zo weinig mogelijk van de logische structuur verdonkeremanen. Een veel betere vertaling van "Niemand kijkt een (hem) gegeven paard in de bek" krijgen we wanneer we als discussiedomein nemen: de verzameling van alle mensen en paarden, en als vertaalsleutel:

- Mx: x is een mens;
- $Gxyz$: x geeft y aan z;
- Px: x is een paard;
- Kxy: x kijkt y in de bek.

De vertaling wordt nu:

$$\forall x\forall y\forall z((Mx \wedge Py \wedge Mz \wedge Gxyz) \rightarrow \neg Kzy).$$

Letterlijk: voor alle drietallen bestaande uit een gever, een gegeven paard, en een ontvanger geldt dat de ontvanger het hem gegeven paard niet in de bek kijkt.

Opdracht 6.8 *Vertaal de volgende zinnen uit het Nederlands naar de taal van de predikatenlogica. Geef steeds het discussiedomein en de vertaalsleutel aan.*

1. *Bhagwan is een profeet of hij is een profiteur.*
2. *Als Bhagwan een profeet is, dan adoreert Annie hem.*
3. *Iedereen die Bhagwan volgt adoreert hem.*
4. *Wie Bhagwan niet adoreert, volgt hem niet.*
5. *Alles wat Bhagwan zegt spreekt Jan tegen.*
6. *Als Bhagwan iets zegt, dan spreekt Jan het tegen.*
7. *Geen van zijn volgelingen spreekt tegen wat Bhagwan zegt.*

U kunt bij het maken van deze opdracht gebruik maken van de aanwijzing die hieronder wordt gegeven.

Aanwijzing: Wanneer u opdracht 6.8 probeert te maken zult u merken dat het vinden van goede predikatenlogische vertalingen soms enige inventiviteit vergt. In onderdeel 4. moet u bij voorbeeld bedenken dat *wie* staat voor *al wie*, zodat er in feite een universele bewering wordt gedaan. In 6. wordt u geconfronteerd met het probleem dat het voornaamwoord *het* in de nazin het woord *iets* in de voorzin als antecedent moet hebben: in de predikatenlogica moet dit verband worden aangegeven door ervoor te zorgen dat de kwantor die de vertaling vormt van het antecedent de variabele bindt die de vertaling vormt van het voornaamwoord. Bij rechttoe, rechtaan vertalen lukt dit niet, want dan is het bereik van de kwantor te klein. U komt hier alleen uit wanneer u 'naar de geest vertaalt', door eerst over te stappen naar een equivalente zin waarin dit probleem niet optreedt.

Opdracht 6.9 *Nog een paar vertaalklussen. Geef weer discussiedomein en vertaalsleutel aan.*

1. *Alle barbiers die zichzelf niet scheren, worden door een barbier geschoren.*
2. *Alle barbiers die zichzelf niet scheren, worden door geen enkele barbier geschoren.*
3. *Geen barbier die zichzelf niet scheert wordt door alle barbiers geschoren.*
4. *Als een barbier zichzelf niet scheert, dan scheert hij niet iedereen.*

5. *Als een barbier zichzelf niet scheert, dan scheert hij niet iedereen die zichzelf niet scheert.*

Opdracht 6.10 *Vertaal de volgende Nederlandse spreekwoorden in de taal van de predikatenlogica. Geef steeds het discussiedomein en de vertaalsleutel aan. Wellicht ten overvloede: u moet de letterlijke zin van de spreekwoorden weergeven. Dit neemt echter niet weg dat u moet weten wat een spreekwoord* betekent *voor u aan het vertalen slaat, want de betekenissen bepalen hoe eventuele syntactische dubbelzinnigheden moeten worden opgelost.*

1. *Wie zijn kinderen liefheeft, kastijdt hen.*
2. *Geen rozen zonder doornen.*
3. *Alle hout is geen timmerhout.*
4. *Het is niet alles goud wat blinkt.*
5. *Kinderen die vragen worden overgeslagen.*
6. *Het zijn niet allen koks, die lange messen dragen.*

De predikatenlogica heeft een zeer grote kracht. Wanneer het u niet gelukt is om alle opgedragen vertalingen te maken ligt dat niet aan de predikatenlogica maar aan u. De benodigde inventiviteit voor het vinden van adekwate predikatenlogische vertalingen kan echter worden aangeleerd.

De kracht van de predikatenlogica blijkt bij voorbeeld uit het feit dat de hele verzamelingenleer (waarover u in hoofdstukken 2 en 3 hierboven het een en ander aan de weet bent gekomen) in predikatenlogica valt uit te drukken.

Opdracht 6.11 *Verzamelingtheoretisch jargon kan worden omgezet in predikatenlogica. Bij voorbeeld: $a \in A$ wordt predikatenlogisch: Aa. Vertaal met behulp van dit gegeven:*

1. $A \subseteq B$
2. $A = B$
3. *Als* $A \subseteq B$ *en* $B \subseteq C$ *dan* $A \subseteq C$.

Complexe predikaten kunnen soms worden opgebouwd uit eenvoudiger predikaten plus wat logisch bindmiddel. Zo kun je *grootvader* definiëren in termen van *ouder* en *mannelijk*. Laat de volgende vertaalsleutel gegeven zijn:

- Mx: x is van het mannelijk geslacht;

- Oxy: x is ouder van y.

Aangenomen dat de verzameling mensen het discussiedomein is levert dit de volgende vertaling op voor "x is grootvader van y": $\exists z(Oxz \wedge Ozy \wedge Mx)$.

Opdracht 6.12 *Neem als discussiedomein de verzameling van alle mensen, en gebruik de vertaalsleutel die hierboven gegeven is. Vertaal nu:*

1. *Niemand is ouder van zichzelf.*
2. *Iedereen heeft een vader.*
3. *Iedereen heeft een moeder.*
4. *Iemand heeft een kind.*
5. *Niet iedereen is vader van iemand.*
6. *Niet iedereen heeft een dochter en een zoon.*
7. *Iedereen heeft een grootmoeder.*
8. *Niet iedereen heeft een kleinkind.*

Opdracht 6.13 *Laat de volgende vertaalsleutel gegeven zijn:*

- Kxy *: x is kind van y;*
- m*: Marie;*
- Mx*: x is van het mannelijk geslacht;*
- p*: Piet.*

Het discussiedomein is: alle mensen. Parafraseer nu de volgende predikatenlogische formules in het Nederlands:

1. $\exists x Kxm$
2. $\exists x(Kxm \wedge Kpx)$
3. $\exists x((Kxm \wedge Mx) \wedge Kpx)$
4. $\exists x((Kxm \wedge Kxp) \wedge \neg Mx)$
5. $\exists x(Kxm \wedge \forall y(Kym \rightarrow Kyp))$
6. $\exists x((Kxm \wedge Mx) \wedge \neg\exists y(Kxy \wedge My))$.

Opdracht 6.14 *Geef de volgende redeneringen predikatenlogisch weer (vertaalsleutels en discussiedomeinen vermelden):*

1. *Alle mensen zijn sterfelijk.*
   Sokrates is een mens.

   Sokrates is sterfelijk.

2. *Geen vis is een zoogdier.*
   Alle knaagdieren zijn zoogdieren.

   Geen knaagdier is een vis.

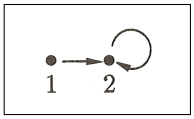
Goed, we kunnen nu redeneringen uit het Nederlands vertalen naar formules uit de predikatenlogica. Wat we vervolgens graag zouden willen is: nagaan of die predikatenlogische redeneringen *geldig* zijn. De vraag naar geldigheid is een semantische vraag; zij komt in de nu volgende paragraaf aan de orde.

6.3 De semantiek van de predikatenlogica

De semantiek van de predikatenlogica is voor het eerst expliciet—dat wil zeggen: in verzamelingstheoretische termen—geformuleerd door de Poolse logicus Alfred Tarski, in 1933. Tarski werkte het idee uit om predikatenlogische formules te interpeteren in verzamelingstheoretische structuren, de zogenaamde *modellen* voor de predikatenlogica. Vooraleer we in verzamelingstheoretische termen gaan uitleggen wat een model voor een predikatenlogische taal $\mathcal{T}$ is, beginnen we met het tekenen van *plaatjes* voor modellen, om zo een intuïtief idee te krijgen van de predikatenlogische semantiek. Bekijk het volgende plaatje:

Laten we zeggen dat dit een plaatje is van een situatie waarin er maar één ding bestaat (dus: het discussiedomein bestaat uit een enkel object), en dat ding heeft een bepaalde relatie (in het plaatje weergegeven met een pijl) tot zichzelf. Wanneer we ons nu voorstellen dat de predikaatletter R verwijst naar de relatie die in het plaatje door de pijl wordt weergegeven, dan kunnen we met behulp van predikatenlogische formules *uitspraken* gaan doen over de situatie in het plaatje. De volgende formules zijn bij voorbeeld *waar* in de situatie van het plaatje: $\forall xRxx$; $\exists xRxx$; $\forall x\exists yRxy$; $\exists x\forall yRxy$ (ga dit na). De volgende formule is *niet waar* in de situatie van het plaatje: $\exists x\neg Rxx$. Uiteraard zijn ook alle ontkenningen van de formules die waar zijn in de situatie van het plaatje, onwaar.

In plaatjes met meerdere objecten geven we voor het gemak de objecten aan met cijfers:

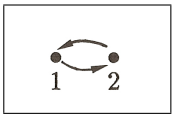


Het plaatje geeft de situatie aan waarin er twee objecten 1 en 2 zijn, waarbij 1 niet in de pijlrelatie staat tot zichzelf, maar wel tot 2, terwijl 2 in de pijlrelatie staat tot zichzelf, maar niet tot 1.

Opdracht 6.15 *Wanneer we R weer interpreteren als de pijlrelatie, welke van de volgende predikatenlogische formules zijn dan waar in de hierboven gegeven situatie?*

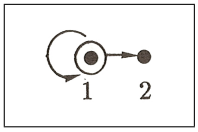
1. $\forall xRxx$
2. $\exists xRxx$
3. $\forall x\exists yRxy$
4. $\exists x\forall yRxy$
5. $\forall x(\exists yRxy \rightarrow Rxx)$
6. $\exists x\neg Rxx$
7. $\forall x\forall y(Rxy \rightarrow Rxx)$.

Opdracht 6.16 *Beschouw nu het volgende plaatje:*



Geef aan welke van de formules uit de vorige opdracht waar zijn in de situatie van dit plaatje.

Laten we, behalve de pijlrelatie, ook nog de cirkeleigenschap invoeren. Alle objecten die omcirkeld zijn hebben de cirkeleigenschap. Een mogelijk plaatje is nu:

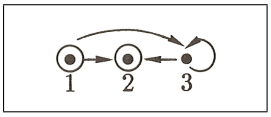


Object 1 heeft de pijlrelatie tot zichzelf en tot 2; 1 heeft bovendien de cirkeleigenschap. 2 heeft de cirkeleigenschap niet, en staat ook tot niets in de pijlrelatie. Veronderstel nu dat de predikaatletter P wordt geïnterpreteerd als de cirkeleigenschap. Dan kunnen we nu onderzoeken of predikatenlogische formules waarin de predikaatletters P en R voorkomen waar zijn in de situatie van dit laatste plaatje.

Opdracht 6.17 *Ga na welke van de volgende formules waar zijn in het laatst getekende plaatje:*

1. $\exists x Px$
2. $\forall x Px$
3. $\forall x(Rxx \rightarrow Px)$
4. $\forall x(Px \rightarrow Rxx)$
5. $\exists x(Px \wedge \neg Rxx)$
6. $\forall x(Px \rightarrow \exists y Rxy)$
7. $\forall x(\exists y Ryx \rightarrow Px)$
8. $\forall x(Rxx \leftrightarrow \neg Px)$
9. $\exists x \exists y(Rxy \wedge \neg Px \wedge \neg Py)$
10. $\forall x(Rxx \rightarrow \exists y(Rxy \wedge Py))$
11. $\forall x(Px \rightarrow \exists y Ryx)$
12. $\exists x \exists y(Rxy \wedge \neg Ryx \wedge \exists z(Rxz \wedge Rzy))$.

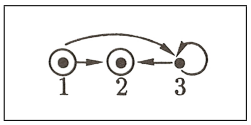
Opdracht 6.18 *Als de vorige opdracht maar nu voor het volgende plaatje:*



De predikatenlogische taal die we nodig hebben om de bovenstaande plaatjes te beschrijven is een hele simpele: hij bevat een eenplaatsige predikaatletter P en een tweeplaatsige predikaatletter R. Laten we nu aannemen dat onze taal ook nog constanten a, b en c heeft. Om te weten hoe we de beschrijving van de plaatjes met behulp van deze taal moeten opvatten, moeten we twee dingen weten. Ten eerste: welke objecten in het plaatje worden door welke constanten benoemd? We kunnen bij voorbeeld afspreken

dat in het plaatje uit de laatste opdracht a een naam is voor object 1, b een naam voor object 2, en c een naam voor object 3. Ten tweede: op welke eigenschap van objecten in het plaatje slaat de predikaatletter P (we hebben hierboven afgesproken: op de cirkeleigenschap), en op welke relatie tussen objecten slaat de predikaatletter R (we hebben hierboven afgesproken: op de pijlrelatie)?

Nu algemener: een paar dat bestaat uit een verzameling dingen D en een functie I die aan de constanten van een predikatenlogische taal $\mathcal{T}$ objecten uit D toekent, aan de eenplaatsige predikaatletters van $\mathcal{T}$ deelverzamelingen van D, aan de tweeplaatsige predikaatletters van $\mathcal{T}$ deelverzamelingen van D^2 (dat wil zeggen: tweeplaatsige relaties op D), aan de drieplaatsige predikaatletters van $\mathcal{T}$ deelverzamelingen van D^3 (drieplaatsige relaties op D), enzovoort, noemen we een *model* voor de taal $\mathcal{T}$. Als $\mathcal{M} = \langle D, I \rangle$ een model is, dan heten D en I respectievelijk het *domein* en de *interpretatiefunctie* van het model. Vaak wordt een model voor een predikatenlogische taal een *structuur* genoemd. We zien nu dat het volgende plaatje, gegeven onze afspraken over wat de constanten a, b, c en de predikaatletters P en R betekenen, een model vertegenwoordigt voor de taal die alleen a, b en c als constanten heeft, en alleen P en R als predikaatletters:



Immers, dat model $\mathcal{M} = \langle D, I \rangle$ ziet er als volgt uit. Het domein bestaat uit de objecten 1, 2 en 3, dus:

$$D = \{1, 2, 3\}.$$

De interpretatiefunctie I heeft als domein de verzameling

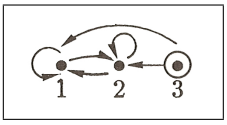
$$\{a, b, c, P, R\},$$

en kent de volgende waarden toe:

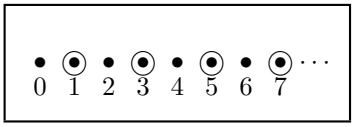
- $I(a) = 1$ [Constante a benoemt object 1.]
- $I(b) = 2$ [Constante b benoemt object 2.]
- $I(c) = 3$ [Constante c benoemt object 3.]
- $I(P) = \{1, 2\}$ [Predikaatletter P wordt geïnterpreteerd als de verzameling van alle objecten die omcirkeld zijn.]

- $I(R) = \{\langle 1,2\rangle, \langle 1,3\rangle, \langle 3,3\rangle, \langle 3,2\rangle\}$ [Predikaatletter R wordt geïnterpreteerd als de verzameling van alle paren waarvan het eerste lid in de pijlrelatie staat tot het tweede.]

Opdracht 6.19 *Omschrijf de structuur die bepaald wordt door het volgende plaatje (neem aan dat de afspraken over wat de constanten a, b en c en de predikaatletters P en R betekenen hetzelfde blijven):*



In alle plaatjes die we tot nu toe getekend hebben kwamen slechts eindig veel objecten voor. Dat was alleen voor het gemak. Een structuur voor een predikatenlogische taal kan heel goed een oneindig domein hebben. Hier is een voorbeeld:



Dit plaatje symboliseert de verzameling $\mathbf{N} = \{0, 1, 2, 3, 4, \dots\}$ van de natuurlijke getallen. De *oneven* getallen zijn omcirkeld. Spreek af dat we de predikaatletter P interpreteren als de eigenschap *oneven zijn*. Spreek verder af dat we R interpreteren als de relatie *groter zijn dan*. Laat opnieuw a, b en c respectievelijk de objecten 1, 2 en 3 benoemen. Dan bepaalt het plaatje (met de afspraken) het volgende model voor de taal $\mathcal{T}$: $\mathcal{M} = \langle D, I\rangle$, waarbij $D = \{0, 1, 2, 3, \dots\}$, $I(a) = 1$, $I(b) = 2$, $I(c) = 3$, $I(P) = \{1, 3, 5, \dots\}$, $I(R) = \{\langle 1,0\rangle, \langle 2,0\rangle, \langle 2,1\rangle, \langle 3,0\rangle, \langle 3,1\rangle, \langle 3,2\rangle, \langle 4,0\rangle, \dots\}$. Merk op dat in deze structuur niet alle objecten een naam hebben. Alleen 1, 2 en 3 worden benoemd, de rest is naamloos.

Opdracht 6.20 *Ga na welke van de volgende formules waar zijn in de hierboven gegeven structuur:*

1. $\forall x \forall y \forall z((Rxy \wedge Ryz) \rightarrow Rxz)$
2. $\forall x \neg Rxx$
3. $\forall x \forall y(Rxy \vee Ryx)$
4. $\forall x \exists y Ryx$
5. $\forall x \exists y Rxy$

6. $\forall x \exists y(Py \wedge Rxy)$
7. $\exists x \forall y Rxy$
8. Pa
9. $Pb \rightarrow \forall x Px$
10. $\forall x(Rxx \rightarrow Px)$
11. $\exists x \forall y(\neg Py \rightarrow Ryx)$.

De oneindigheid van het laatste model is van het simpelste soort. de verzameling natuurlijke getallen is immers *aftelbaar* (zie hoofdstuk 3). De domeinen kunnen natuurlijk ook *overaftelbaar* zijn. Een beroemde stelling (van Löwenheim en Skolem) zegt dat wanneer er een *overaftelbaar* model voor een verzameling formules bestaat, er ook een *aftelbaar* model moet zijn. We zullen er hier niet dieper op ingaan. Wel zullen we verderop zien dat we niet altijd met een *eindig* model kunnen volstaan.

Opdracht 6.21 *Verzamelingen-jargon predikatenlogisch weergeven kan op de manier van opdracht 6.11, maar het kan ook anders. Beschouw een model $\mathcal{M}$ met verzamelingen als individuendomein, en met een twee-plaatsige relatie $\in$ als gegeven. Interpreteer de tweeplaatsige predikaatletter R als $\in$. Vertaal nu de volgende formules terug in verzamelingenjargon.*

1. $\exists x \forall y \neg Ryx$
2. $\neg \exists x Rxx$
3. $\forall x \forall y(Rxy \rightarrow \neg Ryx)$
4. $\forall x \forall y(Rxy \rightarrow \exists z Ryz)$.

We hebben hierboven gedefiniëerd wat we bedoelen met: "$\mathcal{M}$ is een model voor predikatenlogische taal $\mathcal{T}$". In de praktijk van de opdrachten hierboven zijn we ook al geconfronteerd met de vraag: wat betekent het dat een formule φ van taal $\mathcal{T}$ waar is in een model $\mathcal{M}$ voor $\mathcal{T}$? Wat we nu nog moeten doen is de achterliggende theorie onder woorden brengen; we moeten een *definitie* geven van *waarheid* voor willekeurige formules van $\mathcal{T}$ in modellen $\mathcal{M}$ voor $\mathcal{T}$. Preciezer gezegd: we moeten de interpretatiefunctie I van een model $\mathcal{M}$ gaan gebruiken voor het definiëren van een *valuatiefunctie* $V_{\mathcal{M}}$ die aan elke formule van onze taal een waarheidswaarde toekent.

Een kleine complicatie hierbij is dat we een afspraak moeten maken omtrent de interpretatie van de vrije variabelen in de open formules van de taal $\mathcal{T}$. U zou kunnen denken dat dit probleem kan worden omzeild

door $V_{\mathcal{M}}$ alleen te definiëren voor de gesloten formules van $\mathcal{T}$, maar dat wil niet: gesloten formules waarin variabelen voorkomen zijn namelijk samengesteld uit kwantoren plus *open* formules, en de waarheidswaarde van de gesloten formule hangt dan af van de—minder complexe—open formule. De oplossing is als volgt. Neem aan dat we formules van $\mathcal{T}$ evalueren tegen de achtergrond van een zogenaamde *bedeling* of *toekenning* (Engels: *assignment*) voor de variabelen uit de taal $\mathcal{T}$. Een bedeling is een functie die de variabelen afbeeldt op objecten in het domein van het model. We kunnen zo'n bedeling b aangeven door—gegeven een volgorde van de variabelen—de objecten te noemen waar de variabelen stuk voor stuk aan worden gekoppeld. Dus bij voorbeeld:

$$\langle d_1, d_3, d_5, d_3, d_{101}, d_2, \ldots \rangle$$

waarbij $d_1, d_2, d_3, \ldots$ objecten uit D zijn. Merk op dat een object uit D aan meerdere variabelen kan worden toebedeeld.

Omdat de valuatie $V_{\mathcal{M}}$ die we willen gaan definiëren niet alleen van $\mathcal{M}$ maar ook van een bedeling b afhangt, stappen we over van de notatie $V_{\mathcal{M}}$ naar $V_{\mathcal{M},b}$. De functie $V_{\mathcal{M},b}$ die we willen gaan definiëren moet formules gaan afbeelden op waarheidswaarden. Het definiëren van de functie $V_{\mathcal{M},b}$ is niets anders dan: het geven van een *waarheidsdefinitie* voor de predikatenlogica. Deze *waarheidsdefinitie* voor predikatenlogische formules is van Alfred Tarski.

Eerst moeten we gaan kijken hoe de termen (constanten en variabelen) worden geïnterpreteerd. Voor de termen hebben we een functie nodig die objecten uit D als waarden oplevert. Dit heet: een waardetoekenning aan de termen van de taal. Die functie noemen we $W_{\mathcal{M},b}$. De definitie is als volgt. Als t een constante is, dan nemen we gewoon het object dat I aan die constante toekent. Dus:

- $W_{\mathcal{M},b}(t) = I(t)$, als t een constante is.

In het andere geval, dus als t een variabele is, hebben we de bedeling b nodig. Immers, I kent aan de variabelen geen objecten toe; de variabelen hadden juist geen vaste interpretatie. Dus:

- $W_{\mathcal{M},b}(t) = b(t)$, als t een variabele is.

Goed, nu gaan we met behulp van $W_{\mathcal{M},b}$ en I een definitie leveren van $V_{\mathcal{M},b}$. Dat gaat weer recursief, net als de definitie van de valuatiefunctie in de propositielogica.

Eerst het atomaire geval. We willen dat een formule zoals Pa waar is in een model $\mathcal{M}$ precies wanneer het object in $\mathcal{M}$ dat a wordt genoemd in de interpretatie van P zit. En voor Px: dit moet waar zijn in $\mathcal{M}$, gegeven een

bedeling b, wanneer het object dat door de bedeling aan x wordt toegekend, een element is van de interpretatie van P.

Nu algemeen. De algemene vorm van een atomaire formule is $At_1 \dots t_n$, dus:

- $V_{\mathcal{M},b}(At_1 \dots t_n) = 1 \iff \langle W_{\mathcal{M},b}(t_1), \dots, W_{\mathcal{M},b}(t_n)\rangle \in I(A)$.

Dan voor de niet-atomaire formules. De waarheidsfunctionele voegwoorden leveren geen problemen op. Die gaan precies zoals we op grond van het propositielogische geval zouden verwachten:

- $V_{\mathcal{M},b}(\neg\varphi) = 1 \iff V_{\mathcal{M},b}(\varphi) = 0$
- $V_{\mathcal{M},b}((\varphi \wedge \psi)) = 1 \iff V_{\mathcal{M},b}(\varphi) = V_{\mathcal{M},b}(\psi) = 1$
- $V_{\mathcal{M},b}((\varphi \vee \psi)) = 1 \iff V_{\mathcal{M},b}(\varphi) = 1$ of $V_{\mathcal{M},b}(\psi) = 1$
- $V_{\mathcal{M},b}((\varphi \to \psi)) = 1 \iff V_{\mathcal{M},b}(\varphi) = 0$ of $V_{\mathcal{M},b}(\psi) = 1$
- $V_{\mathcal{M},b}((\varphi \leftrightarrow \psi)) = 1 \iff V_{\mathcal{M},b}(\varphi) = V_{\mathcal{M},b}(\psi)$.

Nu de kwantoren nog. Eerst een simpel voorbeeld. Stel, we willen uitleggen wanneer $\exists x Px$ waar is. We willen daarbij gebruik maken van wat we weten over de waarheid van Px. We kunnen *niet* simpelweg zeggen: $V_{\mathcal{M},b}(\exists x Px) = 1$ desda $V_{\mathcal{M},b}(Px) = 1$. Immers, $V_{\mathcal{M},b}(Px)$ hangt af van de waarde die b toekent aan x: $V_{\mathcal{M},b}(Px) = 1$ desda $b(x) \in I(P)$. Nu zou het echter kunnen zijn dat weliswaar $b(x)$ de eigenschap $I(P)$ niet heeft, maar een *ander* object uit D wel. Dus: we moeten niet alleen kijken naar wat bedeling b met x doet, maar we moeten ook *andere* bedelingen in de beschouwingen betrekken.

Met name zijn we geïnteresseerd in bedelingen die van b alleen verschillen wat betreft de waarde die ze aan x toekennen. Het is handig om hiervoor een notatie in te voeren. We spreken daarom af: $b(x|d)$ staat voor de bedeling die precies is als b, behalve dat de waarde voor het argument x het object d is (waar b mogelijkerwijs een andere waarde toekende). Voor de lezers die een precieze definitie wensen:

$$b(x|d)(y) = \begin{cases} b(y) & \text{als } y \neq x \\ d & \text{als } y = x. \end{cases}$$

Gewapend met de nieuwe notatie gaan we nu de kwantoren te lijf.

- $V_{\mathcal{M},b}(\exists v\varphi) = 1 \iff$ voor *tenminste één* $d \in D : V_{\mathcal{M},b(v|d)}(\varphi) = 1$.
- $V_{\mathcal{M},b}(\forall v\varphi) = 1 \iff$ voor *alle* $d \in D : V_{\mathcal{M},b(v|d)}(\varphi) = 1$.

Hiermee hebben we alle mogelijkheden gehad, en de waarheidsdefinitie voor predikatenlogische formules is compleet. Merk op dat—net als bij de propositielogica—de recursieve waarheidsdefinitie de recursieve definitie van de verzameling welgevormde formules op de voet volgt.

Helaas, er zijn allerlei alternatieve notaties in zwang voor $V_{\mathcal{M},b}$. In plaats van $V_{\mathcal{M},b}(\varphi)$ schrijft men wel:

$$[\![\varphi]\!]_{\mathcal{M},b}$$

$V_{\mathcal{M},b}(\varphi) = 1$ wordt wel genoteerd als

$$\models_{\mathcal{M}} \varphi\ [b],$$

of als

$$\mathcal{M} \models \varphi\ [b].$$

Het is maar dat u het weet. Verder zullen we, wanneer het model $\mathcal{M}$ vastligt, de index bij V weglaten.

Net als in de propositielogica wordt $\models$ gebruikt om *logische waarheid* en *logisch gevolg* weer te geven. Bij open formules moeten we rekening houden met bedelingen. Dus:

$$\models \varphi.$$

Dit betekent: φ is logisch waar (of: *universeel geldig*), dat wil zeggen: φ is waar in alle modellen van de taal, voor alle mogelijke bedelingen. Let op: $\models \varphi$ moet niet worden verward met $\models_{\mathcal{M}} \varphi$ of $\mathcal{M} \models \varphi$.

$$\varphi \models \psi.$$

Dit betekent: ψ volgt logisch uit φ, dat wil zeggen voor alle modellen en bedelingen die φ waar maken, is ook ψ waar.

Hier zijn enkele voorbeelden van universeel geldige formules:

- $\forall x(Px \vee \neg Px)$
- $\forall x Px \leftrightarrow \neg\exists x \neg Px$
- $\forall x(Px \wedge Qx) \rightarrow \forall x Px$
- $Pa \rightarrow \exists x Px$.

Vanwege het onderscheid tussen open en gesloten formules hebben we nog wat jargon nodig.

Definitie 6.7 *Een formule φ heet* **vervulbaar** *(Engels:* satisfiable*) in een model $\mathcal{M}$ wanneer er een bedeling b is zo dat $\mathcal{M} \models \varphi\ [b]$.*

We zeggen in zo'n geval dat de bedeling b de formule φ vervult in $\mathcal{M}$.

Definitie 6.8 *Een formuleverzameling* Γ *is* **vervulbaar** *in een model* $\mathcal{M}$ *wanneer er een bedeling* b *is zodat voor elke* $\varphi \in \Gamma$*:* $\mathcal{M} \models \varphi[b]$.

Definitie 6.9 *Een formuleverzameling* Γ *waarvoor er een model* $\mathcal{M}$ *bestaat zo dat* Γ *vervulbaar is in* $\mathcal{M}$ *heet kortweg* **vervulbaar**.

We schakelen nu voor het gemak even over van willekeurige formules naar *gesloten* formules (predikatenlogische zinnen).

Definitie 6.10 *Twee gesloten formules die waar zijn in precies dezelfde modellen heten* **logisch equivalent**.

Voorbeeld: $\forall x Px$ en $\neg\exists x\neg Px$ zijn logisch equivalent. Men noemt een model waarin φ waar is wel *een model van* φ. Dus: twee zinnen φ en ψ zijn logisch equivalent wanneer ze dezelfde modellen hebben. We kunnen nu ook spreken over de klasse van modellen van een *verzameling* zinnen. MOD(Γ) duidt bij voorbeeld de klasse aan van alle modellen voor de taal in kwestie waarin alle zinnen uit de zinnenverzameling Γ waar zijn.

We hebben hierboven bij het uitleggen van het semantische effect van kwantificatie gebruik gemaakt van bedelingen voor variabelen. Deze bedelingen waren nodig omdat—in het algemeen—niet elk ding in het domein van het model een naam hoeft te hebben. Wanneer wel elk ding in het domein van het model een naam heeft, dat wil zeggen wanneer er voor elk object $d \in D$ een constante c uit de taal is zo dat $I(c) = d$, dan kunnen we de werking van kwantificatie uitleggen zonder gebruik te maken van bedelingen.

Terwille van die uitleg maken we eerst een nieuwe notatieafspraak. We spreken af dat $[c/v]\varphi$ betekent: het resultaat van vervanging (substitutie) van alle *vrije voorkomens* van variabele v in φ door constante c. Enkele voorbeelden:

$[a/x]Rxy$	staat voor	Ray
$[a/x]Rxx$	staat voor	Raa
$[a/x](Rxy \wedge \exists x Px)$	staat voor	$Ray \wedge \exists x Px$
$Rxy \wedge [a/x]Px$	staat voor	$Rxy \wedge Pa$.

Opdracht 6.22 *Welke formules worden aangeduid met:*

1. $[a/x](\exists x Rxx \wedge Px)$
2. $[a/x]\exists x \exists y Rxy \rightarrow Px$
3. $[b/y]\exists x \exists y (Rxy \wedge Py)$

4. $[b/y](\exists x \exists y Rxy \wedge Py)$
5. $[a/x]\forall x \forall y Rxy \rightarrow Px$
6. $[a/x](\forall x \forall y Rxy \rightarrow [b/x]Px)$
7. $\exists x Px \wedge [a/x]\exists y Rxy$.

Opdracht 6.23 *Geef een recursieve definitie van de operatie* $[e/v]$.

Voor atomaire formules die vrije variabelen bevatten moeten we nu een afspraak maken die de bedelingen omzeilt. Dit is alleen een kwestie van een knoop doorhakken: we spreken gewoon af dat een atomaire formule waarin variabelen voorkomen waar is in een model $\mathcal{M}$ wanneer er constanten voor die variabelen kunnen worden gesubstitueerd met als resultaat een atomaire formule met alleen constanten die waar is in het model $\mathcal{M}$. Die substitutie moet natuurlijk wel *uniform* gebeuren, dat wil zeggen dezelfde constante moet worden gesubstitueerd voor *alle* voorkomens van een bepaalde variabele, zodat substitutie van a voor x in $Gxyx$ oplevert: $Gaya$. Het kwantor-geval gaat nu als volgt:

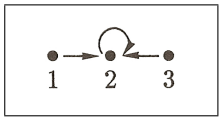
- $V_{\mathcal{M}}(\exists v\varphi) = 1 \iff V_{\mathcal{M}}([c/v]\varphi) = 1$ voor minstens één constante c van de taal.
- $V_{\mathcal{M}}(\forall v\varphi) = 1 \iff V_{\mathcal{M}}([c/v]\varphi) = 1$ voor elke constante c van de taal.

U ziet het, de bedelingen zijn nu overbodig geworden.

Wanneer we ons beperken tot modellen waarin elk object een naam heeft, terwijl bovendien die modellen *eindig* zijn, dan kan het *nog* simpeler. We mogen nu wel aannemen dat de taal ook maar eindig veel constanten heeft, anders zou er minstens één object zijn dat oneindig veel namen heeft, en dat is in een niet-theologische context een beetje overdreven. Dus: de taal heeft een verzameling constanten $c_1, \ldots, c_n$. We kunnen nu de kwantoren volledig *elimineren*, en wel als volgt.

- Vervang $\forall v\varphi$ door $[c_1/v]\varphi \wedge [c_2/v]\varphi \wedge \cdots \wedge [c_n/v]\varphi$.
- Vervang $\exists v\varphi$ door $[c_1/v]\varphi \vee [c_2/v]\varphi \vee \cdots \vee [c_n/v]\varphi$.

Dit ziet er misschien een beetje te abstract uit; daarom snel een simpel voorbeeld. Beschouw de volgende structuur:



We interpreteren R weer als de pijlrelatie, en we nemen aan dat de taal drie namen heeft, waarbij a object 1 benoemt, b object 2, en c object 3. De volgende bewering is waar in het model: $\exists x Rxx$. We kunnen nu echter evengoed zeggen: $Raa \vee Rbb \vee Rcc$. Dit drukt hetzelfde uit. De volgende bewering, die *niet* waar is in het model, kan eveneens worden geparafraseerd met behulp van constanten: $\forall x Rxx$. De parafrase wordt: $Raa \wedge Rbb \wedge Rcc$. Tot slot parafraseren we de formule $\forall x \exists y Rxy$ (waar in het model). Eerst elimineren we de universele kwantor: $\exists y Ray \wedge \exists y Rby \wedge \exists y Rcy$. Nu nog de existentiële kwantor:

$$(Raa \vee Rab \vee Rac) \wedge (Rba \vee Rbb \vee Rbc) \wedge (Rca \vee Rcb \vee Rcc).$$

U ziet: de beperking tot eindige modellen maakt van de predikatenlogica weer een soort van propositielogica. Dit heeft een belangrijke theoretische consequentie. Net als de propositielogica is de aldus 'gekortwiekte' predikatenlogica *beslisbaar*: er kan door het volgen van een mechanische procedure worden uitgemaakt of zekere formule logisch geldig is. Dit is een eigenschap die de predikatenlogica in het algemeen *niet* heeft (zie § 6.8).

6.4 Predikatenlogica met identiteit

Een van de manieren om de uitdrukkingskracht van de predikatenlogica belangrijk te vergroten is door het invoeren van het *identiteitsteken* (Engels: *identity sign*, *equality sign*). Voor identiteit gebruiken we het teken $=$. Dit is een tweeplaatsig predikaatsymbool, maar voor het gemak schrijven we $a = b$ in plaats van $= ab$. Het identiteitsteken heeft een vaste interpretatie, die gegeven wordt door de volgende waarheidsclausule (we stappen weer over naar het meest algemene geval, en maken dus gebruik van bedelingen):

- $V_{\mathcal{M},b}(t_1 = t_2) = 1 \iff W_{\mathcal{M},b}(t_1) = W_{\mathcal{M},b}(t_2)$.

Volgens deze definitie is $a = b$ waar precies in die gevallen waarin a en b hetzelfde object benoemen.

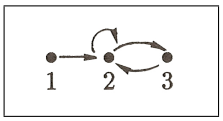
Hoe handig het invoeren van het identiteitsteken is blijkt uit de volgende parafrasen van Nederlandse zinnen:

'Annie houdt alleen van Bhagwan'	$\forall x(Hax \leftrightarrow x = b)$
'Alleen Bhagwan houdt van Annie'	$\forall x(Hxa \leftrightarrow x = b)$
'Piet houdt van iedereen behalve van Bhagwan'	$\forall x(Hpx \leftrightarrow \neg x = b)$
'Iedereen houdt van Bhagwan behalve Piet'	$\forall x(Hxb \leftrightarrow \neg x = p)$.

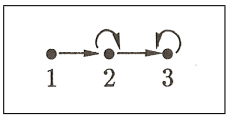
Opdracht 6.24 *Vertaal nu zelf: 'Piet houdt van Annie, maar Annie houdt van een ander'.*

Nog iets moeilijker is: 'Alleen Piet houdt alleen van Annie'. De vertaling wordt: $\forall x(\forall y(Hxy \leftrightarrow y = a) \leftrightarrow x = p)$.

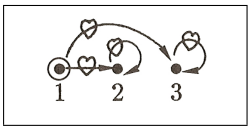
Opdracht 6.25 *Ga na of deze laatste formule waar is in het volgende model (neem aan dat p wordt geïnterpreteerd als individu 1, en a als individu 2, terwijl individu 3 naamloos is):*



Opdracht 6.26 *Zelfde vraag voor het volgende model (de interpretatie van de constanten blijft hetzelfde):*



Opdracht 6.27 *Beschouw het volgende model:*



De pijl $\multimap$ geeft aan wie van wie houdt; $\bigcirc$ geeft aan wie zielig is. De taal die we bekijken heeft een eenplaatsige predikaatletter Z voor 'zielig zijn' en een tweeplaatsige predikaatletter H voor 'houden van'.

Bepaal voor elk van de nu volgende formules eerst of de formule waar is in het model, en geef vervolgens een parafrase van de formules in het Nederlands.

1. $\forall x \exists y Hxy$
2. $\forall x \exists y(\neg x = y \wedge Hxy)$
3. $\forall x(\neg Hxx \rightarrow Zx)$
4. $\exists x \exists y(Hxy \wedge \neg x = y \wedge Zx)$
5. $\forall x(Hxx \rightarrow \neg \exists y(\neg x = y \wedge Hxy)$
6. $\forall x \forall y((Hxy \wedge \neg x = y) \rightarrow Zx)$.

Hoe krachtig = is als uitdrukkingsmiddel blijkt uit het feit dat we nu ook kunnen *tellen*: 'Annie houdt van minstens twee jongens' wordt:

$$\exists x \exists y(\neg x = y \wedge Jx \wedge Jy \wedge Hax \wedge Hay).$$

Wat 'Annie houdt van minstens vierentwintig jongens' wordt, kunt u nu zelf bedenken. Dat het er knap ingewikkeld gaat uitzien blijkt wel uit het feit dat 'er zijn minstens drie x zo dat Px' al niet korter kan dan (we korten $\neg x = y$ af tot $x \neq y$):

$$\exists x \exists y \exists z(x \neq y \wedge x \neq z \wedge y \neq z \wedge Px \wedge Py \wedge Pz).$$

'Er is hoogstens één x zo dat Px' wordt:

$$\forall x \forall y((Px \wedge Py) \rightarrow x = y).$$

Heel fraai is, dat we nu *uniciteit* kunnen uitdrukken. 'Er is precies één x zo dat Px' wordt:

$$\exists x(Px \wedge \forall y(Py \rightarrow y = x)),$$

of, in een iets beknoptere, maar equivalente formulering:

$$\exists x \forall y(Py \leftrightarrow x = y).$$

Bertrand Russell heeft voorgesteld om deze truc te gebruiken om het *bepaald lidwoord* te parafraseren. Zijn beroemde *descriptietheorie* komt neer op het volgende. Lees "*De* koning toornt" als: "Er is precies één x die koning is, en x toornt". Predikaatlogisch wordt dit:

$$\exists x(Kx \wedge \forall y(Ky \rightarrow y = x) \wedge Tx)$$

of equivalent:

$$\exists x(\forall y(Ky \leftrightarrow y = x) \wedge Tx).$$

U ziet het: het zinsdeel *de koning* is in de parafrase niet meer als zelfstandig onderdeel terug te vinden. Datzelfde gold trouwens ook al voor nominale constituenten als *elke jongen* en *sommige meisjes*, die als sneeuw voor de zon verdwijnen bij een vertaling naar de predikatenlogica. Dit heet: *contextuele eliminatie* van syntactische constituenten.

Russell paste zijn descriptietheorie toe om onderscheid te kunnen maken tussen verschillende lezingen van zinnen waarin uniek bepalende beschrijvingen (Engels: *definite descriptions*) voorkomen in samenhang met logische operatoren zoals de negatie-operator. De eliminatie-procedure voor descripties is contextueel. Dit houdt in dat we volgens Russells descriptietheorie "De koning toornt niet" op verschillende manieren kunnen lezen.

Afhankelijk van de vraag wat we als de context beschouwen waaruit *de koning* moet worden geëlimineerd krijgen we een andere predikatenlogische formule als vertaling.

De eerste mogelijkheid is: eliminatie met de hele zin als context. We kunnen nu het bereik van de uniek bepalende beschrijving als volgt aangeven:

- De koning toornt niet.

Dit leidt tot de volgende predikatenlogische formule:

$$\exists x(\forall y(Ky \leftrightarrow x = y) \wedge \neg Tx).$$

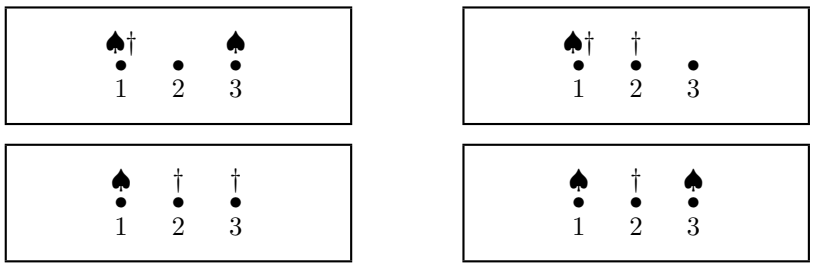
We kunnen echter ook besluiten alleen het predikaat *toornen* als eliminatie-context te beschouwen. De negatie-operator valt dan buiten de context waaruit de beschrijving wordt geëlimineerd, ofwel: de beschrijving heeft nu kleiner bereik dan de negatie-operator. Schematisch:

- De koning toornt niet.

De formele weergave wordt nu:

$$\neg\exists x(\forall y(Ky \leftrightarrow x = y) \wedge Tx).$$

Opdracht 6.28 *Beschouw de volgende modellen:*



Neem aan dat ♠ de eenplaatsige predikaatletter K interpreteert, en † de eenplaatsige predikaatletter T. Geef voor elk van de twee lezingen van "De koning toornt niet" die Russell onderscheidt aan in welke van deze vier modellen zij waar is.

De descriptietheorie van Russell is overigens niet onweersproken gebleven. Zie voor filosofische kritiek: [Strawson 1950]. Strawson stelt als alternatief voor Russells theorie een descriptietheorie voor waarin de uniciteit (in een zekere context) van datgene wat door de beschrijving wordt aangeduid niet wordt *uitgedrukt* in de formele weergave, zoals bij Russell gebeurt, maar

wordt *verondersteld*. Dit voorstel wordt vaak door taalkundigen als uitgangspunt genomen voor een zogenaamde presuppositietheorie van uniek bepalende beschrijvingen. Een vergelijking tussen de visies van Russell en Strawson vindt u in [Gamut 1982, deel 1].

6.5 Functie-symbolen

We hebben gezien hoe de invoering van het identiteitsteken de uitdrukkingskracht van de predikatenlogica vergroot. We gaan nu kort in op een tweede manier om de uitdrukkingskracht van de predikatenlogica te vergroten: het invoeren van *functie-constanten*, namen van een- of meerplaatsige operaties op het individuendomein.

Met functies en meer in het bijzonder met operaties hebben we kennis gemaakt in § 2.8. We recapituleren. Laat een verzameling D gegeven zijn. Een eenplaatsige operatie op D is nu een functie die elementen van D als argumenten neemt (elk element van D kan daarbij als argument optreden), om die elementen af te beelden op elementen van D, de waarden (niet elk element van D hoeft als waarde op te treden). Een tweeplaatsige operatie op D is een functie die *paren* van elementen uit D afbeeldt op elementen van D. Enzovoorts voor drieplaatsige, vierplaatsige,... operaties. Een voorbeeld van een tweeplaatsige operatie op de natuurlijke getallen is de operatie *optellen*. Deze operatie beeldt elk paar van natuurlijke getallen af op een natuurlijk getal, te weten: de som van die getallen.

We breiden nu onze predikatenlogische talen uit met een verzameling van namen voor eenplaatsige operaties, een verzameling van namen voor tweeplaatsige operaties, enzovoorts. Deze uitbreiding betekent dat we nu individuen kunnen aanduiden op nieuwe manieren, en niet meer alleen met behulp constanten of variabelen. Laat f een naam zijn voor een eenplaatsige operatie (we noemen dit: 'een eenplaatsig functiesymbool', of 'een eenplaatsige functieconstante'). Wanneer x en a respectievelijk een variabele en een constante zijn, dan zijn niet alleen x en a manieren om een individu aan te duiden, maar ook fx [of: $f(x)$] en fa [of: $f(a)$] duiden individuen aan. Wanneer h een tweeplaatsig functiesymbool is, dan duidt hax [of: $h(a, x)$] een individu aan.

Hier volgt een schets van de formele uitwerking. De definitie van een *term* wordt nu iets ingewikkelder, en vraagt om een recursieve formulering:

Definitie 6.11 Termen *van een predikatenlogische taal $\mathcal{T}$ met functiesymbolen:*

- *elke variabele of constante is een term;*

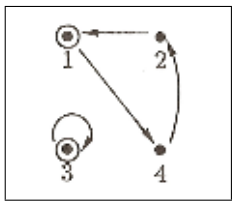
- *als g een n-plaatsig functiesymbool is, en $t_1, \ldots, t_n$ zijn termen, dan is $gt_1 \ldots t_n$ ook een term;*
- *niets anders is een term.*

Neem aan dat f een eenplaatsig functiesymbool is, en g een tweeplaatsig functiesymbool; x en y zijn individuele variabelen; a en b zijn constanten. Hier zijn een aantal voorbeelden van termen: x, a, fa, $fffa$, $gxfb$, $fgxfb$, $ggabgffax$, $ffgffafb$, $gagagagagay$. Ga na dat deze voorbeelden aan de definitie voldoen. Soms worden termen waarin functiesymbolen voorkomen met haakjes geschreven: $f(a)$, $g(x, f(b))$, $f(g(x, f(b)))$, enzovoorts. Strikt genomen is dit echter niet nodig, want door de prefix notatie van de functiesymbolen (dat wil zeggen: Poolse notatie) zijn de termen ook zonder haakjes ondubbelzinnig.

De waarde-definitie voor termen moet nu ook worden aangepast. We moeten nu aannemen dat de interpretatiefuncties van modellen voor een predikatenlogische taal $\mathcal{T}$ waarin functiesymbolen voorkomen zo zijn ingericht dat elk van de functiesymbolen wordt afgebeeld op een operatie met het juiste aantal argumenten. Die waarde-definitie maakt gebruik van wat de interpretatiefunctie met de functie-symbolen doet, en dit werkt vervolgens door in de waarheidsdefinitie.

In de volgende opdrachten kunt u laten zien dat u intuïtief al weet hoe het zit met waarde van termen en waarheid van formules waarin functiesymbolen voorkomen.

Opdracht 6.29 *Beschouw het volgende model:*



Neem aan dat de pijlen het eenplaatsige functiesymbool f interpreteren (ga na dat dit inderdaad een eenplaatsige operatie is op het domein van de structuur), en de cirkels de eenplaatsige predikaatletter P. De constante a benoemt object 1, b benoemt 2, c benoemt 3, en d benoemt 4. Welk object wordt aangeduid met:

1. fa
2. ffb
3. $fffc$

4. $ffffd$.

Opdracht 6.30 *De gegevens zijn als in de vorige opdracht. Welke van de volgende formules zijn waar in de structuur:*

1. $\exists x P f x$
2. $\exists x P f f x$
3. $\forall x(Px \vee P f f x)$
4. $\exists x(Px \wedge P f f x)$
5. $\forall x(P f x \rightarrow \neg Px)$
6. $\forall x(Px \rightarrow (P f x \rightarrow P f f x))$.

De algemene formulering van de waarde-definitie $W_{\mathcal{M},b}$ voor een predikatenlogische taal $\mathcal{T}$ met functiesymbolen (waarbij $\mathcal{M}$ een model is voor $\mathcal{T}$ en b een bedeling voor de variabelen uit $\mathcal{T}$ in $\mathcal{M}$) wordt:

- $W_{\mathcal{M},b}(t) = I(t)$ wanneer t een constante is;
- $W_{\mathcal{M},b}(t) = b(t)$ wanneer t een variabele is;
- $W_{\mathcal{M},b}(t) = I(g)(W_{\mathcal{M},b}(t_1), \ldots, W_{\mathcal{M},b}(t_n))$ wanneer t van de vorm $gt_1 \ldots t_n$ is.

Deze recursieve waarde-definitie voor termen volgt—zoals u natuurlijk ook verwacht had—weer precies de recursieve definitie van de termen zelf. Aan de waarheidsdefinitie hoeft niets te veranderen.

6.6 Semantische tableaus voor de predikatenlogica

In de propositielogica konden semantische tableaus worden gebruikt om in een eindig aantal stappen de geldigheid van een redenering te testen. Er gold: *of* een redenering geldig is blijkt in een eindig aantal stappen. In de predikatenlogica liggen de zaken minder eenvoudig. Hier kunnen we ook semantische tableaus gebruiken, maar er is nu geen garantie dat we er in een eindig aantal stappen achterkomen of een redenering geldig is. Wel geldt het volgende: *als* een redenering geldig is, dan blijkt dat in een eindig aantal stappen. Het verschil is subtiel, maar het zal u hopelijk in de loop van deze paragraaf duidelijk worden.

We introduceren de tableau-methode voor predikatenlogische formules weer aan de hand van voorbeelden.

Voorbeeld 6.1 Is $\forall x(Ax \to Bx), \forall x(Bx \to Cx) \models \forall x(Ax \to Cx)$?

Stel van niet:

waar	onwaar
$\forall x(Ax \to Bx), \forall x(Bx \to Cx)$	$\forall x(Ax \to Cx)$

Om $\forall x(Ax \to Cx)$ onwaar te laten zijn moeten we aannemen dat er minstens één individu d_1 is waarvoor $Ad_1 \to Cd_1$ onwaar is. De tweede stap is nu gewoon propositioneel-logisch: toepassen van de regel voor $\to$-rechts. Daarna moeten de twee universeel gekwantificeerde formules links nog worden behandeld. Wil een universele bewering waar zijn, dan moet zij zeker gelden voor het individu d_1 dat we in de eerste stap hebben ingevoerd. Dit geeft het volgende tableau:

<table>
<tr><td colspan="2">waar
$\forall x(Ax \to Bx), \forall x(Bx \to Cx)$

Ad_1
$Ad_1 \to Bd_1, Bd_1 \to Cd_1$</td><td colspan="3">onwaar
$\forall x(Ax \to Cx)$
$Ad_1 \to Cd_1$
Cd_1</td></tr>
<tr><td>waar</td><td>onwaar
Ad_1</td><td colspan="2">waar
Bd_1</td><td>onwaar</td></tr>
<tr><td></td><td></td><td>waar</td><td>onwaar
Bd_1</td><td>waar
Cd_1</td><td>onwaar</td></tr>
</table>

Het tableau sluit: de redenering is geldig. Hiermee hebben we de geldigheid van Aristoteles' beroemdste syllogisme-figuur, de figuur BARBARA, aangetoond. BARBARA is de naam waarmee de Middeleeuwse scholastici het stramien van de volgende redenering aanduidden:

Alle Grieken zijn mensen.
Alle mensen zijn sterfelijk.

Alle Grieken zijn sterfelijk.

Voorbeeld 6.2 Is $\forall x(Ax \to Bx), \exists x(Ax \wedge Cx) \models \exists x(Cx \wedge Bx)$?

We nemen weer aan van niet:

waar	onwaar
$\forall x(Ax \to Bx), \exists x(Ax \wedge Cx)$	$\exists x(Cx \wedge Bx)$

Om de existentiële formule $\exists x(Ax \wedge Cx)$ waar te maken moet voor minstens één object d_1 gelden dat $Ad_1 \wedge Cd_1$. Om de universele formule $\forall x(Ax \to Bx)$ waar te maken moet voor elk individu in het domein, dus zeker voor d_1, de implicatie gelden. Om de existentiële conjunctie $\exists x(Cx \wedge Bx)$ *onwaar* te maken moet voor *elk individu*, dus zeker ook weer voor d_1, gelden dat de conjunctie onwaar is. Een en ander leidt tot het volgende tableau:

<table>
<tr><td colspan="2">waar
$\forall x(Ax \to Bx), \exists x(Ax \wedge Cx)$
$Ad_1 \wedge Cd_1$
Ad_1, Cd_1
$Ad_1 \to Bd_1$</td><td colspan="4">onwaar
$\exists x(Cx \wedge Bx)$</td></tr>
<tr><td>waar</td><td>onwaar
Ad_1</td><td colspan="2">waar
Bd_1</td><td colspan="2">onwaar
$Cd_1 \wedge Bd_1$</td></tr>
<tr><td rowspan="2"></td><td rowspan="2"></td><td>waar</td><td>onwaar
Cd_1</td><td>waar</td><td>onwaar
Bd_1</td></tr>
</table>

Ook dit tableau sluit: kennelijk is de redenering geldig. Daarmee is de geldigheid aangetoond van een ander Aristotelisch syllogisme, namelijk het stramien van:

Alle Grieken zijn mensen.
Minstens één Griek is kaal.

Minstens één mens is kaal.

Voorbeeld 6.3 Is $\exists x(Gx \wedge Kx), \exists x(Bx \wedge Kx) \models \exists x(Bx \wedge Gx)$?

Dit is het stramien van de volgende redenering:

Minstens één Griek is kaal.
Minstens één barbaar is kaal.

Minstens één barbaar is een Griek.

We nemen weer aan dat de redenering niet geldig is; deze aanname leidt tot het volgende tableau (toelichting volgt):

<table>
<tr><td colspan="4">waar
$\exists x(Gx \wedge Kx), \exists x(Bx \wedge Kx)$
$Gd_1 \wedge Kd_1$
Gd_1, Kd_1

$Bd_2 \wedge Kd_2$
Bd_2, Kd_2</td><td colspan="2">onwaar
$\exists x(Bx \wedge Gx)$

$Bd_1 \wedge Gd_1$

$Bd_2 \wedge Gd_2$</td></tr>
<tr><td colspan="2">waar</td><td colspan="2">onwaar
Bd_1</td><td>waar</td><td>onwaar
Gd_1</td></tr>
<tr><td>waar</td><td>onwaar
Bd_2</td><td>waar</td><td>onwaar
Gd_2</td><td rowspan="2"></td><td rowspan="2"></td></tr>
</table>

Twee subtableaus sluiten, maar het derde subtableau blijft open. De redenering is ongeldig.

Een korte toelichting bij de wijze waarop het tableau tot stand is gekomen is op zijn plaats. De volgorde waarin de formules boven in het tableau zijn behandeld, is als volgt. Eerst leidt de behandeling van de eerste existentiële formule links tot invoering van een d_1. Om de existentiële formule rechts onwaar te maken moet wat die formule zegt *niet* opgaan voor d_1, dat wil zeggen: $Bd_1 \wedge Gd_1$ moet onwaar worden. Vervolgens behandelen we de tweede existentiële formule links. Er moet een object d_2 zijn waarvoor de conjunctie $Bd_2 \wedge Kd_2$ waar is. We hadden hier ook het object d_1 voor kunnen nemen, maar dan lopen we het gevaar dat we dat ene object overbelasten door een combinatie van te zware eisen: dat gaat bij voorbeeld zeker fout bij het waar maken van existentiële formules als $\exists xKx \wedge \exists x\neg Kx$. Om dit soort gevaren te omzeilen spreken we af dat we voor elke existentiële formule links een *nieuw* individu zullen introduceren. Nu er een tweede individu is geïntroduceerd moeten we, om $\exists x(Bx \wedge Gx)$ onwaar te doen zijn, ook eisen dat voor d_2 de formule $Bd_2 \wedge Gd_2$ onwaar is. De stappen die daarna volgen zijn propositielogisch.

Uit het open subtableau in bovenstaand tableau kan als volgt een tegenvoorbeeld worden afgelezen. Als we de atomaire formules in dat subtableau langslopen dan zien we dat het subtableau een situatie beschrijft waarin twee dingen allebei de eigenschap K hebben; een ding heeft bovendien de eigenschap B, maar mist G, en het andere heeft G maar mist B. Twee kale individuen, een Griek en een barbaar. Formeel ziet het tegenvoorbeeld tegen de redenering er als volgt uit: $D = \{1, 2\}$, $I(K) = \{1, 2\}$, $I(G) = \{1\}$, $I(B) = \{2\}$. Merk op dat strikt genomen de d_1, d_2 in het tableau *individuele constanten* zijn (het zijn immers ingrediënten van formules), terwijl 1 en 2 de objecten in het domein van het tegenvoorbeeld $\mathcal{M} = \langle D, I\rangle$ zijn waarnaar die constanten verwijzen.

Voorbeeld 6.4 Is $\exists xAx \models \forall xAx$?

Nee, want het tableau sluit niet:

waar	onwaar
$\exists xAx$	$\forall xAx$
Ad_1	
	Ad_2

Merk op dat het toepassen van $\forall$-rechts op de d_1 die reeds aanwezig is hier ten onrechte tot sluiting zou hebben geleid. Het volgende is dus FOUT:

waar	onwaar
$\exists xAx$	$\forall xAx$
Ad_1	
	Ad_1

Moraal: $\forall$-rechts en $\exists$-links moeten altijd worden toegepast op een *nieuwe* d_i.

We sommen de regels voor het behandelen van de kwantoren op:

- $\exists$-links: voer een *nieuwe* d_i in, en eis dat $\varphi(d_i)$ waar wordt:

waar	onwaar
$\exists v\varphi$	
$\varphi(d_i)$	

- $\forall$-rechts: Voer een *nieuwe* d_i in, en eis dat $\varphi(d_i)$ onwaar wordt:

waar	onwaar
	$\forall v\varphi$
	$\varphi(d_i)$

Merk op dat $\forall$-rechts en $\exists$-links elkaars spiegelbeeld zijn. Net zo zijn $\forall$-links en $\exists$-rechts elkaars spiegelbeeld.

- $\exists$-rechts: eis voor alle in het domein aanwezige objecten d_i dat $\varphi(d_i)$ onwaar wordt:

waar	onwaar
	$\exists v\varphi$
	$\varphi(d_i)$

- $\forall$-links: eis voor alle in het domein aanwezige objecten d_i dat $\varphi(d_i)$ waar wordt:

waar	onwaar
$\forall v\varphi$	
$\varphi(d_i)$	

De regels $\exists$-rechts en $\forall$-links hebben de vorm van een instructie die terwijl het tableau zich ontwikkelt van kracht blijft; als er later nieuwe elementen worden ingevoerd—ten gevolge van het toepassen van $\forall$-rechts of $\exists$-links—dan moet de instructie ook op die objecten worden toegepast.

Voorbeeld 6.5 Is $\exists x\forall yRxy \models \forall y\exists xRxy$?

Ja, kijk maar naar het volgende tableau:

	waar	onwaar
	$\exists x \forall y Rxy$	$\forall y \exists x Rxy$
$\exists$-links:	$\forall y Rd_1 y$	
$\forall$-rechts:		$\exists x Rx d_2$
$\exists$-rechts:		$Rd_1 d_2$
$\forall$-links:	$Rd_1 d_2$	

$\exists x Rx d_2$ moet onwaar worden, en mag dus niet van d_1 gelden: $Rd_1 d_2$ wordt onwaar. $\forall y Rd_1 y$ moet waar worden en moet dus zeker ook van d_2 gelden: $Rd_1 d_2$ wordt waar. Het tableau sluit.

Net als bij propositielogische tableaus hebben we de volgorde waarin de regels worden toegepast voor een deel zelf in de hand. Bij de propositielogische regels is het slim om toepassen van regels die splitsing veroorzaken zo lang mogelijk uit te stellen. Bij de kwantor-regels is het handig om zo mogelijk $\exists$-links en $\forall$-rechts toe te passen *voor* $\forall$-links en $\exists$-rechts.

Voorbeeld 6.6 Is $\forall x Ax \models \exists x Ax$?

Stel van niet:

waar	onwaar
$\forall x Ax$	$\exists x Ax$

We zitten nu al meteen in een impasse: noch $\forall$-links, noch $\exists$-rechts kan worden worden toegepast, want die veronderstellen dat er al individuen zijn ingevoerd. In feite kunnen we hieruit meteen een heel flauw tegenvoorbeeld aflezen, namelijk het model met het lege domein: op het lege domein is $\forall x Ax$ waar (omdat er geen individuen zijn geldt elke eigenschap van elk individu), en $\exists x Ax$ is onwaar (want deze formule beweert dat er een individu bestaat).

Als we dit soort flauwe tegenvoorbeelden willen uitsluiten moeten we eisen dat de domeinen van onze modellen niet leeg zijn. Maar als we dat afspreken mogen we ook een individu d_1 als 'voorgift' nemen. Het tableau gaat er nu zo uitzien:

	waar	onwaar
	$\forall x Ax$	$\exists x Ax$
d_1 is voorgift		
$\forall$-links:	Ad_1	
$\exists$-rechts:		Ad_1

Het tableau sluit, en dat wil het volgende zeggen. Als we aannemen dat de domeinen van onze modellen niet leeg zijn, dan is de redenering geldig.

Voorbeeld 6.7 Is $\forall x(Ax \to Bx) \models \exists x(Ax \wedge Bx)$?

Als we het lege domein niet uitsluiten zitten we, om dezelfde reden als in het vorige voorbeeld, meteen in een impasse; het model met het lege domein is dan een tegenvoorbeeld tegen de redenering. Sluiten we het lege domein wel uit, dan mogen we weer beginnen met een individu d_1 als voorgift. Het tableau wordt nu:

	waar				onwaar			
	$\forall x(Ax \to Bx)$				$\exists x(Ax \wedge Bx)$			
$\exists$-rechts:					$Ad_1 \wedge Bd_1$			
$\forall$-links:	$Ad_1 \to Bd_1$							
	waar		onwaar		waar		onwaar	
$\to$-links:			Ad_1		Bd_1			
	waar	onw	waar	onw	waar	onw	waar	onw
$\wedge$-rechts:		Ad_1		Bd_1		Ad_1		Bd_1

Drie subtableaus blijven open. Elk van deze subtableaus levert een tegenvoorbeeld. Het subtableau helemaal links:

$$\mathcal{M} = \langle D, I\rangle, \text{ met } D = \{1\}, I(A) = \emptyset, I(B) = \emptyset.$$

Het tweede subtableau van links: zelfde tegenvoorbeeld. Het derde subtableau van links:

$$\mathcal{M} = \langle D, I\rangle, \text{ met } D = \{1\}, I(A) = \emptyset, I(B) = \{1\}.$$

Het bestaan van de tegenvoorbeelden toont aan dat een redenering volgens dit stramien, zoals bij voorbeeld

$$\frac{\text{Alle Grieken zijn sterfelijk.}}{\text{Minstens één Griek is sterfelijk.}}$$

ongeldig is. In de Aristotelische logica wordt deze redenering overigens wel als geldig beschouwd, maar dat komt omdat Aristoteles aanneemt dat geen enkel predikaat een lege extensie heeft.

Voorbeeld 6.8 Is $\forall x\exists yRxy \models \forall xRxx$?

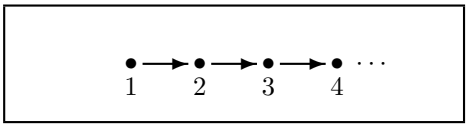
Het tableau wordt:

	waar	onwaar
	$\forall x\exists yRxy$	$\forall xRxx$
$\forall$-rechts:		Rd_1d_1
$\forall$-links:	$\exists yRd_1y$	
$\exists$-links:	Rd_1d_2	
$\forall$-links:	$\exists yRd_2y$	
$\exists$-links:	Rd_2d_3	
	$\vdots$	
	enzovoorts	

We zitten kennelijk in een oneindige lus: $\forall$-links gebiedt dat de formule $\exists y R d_i y$ waar is voor elk individu, en tengevolge van $\exists$-links komen er steeds nieuwe individuen bij. Omdat we weten *hoe* het tableau doorgaat kunnen we zien dat er zeker nooit sluiting zal optreden; er valt daarom uit het tableau een oneindig tegenvoorbeeld af te lezen:

$$\mathcal{M} = \langle D, I \rangle, \text{ met } D = \{1, 2, 3, \ldots\}, I(R) = \{\langle 1, 2\rangle, \langle 2, 3\rangle, \langle 3, 4\rangle, \ldots\}.$$

In een plaatje:



Opdracht 6.31 *De manier van tegenvoorbeelden construeren die we tot nu toe hebben gehanteerd is: in de linkerkolommen van de tableaus aflezen wat verplicht is en ervoor zorgen dat de interpretatie-functie daaraan voldoet. In plaats daarvan zouden we ook een andere strategie kunnen hanteren: in de rechterkolommen van de tableaus aflezen wat verboden is, en de interpretatiefunctie zo inrichten dat de extensies van de predikaten alles omvatten wat* niet *hoeft te worden uitgesloten. Construeer volgens dit principe een tegenvoorbeeld op grond van het tableau uit het bovenstaande voorbeeld.*

In feite hadden we bij de laatste redenering ook kunnen volstaan met een *eindig* tegenvoorbeeld. De reden voor het invoeren van telkens een nieuwe d_i bij $\exists$-links en $\forall$-rechts was dat bij het gebruik van een oude d_i ten onrechte sluiting zou kunnen optreden (vergelijk de opmerkingen bij voorbeelden 3. en 4.). Maar als we weten dat het tableau toch niet sluit geldt deze overweging niet meer. Om een oneindig tegenvoorbeeld om te zetten in een eindig loont het de moeite om de al aanwezige individuen zoveel mogelijk uit te buiten door $\exists$-links en $\forall$-rechts eerst op die individuen uit te proberen. Voor ons voorbeeld geeft dit het volgende tableau:

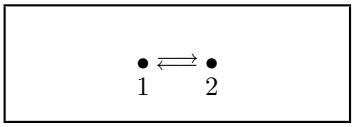
	waar	onwaar
	$\forall x \exists y R x y$	$\forall x R x x$
$\forall$-rechts:		$R d_1 d_1$
$\forall$-links:	$\exists y R d_1 y$	
$\exists$-links:	$R d_1 d_2$	
$\forall$-links:	$\exists y R d_2 y$	
$\exists$-links:	$R d_2 d_1$	

De eerste keer dat $\exists$-links wordt toegepast moeten we een nieuw individu d_2 invoeren: gebruik van de reeds aanwezige d_1 zou ten onrechte tot sluiting

leiden. $\forall$-links is ook voor d_2 van kracht, dus $\exists y R d_2 y$ moet waar zijn. Bij de dan volgende toepassing van $\exists$-links zondigen we tegen de boven gegeven regel (verderop volgt een aangepaste versie) en proberen we het reeds aanwezige individu d_1. Het enige motief voor de zonde is: het ontstaan van een oneindig tegenvoorbeeld vermijden. In dit geval hebben we geluk: Rd_2d_1 links plaatsen leidt *niet* tot sluiting. Er zijn nu geen regels meer toepasbaar, en het tableau sluit niet. Het tableau levert ons een eindig tegenvoorbeeld:

$$\mathcal{M} = \langle D, I \rangle, \text{ met } D = \{1, 2\}, I(R) = \{\langle 1, 2\rangle, \langle 2, 1\rangle\}.$$

Een plaatje:

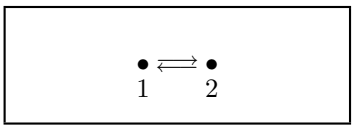


Voorbeeld 6.9 Is $\forall x \exists y R x y \models \exists y \forall x R x y$?

Nee, en de tableau-methode levert weer op systematische wijze een tegenvoorbeeld. We gaan uit van de eis dat het domein niet leeg mag zijn, en gebruiken dus d_1 als voorgift. Bij de eerste toepassing van $\exists$-links zou het gebruik van een al aanwezig individu tot sluiting leiden, en daarom mag het niet. Bij de tweede toepassing van $\exists$-links en bij de toepassingen van $\forall$-rechts leidt het gebruik van oude individuen niet tot sluiting, en daarom mag het(vergelijk de opmerking naar aanleiding van het vorige voorbeeld). Zo blijken we in staat te zijn om een eindig tegenvoorbeeld te construeren:

	waar	onwaar
	$\forall x \exists y R x y$	$\exists y \forall x R x y$
d_1 is voorgift		
$\forall$-links:	$\exists y R d_1 y$	
$\exists$-rechts:		$\forall x R x d_1$
$\forall$-rechts:		$R d_1 d_1$
$\exists$-links:	$R d_1 d_2$	
gebruik van d_1 zou hier ten onrechte tot sluiting leiden		
$\exists$-rechts:		$\forall x R x d_2$
$\forall$-rechts:		$R d_2 d_2$
$\forall$-links:	$\exists y R d_2 y$	
$\exists$-links:	$R d_2 d_1$	

Het tegenvoorbeeld in een plaatje:



Het proberen te werken met al aanwezige individuen levert de volgende amenderingen in de regels voor ∃-links en ∀-rechts op:

- ∃-links (probeer-versie):

  Plaats $\varphi(d_i)$ links voor een of andere al aanwezige d_i. Als dit lukt zonder dat sluiting optreedt, dan klaar. Als het voor elke al aanwezige d_i tot sluiting leidt, voer dan een nieuwe d_i in en plaats $\varphi(d_i)$ links.

- ∀-rechts (probeer-versie):

  Plaats $\varphi(d_i)$ rechts voor een of andere al aanwezige d_i. Als dit lukt zonder dat sluiting optreedt, dan klaar. Als het voor elke al aanwezige d_i tot sluiting leidt, voer dan een nieuwe d_i in en plaats $\varphi(d_i)$ rechts.

Als er bij de probeer-versie van de tableau-regels geen sluiting optreedt bij gebruik van bestaande objecten, en er geldt bovendien dat alle bestaande objecten zijn gebruikt bij toepassingen van ∃-rechts en ∀-links, dan betekent dit dat we erin geslaagd zijn om een *eindig* tegenvoorbeeld te construeren.

Het wordt zo langzamerhand tijd om zelf wat te gaan oefenen met het maken van predikatenlogische tableaus.

Opdracht 6.32 *Test de geldigheid van de volgende redeneringen. Geef beschrijvingen van de eventuele tegenvoorbeelden die u vindt.*

1. $\forall x(Ax \to Bx), \exists x Ax \models \exists x Bx$
2. $\models \forall x(((Ax \to Bx) \wedge (Bx \to Cx)) \to (Ax \to Cx))$
3. $\forall x(Ax \to Bx), \neg\exists x Bx \models \neg\exists x Ax$
4. $\forall x(Ax \to Bx), \neg\exists x Ax \models \neg\exists x Bx$
5. $\exists x \neg Ax \models \forall x(\forall y(Rxy \to Ax) \to \forall y Rxy)$
6. $\exists x \neg Ax \models \forall x(\forall y(Rxy \to Ax) \to \exists y \neg Rxy)$.

Opdracht 6.33 *Hoe, denkt u, zou men de individuele constanten die in predikatenlogische formules kunnen voorkomen moeten behandelen in een semantisch tableau? Geef een regel. Maak vervolgens een semantisch tableau om de predikatenlogische weergave van de volgende redenering te testen, en pas de regel daarin toe:*

Geen Griek veracht zichzelf.
Sokrates is een Griek.

Sokrates veracht zichzelf niet.

Opdracht 6.34 *Geef een regel voor de tableau-behandeling van vrije variabelen.*

We hebben tot nu toe verzuimd om regels te geven voor de behandeling van identiteiten. Hier volgen ze:

- =-rechts:

  Een tableau-tak waarin $t = t$ rechts voorkomt (waarbij t een of andere term is) moet zonder meer sluiten: $t = t$ is immers altijd waar.

- =-links:

  Als $d_i = d_j$ links in een subtableau voorkomt, vervang dan *overal in dat subtableau* (dus tot helemaal bovenaan toe), *zowel links als rechts*, de grootste van de twee indices door de kleinste. Dit zorgt ervoor dat de identiteit die waar moet zijn wordt afgedwongen.

Hier volgen een paar opdrachten die betrekking hebben op redeneringen waarin identiteit een rol speelt.

Opdracht 6.35 *Geef van beide nu volgende redeneringen parafrases in het Nederlands, en ga intuïtief na of ze geldig zijn:*

1. $\forall x \forall y((Ax \wedge Ay) \rightarrow x = y) \models \exists x(Ax \wedge Bx) \wedge \exists y(Ay \wedge \neg By)$
2. $\forall x(\neg Rxx \rightarrow \neg\exists y(\neg x = y \wedge Rxy)), \exists x \exists y(Rxy \wedge \neg x = y) \models \exists x Rxx.$

Opdracht 6.36 *Pas de tableau-regels voor de behandeling van predikatenlogische identiteits-uitspraken toe bij het testen van de geldigheid van de redeneringen uit de vorige opdracht.*

In de aanhef bij deze paragraaf hadden we gezegd dat de semantische tableau-methode niet in alle gevallen uitsluitsel oplevert over de vraag of een predikatenlogische redenering geldig is. Technisch gezegd: het is geen *beslissingsmethode*. Uit de voorbeelden die we tot nu toe behandeld hebben blijkt dit overigens niet: in alle gevallen waren we in staat om na te gaan of de redenering uit het voorbeeld geldig was of niet. Kennelijk konden zich drie mogelijkheden voordoen:

- sluiting van het tableau,
- geen sluiting, maar er zijn geen regels meer van toepassing,

- het tableau raakt in een oneindige lus.

We zetten deze drie mogelijkheden nog eens op een rijtje:

1. Sluiting geeft aan dat het systematisch zoeken naar een tegenvoorbeeld is mislukt. We weten nu dat de redenering geldig is. Sluiting van een tableau treedt altijd op na een eindig aantal regel-toepassingen.

2. Impasse: geen sluiting, en er kunnen geen regels meer worden toegepast. In dit geval valt er een eindig tegenvoorbeeld af te lezen uit een open tak in het tableau. Overigens is het wel steeds opletten geblazen om te kijken of we in een *echte* impasse zitten. Soms is een impasse slechts schijnbaar omdat we een regeltoepassing vergeten zijn.

3. Een oneindige lus: we kunnen regels blijven toepassen, maar daarbij blijft een en hetzelfde patroon zich herhalen. Als deze situatie zich voordoet kunnen we een oneindig tegenvoorbeeld aflezen.

Vergelijk dit met de propositielogica: daar hadden we alleen de gevallen 1. en 2. Daar konden we het volgende zeggen: de tableaumethode is mechanisch, dus we maken er een computerprogramma van, en we laten de computer draaien tot het tableau af is. Na afloop van het programma kunnen we dan zien of we in situatie 1. of 2. zitten. In het eerste geval was de redenering geldig, in het tweede geval was zij ongeldig. Kortom, in de propositielogica levert de tableau-methode een beslissingsmethode voor geldigheid.

Bij de predikatenlogica zit er hier een addertje onder het gras: omdat ook geval 3. zich kan voordoen heb je kans dat de computer niet stopt. Zolang de computer niet gestopt is weet je niet of de redenering geldig is of niet. Het programma onderbreken om te kijken of het in een lus zit helpt ook niet. Stel dat je het programma na 100 uur draaien onderbreekt, en het blijkt niet in een lus te zitten. We weten dan nog niets over de geldigheid van de redenering; we weten immers niet of het programma niet later, bij voorbeeld na 100 uur en drie minuten draaien, toch nog in een lus zou zijn terechtgekomen, of zou zijn gestopt in geval 1. of 2. Kortom, we weten het volgende, niet meer en niet minder:

> Als een predikatenlogische redenering geldig is, dan blijkt dat in een eindig aantal stappen, want dan zal een computerprogramma dat een tableau maakt voor die redenering na het verstrijken van een eindige hoeveelheid tijd stoppen met een gesloten tableau.

Waarom kunnen we, als we dit weten, dan niet in een eindig aantal stappen nagaan *of* een predikatenlogische redenering geldig is? Het probleem zit

hem hier in de omstandigheid dat we, hoewel we weten dat de geldigheid van een geldige redenering in eindig veel stappen aan het licht zal komen, niet van tevoren kunnen zeggen *hoeveel* stappen dat zijn. Zolang de computer niet stopt weten we niets.

6.7 Axiomatiek

Eerder hebben we gezien hoe we redeneringen in de predikatenlogica *semantisch* kunnen benaderen: we hebben het begrip *logisch geldig* uitgelegd in termen van *waarheid* in predikatenlogische modellen. Net als in de propositielogica bestaat er daarnaast ook een syntactisch-axiomatische kijk, waarbij redeneringen te lijf worden gegaan met behulp van de begrippen *afleidbaarheid* en *bewijsbaarheid*.

Het recept is weer: wijs eerst een aantal bona fide predikatenlogische formules aan als *axioma's*, geef vervolgens een of meer *redeneerregels*, en spreek af: alles wat in een eindig aantal stappen met behulp van de redeneerregels uit de axioma's kan worden afgeleid is een *stelling* van de predikatenlogica.

We definiëren dus weer een axioma-verzameling. Eerst nemen we de axioma-schema's van de propositielogica over:

Axioma 6.1 $\varphi \to (\psi \to \varphi)$.

Axioma 6.2 $(\varphi \to (\psi \to \chi) \to ((\varphi \to \psi) \to (\varphi \to \chi))$.

Axioma 6.3 $(\neg\varphi \to \neg\psi) \to (\psi \to \varphi)$.

Er is echter meer, want ook de kwantoren doen nu mee. We geven eerst een voorlopige formulering van een vierde axioma-schema:

Axioma 6.4 (voorlopige versie:)
$\forall v\varphi \to [t/v]\varphi$ *voor elke individuele variabele* v *en elke term* t.

Dit schema drukt uit dat een universele bewering elk van zijn 'toepassingen op een of ander bijzonder geval' impliceert. Dus: $\forall xPx \to Pa$ is een axioma van deze vorm. Er is echter een moeilijkheid verbonden aan de voorlopige versie van dit laatste axioma-schema.

Opdracht 6.37 *Waarom kan de formule*

$$\forall y\neg\forall x\ x = y \to [x/y]\neg\forall x\ x = y,$$

die voldoet aan de voorlopige versie van axioma-schema 4, geen axioma zijn? Met andere woorden: wat gaat er mis wanneer we in dit geval x *voor* y *substitueren in de deelformule* $\neg\forall x\ x = y$*?*

Uit het probleem in opdracht 6.37 blijkt dat de voorlopige versie van 4. te ruim is geformuleerd. We moeten een beperking opleggen aan het soort termen t dat voor v mag worden gesubstitueerd in φ. In wat nu volgt gaan we voor het gemak uit van een predikatenlogische taal die geen functiesymbolen bevat. Met *term* bedoelen we dus: constante of variabele. Hoe de functiesymbolen zouden moeten worden verdisconteerd gelieve u zelf te bedenken. We voeren het volgende begrip in teneinde de beperking die we aan 4. willen opleggen te formuleren.

Definitie 6.12 *De term t is* **vrij substitueerbaar** *voor de variabele v in de formule φ wanneer t ofwel een constante is, ofwel een variabele die vrij voorkomt in $[t/v]\varphi$ op elke plaats waar v vrij voorkwam in φ.*

Gauw een paar voorbeelden. a is vrij substitueerbaar voor y in $\forall xRxy$, want a is een constante. z is vrij substitueerbaar voor y in $\forall xRxy$, want $[z/y]\forall xRxy$ is gelijk aan $\forall xRxz$, en hierin is z vrij. x is *niet* vrij substitueerbaar voor y in $\forall xRxy$, want $[x/y]\forall xRxy$ is gelijk aan $\forall xRxx$, en hierin is x *niet* vrij op de plaats waar y eerst *wel* vrij was: de variabele wordt als het ware 'ingevangen' door de universele kwantor. z is *niet* vrij substitueerbaar voor y in $\forall z(Pz \rightarrow \exists xRxy)$; u gelieve zelf te bedenken waarom. Nog wat jargon: in plaats van *vrij substitueerbaar voor* zegt men ook wel *vrij voor* of *substitueerbaar voor*.

Opdracht 6.38 *Ga na of x vrij substitueerbaar is voor y in de volgende formules:*

1. $Py \wedge \exists xRxy$
2. $Px \wedge \exists zRzy$
3. $\forall y(Pz \rightarrow \exists xRxy)$
4. $\forall x(Py \rightarrow \exists yRxy)$
5. $\exists x(Px \wedge \exists yRxy)$
6. $Py \wedge \forall x\exists yRxy$.

Nu we de beschikking hebben over het begrip *vrij substitueerbaar zijn voor* kunnen we de correcte versie van axioma-schema 4. formuleren:

Axioma 6.4 *$\forall v\varphi \rightarrow [t/v]\varphi$ voor elke individuele variabele v en elke term t die vrij substitueerbaar is voor v in φ.*

Het nu volgende axioma-schema 5. is een beetje flauw. Het is nodig omdat we nu eenmaal hebben besloten om loze kwantificatie toe te staan.

Axioma 6.5 $\varphi \to \forall v\varphi$ *mits* v *niet vrij voorkomt in* φ.

Tenslotte hebben we:

Axioma 6.6 $\forall v(\varphi \to \psi) \to (\forall v\varphi \to \forall v\psi)$.

Dat waren de axioma-schema's. De verhouding van de echte axioma's tot de schema's is hier iets ingewikkelder dan in het geval van de propositielogica. Om de verzameling van predikatenlogische axioma's te kunnen definiëren moeten we eerst definiëren wat een generalizering van een formule is:

Definitie 6.13 *Een* **generalizering** *van een formule* φ *is een formule die ontstaat door 0 of meer universele kwantoren (met bijbehorende variabelen) te schrijven voor* φ.

Dus: Px, $\forall xPx$, $\forall x\forall zPx$,... zijn generalizeringen van Px. De verzameling predikatenlogische axioma's is nu als volgt gedefinieerd:

Definitie 6.14 *Een* **predikatenlogisch axioma** *is een generalizering van een formule volgens een van de schema's 1. tot en met 6.*

Voorbeelden van axioma's zijn:

- $\forall x(Px \to (Qx \to Px))$ [een generalizering van 1];
- $\forall xPx \to Pa$ [een generalizering van 4];
- $\forall x(\forall yPy \to Px)$ [een generalizering van 4];
- $\forall x(Px \to \forall yPx)$ [een generalizering van 5];
- $\forall x(Px \to Qx) \to (\forall xPx \to \forall xQx)$ [een generalizering van 6];
- $\forall z\forall y(\forall x(Px \to Qx) \to (\forall xPx \to \forall xQx))$ [een generalizering van 6].

Wanneer we een axiomaverzameling willen definiëren voor een predikatenlogische taal waarin ook het identiteitsteken wordt gebruikt hebben we nog twee extra axioma's nodig:

Axioma 6.7 $v = v$ *voor elke variabele* v.

Axioma 6.8 $(v_1 = w_1 \to (\ldots(v_n = w_n \to (Av_1 \ldots v_n \to Aw_1 \ldots w_n))\ldots))$
voor elke n*-plaatsige predikaatletter* A
en voor alle variabelen $v_1, \ldots, v_n$, $w_1, \ldots, w_n$.

Axioma-schema 8. ziet er misschien een beetje ingewikkeld uit, maar het idee is gewoon: in atomaire formules moet je variabelen die naar hetzelfde ding verwijzen voor elkaar kunnen substitueren. Een andere formulering van 8. is deze:

Axioma 6.8 (herformulering:)
$(v_1 = w_1 \wedge \ldots \wedge v_n = w_n \wedge Av_1 \ldots v_n) \rightarrow Aw_1 \ldots w_n$.

De 'implicatie-versie' die wij hebben gekozen verdient echter de voorkeur omdat de afleidingsregel Modus Ponens er mooier bij aansluit.

Wanneer we een calculus willen geven voor een taal die behalve het identiteitsteken ook functiesymbolen bevat hebben we nog een negende axioma nodig, om te garanderen dat we in termen die van de vorm $gt_1 \ldots t_n$ zijn, termen voor $t_1, \ldots, t_n$ mogen substitueren mits ze naar hetzelfde individu verwijzen. We laten dat axioma hier nu maar voor het gemak achterwege.

Hier volgen enkele voorbeelden van generalizeringen van 7. en 8.

- $\forall x\ x = x$
- $\forall x \forall y(x = y \rightarrow (Px \rightarrow Py))$
- $(x = y \rightarrow (Px \rightarrow Py))$
- $\forall x \forall y \forall z \forall u(x = y \rightarrow (z = u \rightarrow (Rxz \rightarrow Ryu)))$.

De definitie van de verzameling axioma's van een predikatenlogische taal met identiteit wordt nu:

Definitie 6.15 *Een* **predikatenlogisch axioma** *voor een predikatenlogische taal $\mathcal{T}$ met identiteit is een generalizering van een formule volgens een van de axioma-schema's 1. tot en met 8.*

De afleidingsregel is weer *Modus Ponens*: concludeer uit φ en $\varphi \rightarrow \psi$ tot ψ.

Tenslotte voeren we $\wedge$, $\vee$, $\leftrightarrow$ and $\exists$ als afkortingen in. Naast de definities uit § 5.8 hebben we nog nodig:

- $\exists v\varphi$ is een afkorting voor $\neg\forall v\neg\varphi$.

Hiermee hebben we een deductief systeem (of: een axiomatiek, of: een calculus) voor de predikatenlogica gegeven. Er bestaan vele andere deductieve systemen voor de predikatenlogica die *equivalent* zijn in de zin dat ze dezelfde verzameling formules als stellingen opleveren. De definities van *afleiding* (of: *bewijs*), en *stelling* zijn als bij de propositielogica.

Net als bij de propositie-logische calculus die we in § 5.7 gepresenteerd hebben, moeten we weer onderscheid maken tussen bewijzen *in* de calculus en bewijzen *over* de calculus. Hier is een voorbeeld van een bewijs *in* de calculus.

Stelling 6.1 *Voor elke constante a is de formule $a = a$ een stelling.*

Bewijs:

1	$\forall x\ x = x$	[axioma volgens schema 7]
2	$\forall x\ x = x \rightarrow a = a$	[axioma volgens schema 4]
3	$a = a$	[MP uit 1 en 2].

■

De notatie voor "$a = a$ is een stelling" is weer: $\vdash a = a$. We roepen de notatie in herinnering voor "φ is afleidbaar uit formuleverzameling Γ":

$$\Gamma \vdash \varphi.$$

Voorbeeld: laat Γ de verzameling $\{\forall xRxx\}$ zijn. Dan kunnen we een afleiding geven van Raa:

Stelling 6.2 $\forall xRxx \vdash Raa.$

Bewijs:

1	$\forall xRxx$	[volgens aanname]
2	$\forall xRxx \rightarrow Raa$	[axioma volgens schema 4]
3	Raa	[MP uit 1 en 2].

■

Een andere notatie die ook wel wordt gebruikt voor het resultaat uit de stelling is:

$$\forall xRxx \vdash Raa.$$

Opdracht 6.39 *Laat zien:*

$$\forall x(Ax \rightarrow Bx), Aa \vdash Ba.$$

Opdracht 6.40 *Laat zien:*

$$\forall x(\neg Ax \rightarrow \neg Bx), \forall x(Ax \rightarrow Cx), Ba \vdash Ca.$$

Hier is nog een voorbeeld van een bewijs van een stelling van de calculus.

Stelling 6.3 $\vdash x = y \rightarrow y = x.$

Bewijs:

1	$x = y \to (x = x \to (x = x \to y = x))$	[ax 8]
	NB $=$ is een tweeplaatsig predikaat	
2	$(x = y \to (x = x \to (x = x \to y = x))) \to$ $((x = y \to x = x) \to$ $(x = y \to (x = x \to y = x)))$	[ax 2]
3	$(x = y \to x = x) \to (x = y \to (x = x \to y = x))$	[MP uit 1,2]
4	$x = x$	[ax 7]
5	$x = x \to (x = y \to x = x)$	[ax 1]
6	$x = y \to x = x$	[MP uit 4,5]
7	$x = y \to (x = x \to y = x)$	[MP uit 3,6]
8	$(x = y \to (x = x \to y = x)) \to$ $((x = y \to x = x) \to (x = y \to y = x))$	[ax 2]
9	$(x = y \to x = x) \to (x = y \to y = x)$	[MP uit 7,8]
10	$x = y \to y = x$	[MP uit 6,9].

■

U ziet het: een simpel principe als $x = y \to y = x$ vereist al het een en ander aan formule-manipulatie. Overigens zijn, afgezien van het gebruik van de axioma's 7 en 8, alle stappen in het bewijs puur propositielogische stappen; kwantormanipulatie speelt in het bewijs geen rol.

Het is niet de bedoeling dat u zich over het vinden van dit soort bewijzen het hoofd breekt. Wel is het nuttig dat u een idee heeft van wat een *bewijs in de predikaatlogische calculus* is. Vandaar de bovenstaande voorbeelden.

Weer is—net als in het propositielogische geval—het onderscheid tussen een bewijs *in* en een bewijs *over* de calculus van groot belang. Hier is een voorbeeld van een metastelling voor de predikatenlogische calculus:

Stelling 6.4 *De predikatenlogische calculus voldoet aan het principe van Universele Generalisatie: Als* $\vdash \varphi$ *dan ook* $\vdash \forall v\varphi$.

Bewijs: We gebruiken weer inductie naar de lengte van een bewijs in de calculus.

- *Basisstap*: het bewijs van φ bestaat uit het rijtje $\langle\varphi\rangle$ dat alleen de formule φ bevat. In dit geval moet φ een axioma zijn. Maar dan is ook $\forall v\varphi$ een axioma, want: $\forall v\varphi$ is een *generalisering* van φ. Daarmee is $\langle\forall v\varphi\rangle$ een bewijs voor $\forall v\varphi$. We hebben dus: $\vdash \forall v\varphi$, en het basisgeval is afgewerkt.

- *Inductiestap*: De inductiehypothese is: als φ een bewijs heeft van lengte n (of kleiner), dan geldt $\vdash \forall v\varphi$. Merk op dat de inductiehypothese *niets* zegt over de lengte van het bewijs van $\forall v\varphi$. We moeten laten zien dat nu uit $\vdash \varphi$ en "φ heeft een afleiding van lengte $n + 1$" ook

volgt: $\vdash \forall v\varphi$. Dat gaat zo: of φ is een axioma, en we redeneren als in het basisgeval en klaar, of φ volgt via Modus Ponens uit ψ en $\psi \to \varphi$, waarbij op ψ en $\psi \to \varphi$ de inductiehypothese van toepassing is (ga na waarom). We hebben dus: $\vdash \forall v\psi$ en $\vdash \forall v(\psi \to \varphi)$. Hieruit leiden we $\forall v\varphi$ als volgt af:

1	$\forall v(\psi \to \varphi) \to (\forall v\psi \to \forall v\varphi)$	[ax 6]
2	$\forall v(\psi \to \varphi)$	[gegeven]
3	$\forall v\psi \to \forall v\varphi$	[MP uit 1,2]
4	$\forall v\psi$	[gegeven]
5	$\forall v\varphi$	[MP uit 3,4].

We hebben laten zien dat $\vdash \forall v\varphi$. Hiermee is de inductiestap compleet, en dus is het bewijs van het principe van Universele Generalisatie rond.

■

Het principe van Universele Generalisatie staat ons nu toe om uit

$$\vdash x = y \to y = x$$

(stelling 6.3) het volgende af te leiden:

$$\vdash \forall x \forall y(x = y \to y = x).$$

Met andere woorden: we hebben in de predikatenlogische calculus bewezen dat de identiteits-relatie *symmetrisch* is.

Een tweede voorbeeld van een metastelling voor de predikatenlogische calculus is de deductiestelling. Alle principes die we hebben gebruikt in het bewijs van de deductiestelling voor de propositielogica (vergelijk § 5.7) gaan ook op voor de predikatenlogische calculus, dus het eerder gegeven bewijs geldt ook hier.

Metastellingen als het principe van universele generalisatie en de deductiestelling geven ons in feite extra redeneerregels in handen. Op het feit dat de deductiestelling buitengewoon nuttig is als extra redeneerregel hebben we in § 5.7 al gewezen. Voor de algemene variant,

$$\text{als } \varphi_1, \ldots, \varphi_n, \varphi_{n+1} \vdash \psi \text{ dan } \varphi_1, \ldots, \varphi_n \vdash (\varphi_{n+1} \to \psi),$$

geldt dit in nog sterkere mate (zie opdracht 5.42 aan het eind van § 5.7). Stel dat we willen laten zien:

$$\vdash (\varphi \to (\psi \to \chi)) \to (\psi \to (\varphi \to \chi)),$$

dan is het volgens de deductiestelling genoeg om te laten zien:

$$\varphi \to (\psi \to \chi) \vdash \psi \to (\varphi \to \chi).$$

Om dat te laten zien is het, weer volgens de deductiestelling (maar nu in de algemene variant), genoeg om te bewijzen:

$$\varphi \to (\psi \to \chi), \psi \vdash \varphi \to \chi$$

en daarvoor is een bewijs van

$$\varphi \to (\psi \to \chi), \psi, \varphi \vdash \chi$$

weer genoeg. Dat bewijs gaat als volgt:

1	$\varphi \to (\psi \to \chi)$	[gegeven]
2	φ	[gegeven]
3	$\psi \to \chi$	[MP uit 1,2]
4	ψ	[gegeven]
5	χ	[MP uit 3,4].

U ziet het: heel wat gemakkelijker dan direct een bewijs leveren voor de formule waar we mee begonnen.

We zijn geïnteresseerd in de verzameling van alle formules die afleidbaar zijn uit een bepaalde aannamen-verzameling T in de predikatenlogische calculus. Wat dit betekent is dat we door het kiezen van geschikte uitgangspunten een specifiek *soort* van predikatenlogische modellen kunnen beschrijven. Om gemakkelijk over dit soort zaken te kunnen praten voeren we nieuw jargon in.

Definitie 6.16 *Een* **predikatenlogische theorie** T *is een verzameling van predikatenlogische formules.*

We noemen de formules uit T de *niet-logische axioma's* van de theorie. In feite leggen de niet-logische axioma's samen met de predikatenlogische axioma's (waarvoor we het recept hierboven hebben gegeven) en de afleidingsregel Modus Ponens de theorie vast.

Definitie 6.17 *De* **deductieve afsluiting** *van een theorie T, notatie $\overline{T}$, is de verzameling $\{\varphi \mid T \vdash \varphi\}$.*

De deductieve afsluiting van T is de verzameling van alle formules die met behulp van Modus Ponens kunnen worden afgeleid uit formules in T en logische axioma's.

Niet alle formuleverzamelingen T zijn even interessant. We zijn met name geïnteresseerd in theorieën T die *consistent* zijn (zie definitie 5.15 in § 5.8).

Opdracht 6.41 *Laat zien dat als T consistent is, dan $\overline{T}$ ook.*

Opdracht 6.42 *Laat zien dat als $\overline{T}$ consistent is, dan T ook.*

Een heel simpel voorbeeld van een predikatenlogische theorie is de theorie T die alleen de ene formule $\exists x\ x = x$ bevat, dus: $T = \{\exists x\ x = x\}$. Deze theorie is waar in alle predikaatlogische modellen met een niet-leeg domein: de formule $\exists x\ x = x$ is waar precies wanneer er minstens één ding bestaat. Dit sluit alleen het model met het lege domein uit. Merk op dat het model met het lege domein *niet* wordt uitgesloten door de definitie van 'model voor een predikatenlogische taal'.

Hier is een wat serieuzer voorbeeld van een predikatenlogische theorie, de zogenaamde theorie van de *partiële ordes*:

$$\{\forall x Rxx,\\ \forall x\forall y((Rxy \wedge Ryx) \rightarrow x = y),\\ \forall x\forall y\forall z(Rxy \rightarrow (Ryz \rightarrow Rxz))\}.$$

De drie formules die de theorie uitmaken drukken respectievelijk uit dat de relatie die door R wordt benoemd reflexief, antisymmetrisch en transitief is. Elk model voor deze theorie moet een partieel geordende verzameling zijn, dat wil zeggen een verzameling waarop een partiële orde is gedefinieerd (vergelijk § 2.7).

Een ander voorbeeld van een predikatenlogische theorie is de zogenaamde theorie van de *dichte onbegrensde lineaire ordes*. We laten de niet-logische axioma's die de theorie uitmaken een voor een de revue passeren:

1. $\forall x\forall y\forall z(Rxy \rightarrow (Ryz \rightarrow Rxz))$.

   Deze formule drukt uit dat de relatie die door R wordt benoemd *transitief* is.

2. $\forall x\forall y(Rxy \rightarrow \neg Ryx)$.

   Deze formule drukt uit dat de relatie die door R wordt benoemd *asymmetrisch* is.

3. $\forall x\forall y(Rxy \vee Ryx \vee x = y)$.

   Deze formule drukt uit dat de relatie die door R wordt benoemd de eigenschap van *samenhang* heeft (of: samenhangend is).

4. $\forall x\forall y(Rxy \rightarrow \exists z(Rxz \wedge Rzy))$.

   Deze formule drukt uit dat de relatie die door R wordt benoemd de eigenschap van *dichtheid* heeft (of: dicht is).

5. $\forall x \exists y Rxy$.

   Deze formule drukt uit dat de relatie die door R wordt benoemd *voortzetting naar rechts* kent.

6. $\forall x \exists y Ryx$.

   Deze formule drukt uit dat de relatie die door R wordt benoemd *voortzetting naar links* kent.

We kunnen nu gaan kijken naar *modellen* voor deze theorie. Een voorbeeld van zo'n model is de verzameling van alle *tijdstippen*, met R geïnterpreteerd als de relatie *eerder dan*. De formules van de theorie worden nu uitspraken over de tijd. Die uitspraken zijn als zodanig voor discussie vatbaar, maar dat is een andere kwestie. 1. wordt nu: als tijdstip 1 eerder valt dan tijdstip 2, en tijdstip 2 valt eerder dan tijdstip 3, dan valt tijdstip 1 eerder dan tijdstip 3. Net zo voor de andere formules. Het model voor de tijd dat hier beschreven wordt ziet er dus zo uit:

$\leftarrow$*vroeger* *later* $\rightarrow$

Vergelijk § 5.10, waar een dergelijke tijdsas figureert in de modellen voor de propositionele tijdslogica. Merk op dat in dit model de tijd zich onbegrensd uitstrekt naar verleden en toekomst, en dat er zich tussen elk tweetal tijdstippen, hoe dicht ze ook bij elkaar liggen, een derde bevindt. Op de filosofische implicaties van deze visie op tijd gaan we hier niet in.

Een ander model voor de theorie van de dichte onbegrensde lineaire ordes is het domein van alle positieve en negatieve breuken en het getal 0, met R geïnterpreteerd als de relatie *kleiner dan*. Dit model wordt vaak aangeduid als **Q**. U gelieve zelf na te gaan dat formules 1. tot en met 6. waar zijn in **Q**.

Andere voorbeelden van predikatenlogische theorieën zijn er te over. In de meeste gevallen hebben ze, in tegenstelling tot de bovengenoemde voorbeelden, niet een eindig maar een *oneindig* stel niet-logische axioma's. Het feit dat een theorie een oneindige verzameling niet-logische axioma's heeft is niet zo bezwaarlijk als op het eerste gezicht lijkt: we zijn met name geïnteresseerd in theorieën waarvan de axioma's *recursief* gedefinieerd kunnen worden. Als dat kan is de axioma-verzameling mogelijkerwijs wel oneindig, maar toch met eindige middelen weer te geven (net als de verzameling predikatenlogische axioma's). We zeggen dan dat zo'n oneindige theorie *oneindig recursief is geaxiomatiseerd*. Een beroemd voorbeeld is de theorie van de rekenkunde zoals geaxiomatiseerd in de axioma's van Peano (de zogenaamde Peano-rekenkunde). Deze theorie wordt vaak afgekort als PA.

Ook de verzamelingenleer kan in axiomatische vorm worden gepresenteerd. Dit axiomatiseren van de verzamelingenleer is ter hand genomen nadat Bertrand Russell een tegenspraak had afgeleid uit Georg Cantor's oorspronkelijke verzamelingenleer (de zogenaamde naïeve verzamelingenleer; vergelijk de titel van hoofdstuk 2 van dit boek). Het separatie-axioma, vermeld in § 3.5, ziet er nu bij voorbeeld als volgt uit:

$$\forall x \exists y \forall z (Rzy \leftrightarrow (Rzx \wedge \varphi(z))).$$

In deze formule is R een tweeplaatsige predikaatletter die wordt geïnterpreteerd als *is element van* (dat wil zeggen: de $\in$ relatie); $\varphi(z)$ is een formule waarin alleen de variabele z vrij voorkomt. In het axioma wordt $\varphi(z)$ gebruikt om de verzamelingen die voldoen aan φ te karakteriseren. Door een verstandige keuze van de overige axioma's kan nu worden bewerkstelligd dat de Russell paradox en verwante problemen niet meer optreden. Verschillende systemen zijn voorgesteld; het bekendste axiomastelsel is de zogenaamde Zermelo-Fraenkel verzamelingenleer (ingewijden korten dit af als ZF). Zie voor meer informatie het al eerder genoemde leerboek [Van Dalen e.a. 1975].

6.8 Correctheid, volledigheid, onbeslisbaarheid

Net als bij de propositielogica moeten we de brandende vraag beantwoorden naar de verhouding tussen de begrippen *afleidbaar* en *logisch gevolg*. Dus ten eerste: is de predikatenlogische calculus correct? Met andere woorden, geldt het volgende: als $\Gamma \vdash \varphi$, dan $\Gamma \models \varphi$ (waarbij Γ een willekeurige formuleverzameling is, en φ een willekeurige formule)? Om dit te laten zien moet eerst het feit uit de volgende opdracht worden aangetoond.

Opdracht 6.43 *Laat zien: als $b(v) = b'(v)$ voor elke vrije variable v in φ, dan $V_b(\varphi) = V_{b'}(\varphi)$.*

Het bewijs van de correctheid van de predikatenlogische calculus heeft nogal wat voeten in de aarde.[1]

Stelling 6.5 (Correctheid van de predikatenlogica)
Voor elke predikatenlogische formuleverzameling Γ en elke predikatenlogische formule φ: als $\Gamma \vdash \varphi$, dan $\Gamma \models \varphi$.

Bewijs: We plegen inductie naar de lengte van de afleiding van φ uit Γ, en maken daarbij gebruik van het resultaat uit opdracht 6.43. Één-stapsafleidingen zijn er in twee soorten:

[1] Als u wilt mag u het nu volgende bewijs bij eerste lezing overslaan.

- $\varphi \in \Gamma$. In dat geval geldt voor elk model $\mathcal{M} = \langle D, I \rangle$ met bijbehorende valuatie V en iedere bedeling b dat $V_b(\varphi) = 1$ als voor elke $\gamma \in \Gamma$ geldt $V_b(\gamma) = 1$. Dus is in dit geval $\Gamma \models \varphi$.

- φ is een predikatenlogisch axioma. Zo'n axioma is een generalisering van één van de schema's 1–6. De geldigheid hiervan kunnen we wederom bewijzen met inductie (naar het aantal universele kwantoren dat bij generalisering voorop geplaatst wordt):

  - De schema's 1–3 gelden nog steeds; weliswaar mogen de formules nu variabelen, kwantoren, predikaten, enzovoort bevatten, de geldigheid van deze schema's berust uitsluitend op de predikatenlogische valuatie van $\rightarrow$ en $\neg$ en die is in essentie hetzelfde als in de propositielogica.
  - De geldigheid van axiomaschema 4 is zeker niet triviaal. Stel voor willekeurige valuatie V en bedeling b met domein D dat $V_b(\forall v\varphi) = 1$, dan is voor ieder object $d \in D$: $V_{b(v|d)}(\varphi) = 1$. We moeten nu laten zien dat $V_b([t/v]\varphi) = 1$ mits t substitueerbaar is voor v in φ. Merk op dat als $d = W_b(t)$ dan

    $$V_b([t/v]\varphi) = V_{b(v|d)}(\varphi) \text{ mits } t \text{ substitueerbaar voor } v \text{ in } \varphi.$$

    Deelbewijs: opnieuw met inductie, ditmaal naar de opbouw van φ (het totale bewijs heeft dus de vorm van een 3-traps-inductie). Zowel de basisstap(pen) voor atomaire formules als de inductiestappen voor de connectieven zijn eenvoudig. De kwantorstap is echter wel complex:
    Neem voor φ de formule $\forall u\varphi$. We onderscheiden twee gevallen:

    $v = u$**.** Dan $V_b([t/v]\forall u\varphi) = V_b(\forall u\varphi) = V_{b(u|d)}(\forall u\varphi) = V_{b(v|d)}(\forall u\varphi)$ (de voorlaatste stap gebruikt het resultaat uit opdracht 6.43);

    $v \neq u$**.** Dan $V_b([t/v]\forall u\varphi) = V_b(\forall u[t/v]\varphi) = 1$ desda voor elke $e \in D$: $V_{b(u|e)}([t/v]\varphi) = 1$. De inductiehypothese (toegepast op bedeling $b(u|e)$) voor dit deelbewijs levert dat voor elke $e \in D$ $V_{b'}(\varphi) = 1$, waarbij $b' = b(u|e)(v|d)$ en substitueerbaar-zijn met zich meebrengt dat $t \neq u$, en derhalve $W_{b(u|e)}(t) = W_b(t) = d$. Omdat $v \neq u$ geldt echter ook dat $b' = b(v|d)(u|e)$ en daarom volgt de equivalentie met $V_{b(v|d)}(\forall u\varphi) = 1$, hetgeen te bewijzen was.
  - Axiomaschema 5 laat zich eenvoudig verifiëren: stel maar voor willekeurige D,V en b: $V_b(\varphi) = 1$ en v niet vrij in φ. Voor willekeurige $d \in D$ geldt dan volgens het resultaat uit opdracht 6.43

(immers b en $b(v|d)$ zijn dan identiek voor de vrije variabelen in φ): $V_{b(v|d)}(\varphi) = 1$, ergo $V_b(\forall v\varphi) = 1$.

– De geldigheid van schema 6 bewijzen we indirect: stel dat 6 niet geldt. Dan is er een dus een model met valuatie V, domein D en bedeling b zodat (a) $V_b(\forall v(\varphi \to \psi) = 1$ en (b) $V_b(\forall v\varphi \to \forall v\psi) = 0$. Uit (b) volgt dat (c) $V_b(\forall v\varphi) = 1$ en (d) $V_b(\forall v\psi) = 0$. (d) impliceert dat er een $e \in D$ zou bestaan waarvoor $V_{b(v|e)}(\psi) = 0$. Uit (a) en (c) volgen echter respectievelijk dat $V_{b(v|e)}(\varphi \to \psi) = 1$ en $V_{b(v|e)}(\varphi) = 1$, en samen levert dit $V_{b(v|e)}(\psi) = 1$, wat in tegenspraak is met het gevolg van (d).

– We bewijzen de semantische geldigheid van het principe van Universele Generalisatie: laat $\models \varphi$. Dan geldt voor elke valuatie V en bedeling b dat $V_b(\varphi) = 1$ en dus ook elke V en b: $V_{b(v|d)}(\varphi) = 1$ voor elk object d uit het domein. Dus geldt per definitie voor elke V en B: $V_b(\forall v\varphi) = 1$, kortom $\models \forall v\varphi$.

De geldigheid van Universele Generalisatie en de schema's 1–6 heeft de geldigheid van alle predikatenlogische axioma's tot gevolg.

- De enige afleidingsregel van het formele systeem is Modus Ponens. Stel nu dat $\Gamma \models \varphi$ en $\Gamma \models \varphi \to \psi$. Kies een valuatie V en een bedeling b zodat $V_b(\gamma) = 1$ voor elke $\gamma \in \Gamma$. Uit de veronderstellingen volgt nu $V_b(\varphi) = 1$ en $V_b(\varphi \to \psi) = 1$, en dus $V_b(\psi) = 1$. Maar dan $\Gamma \models \psi$. ∎

Dat was het bewijs van de predikatenlogische correctheid. Ten tweede: is de predikatenlogische calculus volledig, in de zin dat het volgende geldt: als $\Gamma \models \varphi$, dan $\Gamma \vdash \varphi$ (weer met Γ een willekeurige formuleverzameling en φ een willekeurige formule) ? De logicus Kurt Gödel bewees in 1930 dat dit inderdaad het geval is. Deze stelling heet: de volledigheidsstelling van Gödel.

Wellicht vraagt de lezer zich af hoe Gödel in 1930 een eigenschap van een systeem kon bewijzen die Tarski eerst in 1933 voorstelde. De verklaring van dit schijnbare anachronisme is dat de ideeën over modeltheoretische semantiek al geruime tijd circuleerden in werk van bij voorbeeld Hilbert, Bernays en Ackermann. Tarski heeft aan die gedachten een precieze en algemene vorm gegeven, maar voor Gödel waren de eerdere voorstellen al voldoende. Zijn bewijs maakte geen gebruik van de methode van Henkin, maar wel van de in § 6.3 genoemde stelling van Löwenheim en Skolem. Wij laten een bewijs van de volledigheid van de predikatenlogica hier overigens achterwege.

Een alternatieve formulering voor de volledigheidsstelling van Gödel is de volgende: elke (syntactisch) consistente verzameling formules is vervulbaar (in enig model). We laten zien dat dit volgt uit de gebruikelijke formulering van de volledigheidsstelling. Consistentie van een formuleverzameling Γ komt—zoals we in stelling 5.8 bewezen hebben—neer op:

Er is een formule φ zo dat $\Gamma \nvdash \varphi$.

Met behulp van de volledigheidsstelling volgt hieruit, door toepassen van contrapositie:

$\Gamma \not\models \varphi$.

Wat wil dit zeggen? Niets anders dan: er is minstens één model $\mathcal{M}$ en een bedeling b waarin elke formule uit Γ vervulbaar is, maar φ niet. Met andere woorden: Γ is vervulbaar.

Deze herformulering van Gödels volledigheidsstelling maakt duidelijk hoe de syntactische definitie van *consistentie* uit 6.7 kan leiden tot de semantische karakterisering van *inconsistente verzameling formules* als: verzameling formules die niet vervulbaar is.

Een ander punt is de *beslisbaarheid* van de predikatenlogica. Een theorie T is *beslisbaar* wanneer er een mechanische procedure bestaat om, bij een gegeven formule φ van de taal, uit te maken of $T \vdash \varphi$ dan wel $T \nvdash \varphi$. De vraag naar de beslisbaarheid van de predikatenlogica is een bijzonder geval: het geval waar T leeg is. Dus: is er een mechanische procedure om bij een gegeven formule φ van de taal uit te maken of φ een stelling is? In 6.6 hebben we gezien dat de methode van de semantische tableaus voor de predikatenlogica geen beslissingsprocedure is. Dat zat hem in het feit dat de tableau-regels voor de kwantoren herhaald moeten worden toegepast, zodat er geen garantie meer is dat een tableau in eindig veel stappen kan worden voltooid. We kunnen nu direct het volgende inzien.

Stelling 6.6 *De verzameling van alle* **kwantorvrije stellingen** *van de predikatenlogica is beslisbaar.*

Bewijs: Wanneer φ een kwantorvrije predikatenlogische formule is, dan komen er aan het tableau alleen propositielogische tableauregels te pas, regels die variabelen en constanten vervangen door 'standaardnamen' uit het rijtje $d_1, d_2, \ldots$ plus eventueel de regels voor identititeit. De regel voor $=$-links moet herhaald worden toegepast, maar het aantal malen is eindig, want het aantal voorkomens van standaard-namen uit $d_1, d_2, \ldots$ in het tableau is beperkt tot namen die zijn ingevoerd voor variabelen en constanten in de oorspronkelijke formule φ, en dat zijn er eindig veel. Hieruit zien we dat het tableau voor φ in eindig veel stappen is voltooid. Vanwege het feit dat

de predikatenlogica volledig is mogen we deze beslissingsprocedure voor $\models$ beschouwen als een beslissingsprocedure voor $\vdash$. ∎

Aantonen dat de predikatenlogica als geheel niet beslisbaar is heeft veel meer voeten in de aarde: uit het feit dat de tableau-methode geen beslissingsmethode is volgt uiteraard nog niet dat er niet een andere testmethode zou kunnen zijn die *wel* een beslissingsmethode is.

Hoewel de predikatenlogica als geheel niet beslisbaar is het heel wel mogelijk om in de predikatenlogica beslisbare *theorieën* te formuleren: de extra niet-logische axioma's van de theorie perken de klasse van modellen dusdanig in dat de notie $\models$ (en dus $\vdash$) wat hanteerbaarder wordt, zou je kunnen zeggen. Een voorbeeld van een beslisbare predikatenlogische theorie is de theorie van de dichte onbegrensde lineaire ordes (vergelijk § 6.7). Bij de theorie van de Peano-rekenkunde (PA) en de Zermelo-Fraenkel axiomatisering van de verzamelingenleer (ZF) ligt het anders. Deze theorieën zijn *niet* beslisbaar. Een **Pascal** programma dat uitmaakt of een willekeurige bewering φ afleidbaar is uit PA of uit ZF valt niet te schrijven.

De welgevormde formules van een predikatenlogische taal kunnen worden opgesomd. Het zijn immers eindige rijtjes tekens in een eindig alfabet, en zodra er een volgorde voor dat alfabet is vastgelegd kunnen alle formules van de taal in de in hoofdstuk 3 besproken telefoonboek-volgorde achter elkaar worden gezet. Dit levert een gigantisch maar aftelbaar telefoonboek op. Neem aan dat de formules genummerd zijn als 0, 1, 2, 3 enzovoort: met iedere formule correspondeert dus een natuurlijk getal. Het volgende is nu vrij eenvoudig in te zien.

Stelling 6.7 *De verzameling van de natuurlijke getallen die corresponderen met de stellingen van de predikatenlogica is opsombaar.*

Bewijs: We mogen de bewering uit de stelling wel parafraseren als: "Er bestaat een computerprogramma dat de verzameling *stellingen* van de predikatenlogica opsomt". Het gezochte programma moet dus elke formule die een stelling is, of het nummer van die formule in het formules-telefoonboek, vroeg of laat afdrukken.

Dat er zo'n programma bestaat blijkt als volgt. Een stelling van de predikatenlogica is de laatste formule in een *bewijs*, en een bewijs is een eindig rijtje formules dat aan zekere eisen voldoet. De verzameling van alle eindige rijtjes formules is weer *aftelbaar* (gebruik het telefoonboek dat alle formules opsomt om een nieuw telefoonboek te maken dat alle eindige *rijtjes* van formules opsomt). Ons computerprogramma werkt nu als volgt (commentaar staat tussen { }):

BEGIN

maak n gelijk aan 0;
(1) controleer of het formulerijtje dat positie n inneemt een bewijs is;
{deze controle kan mechanisch worden verricht:
er hoeft alleen maar te worden gecontroleerd of de formules
in het rijtje axioma's zijn, dan wel resultaten
van het toepassen van MP op twee voorafgaande formules}
ALS formulerijtje met nummer n een bewijs is DAN
druk de laatste formule in het rijtje af; {want dat is een stelling}
maak n gelijk aan $n + 1$;
GA NAAR (1)
EINDE.

Het is duidelijk dat alle stellingen zo vroeg of laat te voorschijn komen. ∎

Merk op dat het programma dat in de stelling beschreven wordt nooit stopt: het blijft tot in lengte van dagen formules uitbraken.

U ziet het: de opsombaarheid van de verzameling stellingen van de predikatenlogica hangt samen met de mogelijkheid om *mechanisch te controleren* of een gegeven rijtje formules een bewijs is voor een 'kandidaat-stelling'. Als we vragen of de verzameling stellingen van de predikatenlogica beslisbaar is vragen we (veel) meer: we vragen dan of er ook een mechanische methode is om bewijzen te *vinden*. Alonzo Church heeft in 1936 bewezen dat het antwoord op deze vraag *nee* is.

Stelling 6.8 (Stelling van Church) *Er bestaat geen recursieve procedure die de verzameling stellingen van de predikatenlogica herkent.*

Bewijs: Zie [Van Dalen 1983] of [Enderton 1972]. ∎

Met behulp van de these van Church volgt uit deze stelling: de predikatenlogica is *niet beslisbaar*. De stelling van Church leidt meteen tot de volgende stelling.

Stelling 6.9 *De verzameling van niet-stellingen van de predikatenlogica is* **niet** *opsombaar.*

Bewijs: Stel dat de verzameling van niet-stellingen van de predikatenlogica wel opsombaar zou zijn. Dan zouden we een computerprogramma hebben dat de verzameling formules opsomt die *geen* stellingen zijn van de predikatenlogica. Noem dit het niet-stellingen-programma. We hadden ook al een programma dat de stellingen van de predikatenlogica opsomde (zie stelling 6.7). Noem dit programma het stellingen-programma. Deze twee programma's kunnen nu worden gecombineerd tot een beslissingsprogramma dat voor willekeurige formules nagaat of het stellingen zijn of niet. In het beslissingsprogramma hieronder is φ een variabele die de formule bevat die we

onderzoeken. We maken gebruik van de notatie := voor het toekennen van een waarde aan een variabele; $n := 0$ staat dus voor "maak de waarde van n gelijk aan 0". *WelEenStelling* en *GeenStelling* zijn twee boolese variabelen, dat wil zeggen: variabelen die een waarheidswaarde bevatten.

```
BEGIN
   n := 0;
   WelEenStelling := onwaar;
   GeenStelling := onwaar;
   HERHAAL
        ALS φ gelijk is aan formule n in
        de uitvoer van het stellingen-programma DAN
              WelEenStelling := waar
        ANDERS ALS φ gelijk is aan formule n in
        de uitvoer van het niet-stellingen-programma DAN
              GeenStelling := waar;
        n := n + 1
   TOTDAT WelEenStelling of GeenStelling;
   ALS WelEenStelling DAN schrijf (φ, ' is een stelling.')
   ANDERS schrijf (φ, ' is geen stelling.')
EINDE.
```

Het bestaan van zo'n beslissingsprogramma is in strijd met de stelling van Church, dus de aanname dat de verzameling niet-stellingen van de predikatenlogica kan worden opgesomd moet worden verworpen. ∎

6.9 Volledige en onvolledige theorieën

Om nog iets naders te kunnen zeggen over de eisen waaraan een predikatenlogische theorie moet voldoen om beslisbaar te zijn voeren we nu een nieuw begrip in.

Definitie 6.18 *Een theorie T heet* **volledig** *wanneer voor elke zin φ van de taal geldt: $T \vdash \varphi$ of $T \vdash \neg\varphi$.*

Merk op dat deze definitie betrekking heeft op de *zinnen* (dat wil zeggen: de gesloten formules) van een theorie. Het is niet redelijk om van een theorie te verwachten dat ook voor open formules (zoals Rxy) geldt dat ofwel de formule zelf ofwel zijn negatie afleidbaar is. We hoeven niet bang te zijn dat er door deze beperking interessante formules uit de boot vallen, want (zoals u gemakkelijk kunt nagaan): als een of andere open formule φ afleidbaar is uit een theorie T, dan is zijn *universele afsluiting* (dat wil zeggen:

het resultaat van universeel kwantificeren over alle variabelen die vrij in φ voorkomen) ook afleidbaar uit T.

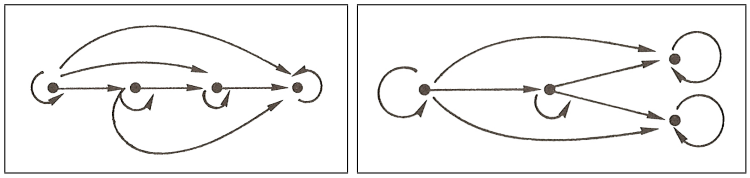
Was de eis van consistentie een soort minimum-eis die aan theorieën kan worden gesteld, *volledigheid* is een maximum-eis, en er is slechts een select gezelschap van theorieën dat eraan voldoet.

Opdracht 6.44 *Bewijs dat voor wat betreft gesloten formules:*

T is consistent en volledig $\iff$ $\overline{T}$ is maximaal consistent.

Let op: het begrip *volledigheid* dat hier wordt geïntroduceerd is een *ander* begrip dan de notie die we in de paragrafen 5.8 en 6.8 hebben gebruikt. Het is ook verschillend van het begrip *functionele volledigheid* uit § 5.9. De predikatenlogica is *volledig* in de zin dat predikatenlogische geldigheid predikatenlogische bewijsbaarheid impliceert, maar de predikatenlogische theorie $\emptyset$ is niet volledig in de zin van de zojuist gegeven definitie van volledigheid voor theorieën.

De theorie van de partiële ordes is ook niet volledig. Hier zijn twee voorbeelden van structuren die aan de niet-logische axioma's van de theorie voldoen; in de ene is de zin $\forall x \forall y(Rxy \vee Ryx)$ waar, in de andere niet. Beide structuren zijn partiële ordes.



Kennelijk zit noch $\forall x \forall y(Rxy \vee Ryx)$ noch $\neg\forall x \forall y(Rxy \vee Ryx)$ in de deductieve afsluiting van de theorie van de partiële ordes. U gelieve zelf na te gaan waar deze conclusie op berust.

We stellen hier zonder bewijs dat de theorie van de dichte onbegrensde lineaire ordes volledig is. Voor het bewijs is een modeltheoretische exercitie nodig die buiten het bestek van dit boek valt. Hieruit, in combinatie met het feit dat de theorie in kwestie een eindige verzameling axioma's heeft, volgt de beslisbaarheid van de theorie, met behulp van de volgende stelling.

Stelling 6.10 *Als theorie T eindig geaxiomatiseerd is en volledig, dan is T beslisbaar.*

Bewijs: Omdat T eindig geaxiomatiseerd is mogen we aannemen dat de verzameling

$$\{\varphi \mid \varphi \text{ is een gesloten formule en } T \vdash \varphi\}$$

opsombaar is. Er is dus een programma dat de gesloten formules in de deductieve afsluiting van T opsomt. Uit de volledigheid van T volgt nu dat het volgende programma een beslissingsprogramma is voor de gesloten formules die uit T afleidbaar zijn. Neem weer aan dat φ een variabele is die een ingelezen formule bevat; *WelAfleidbaar* en *NietAfleidbaar* zijn boolese variabelen):

```
BEGIN
   n := 0;
   WelAfleidbaar := onwaar;
   NietAfleidbaar := onwaar;
   HERHAAL
        ALS φ gelijk is aan formule n in de uitvoer van
        het opsom-programma DAN
            WelAfleidbaar := waar;
        ANDERS
            BEGIN
                {in de nu volgende opdracht slaat 'concatenatie'
                op het aan-elkaar-plakken van schrijftekens: }
                ALS concatenatie('¬', φ) gelijk is
                aan formule n in de uitvoer
                van het opsom-programma DAN NietAfleidbaar := waar
            EINDE;
        n := n + 1
   TOTDAT WelAfleidbaar of NietAfleidbaar;
   ALS WelAfleidbaar DAN schrijf ( φ, ' is afleidbaar.')
   ANDERS schrijf (φ, ' is niet afleidbaar.')
EINDE.
```

Hiermee is het bewijs van de stelling geleverd. ∎

Deze stelling laat zien dat volledige theorieën, mits ze overzichtelijk zijn geaxiomatiseerd (in feite hoeft de verzameling axioma's niet eindig te zijn, als ze maar beslisbaar is), 'geknipt zijn voor de computer'. Echter, zulke theorieën zijn schaars. Voor logici en wiskundigen is dat trouwens maar gelukkig ook, want anders zou er, als we de computers eenmaal geprogrammeerd hebben voor de beslisprocedures van de wiskundige theorieën die ons interesseren, voor hen weinig meer te doen zijn.

Zoals Kurt Gödel in 1931 bewees: PA en ZF zijn *niet* volledig (mits ze consistent zijn). Dit zijn de beroemde *onvolledigheidsresultaten* van Gödel. De precieze gedachtengang vindt u weer in [Enderton 1972]. Meer populaire versies zijn te vinden in [Nagel & Newman 1975] en [Hofstadter 1979]. We geven een zeer beknopte samenvatting.

PA, de Peano-axiomatisering van de rekenkunde, is een verzameling axioma's die bedoeld is om de structuur bestaande uit de verzameling $\mathbf{N}$, met daarop de rekenkundige operaties, te beschrijven. Deze structuur duiden we voor het gemak aan met $\mathbf{N}$. Om aan te tonen dat PA onvolledig is, aangenomen dat deze theorie consistent is, construeert Gödel een zin ψ zo dat $\mathrm{PA} \nvdash \psi$, en zo dat ψ *waar* is voor de natuurlijke getallen (dat wil zeggen: $\mathbf{N} \models \psi$). Waarom impliceert het bestaan van deze ψ dat PA onvolledig is? Dat zit zo. Uit de volledigheid van de predikatenlogica volgt: als $\mathrm{PA} \vdash \neg\psi$, dan $\mathrm{PA} \models \neg\psi$. Maar vanwege het feit dat $\mathbf{N} \models \mathrm{PA}$ (alle Peano-axioma's zijn waar op de natuurlijke getallen) hebben we: $\mathbf{N} \models \neg\psi$, en dit is in strijd met het gegeven dat ψ waar was voor de natuurlijke getallen. Dus kan $\neg\psi$ niet afleidbaar zijn uit PA.

Een voor de hand liggende opmerking is nu: blijkbaar was PA te 'zuinig' als axiomatisering van de rekenkunde; laten we gewoon ψ aan de verzameling niet-logische axioma's toevoegen, en klaar. Dit lukt echter niet. Gödels constructievoorschrift is zo universeel dat voor $\mathrm{PA} \cup \{\psi\}$, de nieuwe theorie, weer een nieuwe voortvluchtige formule ψ' te construeren valt zodanig dat $\mathrm{PA} \cup \{\psi\} \nvdash \psi'$ en $\mathbf{N} \models \psi'$, enzovoorts.

Het basisidee achter de constructie van Gödels zin ψ stamt uit de filosofie van de Oudheid. Gödels zin houdt verband met de zogenaamde 'Leugenaarparadox' van de stoïcijnen ($\pm$300 voor Christus). Epimenides de Cretenzer zegt: "Alle Cretenzers liegen (altijd)". Deze bewering kan niet waar zijn. Immers, als Epimenides de waarheid spreekt, dan kunnen we uit de inhoud van zijn uitspraak afleiden dat Epimenides—zelf een Cretenzer—altijd liegt, dus dan is zijn uitspraak niet waar, en hebben we een tegenspraak. De uitspraak van Epimenides weerlegt dus zichzelf. Dit is een voorbeeld van wat we een 'halve' paradox zouden kunnen noemen, in tegenstelling tot een 'volkomen' paradox als de Russell-paradox die we in § 6.7 zijn tegengekomen. Bij een 'volkomen' paradox leidt zowel de aanname dat zekere bewering waar is als de aanname dat ze onwaar is tot een tegenspraak. Gödel had voor zijn onvolledigheids-redenering een halve paradox nodig omdat hij één vluchtweg wilde afsluiten: de halve paradox dient om de mogelijkheid uit te sluiten dat de formule ψ waar het verhaal om draait afleidbaar is in PA.

Opdracht 6.45 *Laat zien dat "Dit boek bevat fouten" een voorbeeld is van een halve paradox.*

Het is overigens niet moeilijk de leugenaar-paradox zo te herformuleren dat het een volkomen paradox wordt. "Ik lieg nu", en "Deze zin is onwaar" zijn voorbeelden van volkomen paradoxen.

Opdracht 6.46 *Laat zien dat de bewering "Deze zin is onwaar" een volkomen paradox is.*

Liefhebbers van paradoxen moeten [Hughes & Becht 1978] lezen; fraaie puzzelvariaties op de leugenaarparadox zijn te vinden in [Smullyan 1978] en [Smullyan 1982]. Dit laatste boek biedt een smeuïge en informele inleiding in de achtergronden van Gödel's onvolledigheids-resultaten.

Gödels bewering ψ maakt gebruik van het feit dat de theorie van de rekenkunde rijk genoeg is om—door middel van een geschikte *codering*—met behulp van rekenkundige zinnen te kunnen praten over de *bewijsbaarheid* van andere rekenkundige zinnen. De leugenaarparadox krijgt nu de volgende vorm: ψ drukt uit dat *hijzelf* niet in PA afleidbaar is. Met andere woorden:

$$\psi = \text{“PA} \nvdash \psi\text{”}.$$

Als $\text{PA} \vdash \psi$, dan is ψ waar (dat wil zeggen $\mathbf{N} \models \psi$), want de stellingen van PA zijn ware beweringen over de natuurlijke getallen, en dus weten we, op grond van wat ψ uitdrukt, dat $\text{PA} \nvdash \psi$, en we hebben een tegenspraak. Dus moet gelden:

$$\text{PA} \nvdash \psi.$$

Maar dit is nu juist wat ψ zelf uitdrukte. Vanwege de volledigheid van de predikatenlogica volgt hieruit: ψ is waar. Als ψ waar is, dan kan $\neg\psi$ niet waar zijn, en wat niet waar is, is niet bewijsbaar (anders zouden de PA-axioma's niet *correct* zijn). Dus hebben we ook:

$$\text{PA} \nvdash \neg\psi.$$

Zo, een grovere schets is haast niet denkbaar, maar toch laten we het hierbij. U heeft nu de grote lijn van de onvolledigheidsstelling van Gödel gezien, en als u nieuwsgierig bent geworden naar de technische details kunt u de genoemde literatuur raadplegen. We besluiten met een opdracht die een variant is op een raadsel uit [Smullyan 1982]:

Opdracht 6.47 *Een* Gödel-schrijfmachine *is een schrijfmachine die door middel van het afdrukken van symbolen beweringen doet over zijn eigen afdruk-mogelijkheden. We nemen aan dat de machine correcte uitspraken doet over wat hij kan afdrukken. Met andere woorden: wanneer de machine zegt dat hij een rijtje symbolen wel of niet kan afdrukken, dan is dat ook zo.*

De machine gebruikt alleen de tekens `A`*,* `N` *en* `V`*. Om te kunnen omschrijven wat de betekenis is hebben we een metavariabele X nodig voor rijtjes symbolen uit $\{\mathtt{A}, \mathtt{N}, \mathtt{V}\}$. Als X een rijtje is, dan noemen we XX de verdubbeling van X. De betekenis van de rijtjes die de machine afdrukt wordt gegeven door de volgende clausules:*

- `A`X *is waar desda het rijtje X afdrukbaar is.*
- `NA`X *is waar desda het rijtje X niet afdrukbaar is.*

- `VA`X *is waar desda de verdubbeling van* X *afdrukbaar is.*
- `VNA`X *is waar desda de verdubbeling van* X *niet afdrukbaar is.*

Wat is nu de simpelste ware bewering, uitgedrukt als een rijtje van tekens uit $\{\texttt{A}, \texttt{N}, \texttt{V}\}$*, die de machine* niet *kan afdrukken?*

Hoofdstuk 7

Uitbreidingen van de predikatenlogica

7.1 Meersoortige predikatenlogica

In de semantiek van de predikatenlogica zoals we die in § 6.3 gepresenteerd hebben was sprake van een domein van individuen. Alle variabelen werden toebedeeld aan objecten in dit ene domein. In de standaard-predikatenlogica zijn—zo zou je kunnen zeggen—alle objecten waarover wordt gekwantificeerd van hetzelfde soort. Wanneer je onderscheid wilt maken moet je dat doen met behulp van een geschikt predikaat. Neem de volgende twee zinnen:

(1) *Sommige mensen komen ongelegen.*

(2) *Sommige tijdstippen komen ongelegen.*

Neem als vertaalsleutel: Mx: x is een mens; Tx: x is een tijdstip; Oxy: x komt ongelegen op moment y; het discussiedomein is: mensen en tijdstippen. De predikatenlogische vertalingen worden nu, respectievelijk:

(3) $\exists x(Mx \wedge \exists y Oxy)$.

(4) $\exists y(Ty \wedge \exists x Oxy)$.

Door deze manier van onderscheidingen maken met behulp van predikaten kunnen de vertalingen behoorlijk ingewikkeld worden.

(5) *Iedereen komt soms ongelegen,*
maar sommigen komen altijd ongelegen.

De vertaling wordt:

(6) $\forall x(Mx \rightarrow \exists y(Ty \wedge Oxy)) \wedge \exists z(Mz \wedge \forall u(Tu \rightarrow Ozu))$.

De ingewikkeldheid van de vertalingen is op zichzelf niet zo erg—een kwestie van wennen, nietwaar—maar er is nog een andere moeilijkheid: je zou kunnen zeggen dat de formule Oxy (voor: 'x komt ongelegen op y') alleen zinnig kan worden geïnterpreteerd wanneer het eerste argument van het predikaat O een mens is en het tweede een tijdstip. Immers: 'Vastenavond komt ongelegen op Jan' is niet gewoon onwaar, het is gewoon onzin. Het is een beetje vervelend dat dit soort onzin *wel* predikatenlogisch kan worden uitgedrukt, gegeven deze vertaalsleutel. 'Jan komt ongelegen op vastenavond' kan daarentegen zowel waar als onwaar zijn. (Overigens moet *vastenavond* eigenlijk eerder geïnterpreteerd worden als een *tijdsspanne* dan als een tijdstip. We zullen daar echter nu niet moeilijk over doen.)

Remedie: stap over van de gewone predikatenlogica naar *meersoortige* predikatenlogica. In plaats van een individuendomein hebben we nu meerdere individuendomeinen, een voor elke 'soort' van individuen. Bij kwantificatie moet nu worden aangegeven over welke soort van individuen de kwantoren lopen. Dat kan op twee manieren (en het maakt niets uit welke manier we kiezen): we kunnen bij iedere kwantor met behulp van een *index* aangeven over welke *soort* hij loopt, of we kunnen elke soort zijn eigen variabelen-verzameling geven. We geven een schets van beide werkwijzen voor een predikatenlogische taal met twee soorten: de soort *mens* en de soort *tijdstip*.

Eerste notatievariant: de kwantoren $\forall, \exists$ worden vervangen door $\forall_m$, $\exists_m$, $\forall_t$, $\exists_t$. De verzameling variabelen blijft: $x, y, z, u, v, \ldots$ De vertaling van (5) wordt nu:

(7) $\forall_m x \exists_t y Oxy \wedge \exists_m z \forall_t u Ozu$.

Nog een paar vertalingen:

(8) *Sommige mensen komen soms ongelegen* $\rightsquigarrow \exists_m x \exists_t y Oxy$.

(9) *Altijd is Kortjakje ziek* $\rightsquigarrow \forall_t x Zkx$.

Voor dit laatste voorbeeld is er nog een preciezer vertaling mogelijk, voor het geval je ook nog expliciet zou willen uitdrukken dat k de naam is van een mens. Dat kan met een trucje:

(10) $\exists_m x \forall_t y(k = x \wedge Zxy)$.

Opdracht 7.1 *Vertaal op deze manier, na een discussiedomein met meerdere soorten van objecten en een vertaalsleutel te hebben gegeven:*

1. *Soms is Kortjakje ziek, maar niet altijd.*

2. *Elke zondag gaat Kortjakje naar een kerk.*

3. *Iedereen is soms ziek.*

Tweede notatievariant: de kwantoren $\forall$ en $\exists$ blijven gehandhaafd, maar we voeren twee verschillende verzamelingen van variabelen in:

- $x, x', x'', \ldots$ voor mensen;
- $t, t', t'', \ldots$ voor tijdstippen.

De vertalingen worden nu, respectievelijk:

$$\forall x \exists t Oxt \wedge \exists x' \forall t' Ox't' \ ; \ \exists x \exists t Oxt \ ; \ \forall t Zkt.$$

Het Kortjakje-voorbeeld maakt duidelijk dat we ook voor de constanten van de taal zouden moeten aangeven in welke soort ze geïnterpreteerd moeten worden. Voor de predikaatletters geldt iets analoogs: voor elke predikaatletter moet voor elke argumentplaats worden vastgelegd over welke soort hij loopt. Voorbeeld: Oxy voor '(mens) x komt ongelegen op (tijdstip) y' zou eigenlijk moeten worden genoteerd als $O_{\langle m,t\rangle}xy$, om aan te geven dat de eerste argumentplaats in de soort *mens* moet worden geïnterpreteerd, de tweede in de soort *tijdstip*. Jargon: het paar $\langle m, t\rangle$ wordt wel een *soort-type* genoemd.

Opdracht 7.2 *Vertaal de zinnen uit opdracht 7.1 op de manier van deze tweede notatiewijze.*

We zullen de semantiek voor meersoortige predikatenlogica hier niet helemaal uitspellen. Globaal komt het er op neer dat het domein D wordt opgesplitst in subdomeinen voor de verschillende soorten. Dus voor het mensen-tijdstippen voorbeeld zou de structuur er zo uitzien $\langle D_m, D_t, I\rangle$. D_m zijn de mensen, D_t de tijdstippen. We mogen wel aannemen dat geen enkel object zowel een mens als een tijdstip is: de doorsnede van D_m en D_t is leeg. Dit laatste kan overigens van geval tot geval verschillen; in het algemeen hoeft het niet zo te zijn dat de subdomeinen voor de verschillende soorten disjunct zijn.

In systemen voor het 'vertalen' van natuurlijke taal zinnen in logische formules die momenteel worden ontwikkeld in het kader van Kunstmatige Intelligentie onderzoek speelt het gebruik van soort-type restricties een belangrijke rol bij het uitsluiten van niet-plausibele lezingen van zinnen. Zoals reeds opgemerkt zijn formules waarin gezondigd wordt tegen bepaalde soort-type restricties, bij voorbeeld de formule die de vertaling is van

"Vastenavond komt ongelegen op Jan", eerder onzinnig dan onwaar. Dit feit kan worden verdisconteerd door de valuatiefunctie *partieel* te maken (zie § 2.8). Bij voorbeeld voor een atomaire formule als $O_{\langle m,t\rangle}ab$ geldt dan:

$$\begin{array}{lcl} V(O_{\langle m,t\rangle}ab) = 1 & \Longleftrightarrow & \langle I(a), I(b)\rangle \in I(O_{\langle m,t\rangle}) \\ V(O_{\langle m,t\rangle}ab) = 0 & \Longleftrightarrow & \langle I(a), I(b)\rangle \notin I(O_{\langle m,t\rangle}) \\ & & \;\&\; I(a) \in D_m \;\&\; I(b) \in D_t \\ V(O_{\langle m,t\rangle}ab) = \# & \Longleftrightarrow & I(a) \notin D_m \;\text{ of }\; I(b) \notin D_t. \end{array}$$

De valuatie van de connectieven gaat dan volgens de zwakke waarheidstafels uit § 5.10. In het vervolg zullen we omwille van de eenvoud de valuatiefunctie gewoon *totaal* houden.

De interpretatiefunctie I moet aan de eisen voldoen die we hierboven hebben aangestipt: constanten die namen zijn voor mensen moeten interpretaties in D_m hebben, interpretaties voor namen van tijdstippen moeten juist in D_t zitten. Predikaatletters moeten worden geïnterpreteerd in overeenstemming met hoe de argumentplaatsen geïndiceerd zijn voor de soorten (bij voorbeeld $O_{\langle m,t\rangle}$ dient geïnterpreteerd te worden als een relatie tussen D_m en D_t). De identiteits-relatie $=$ dient op elke soort als identiteit te worden geïnterpreteerd; in ons voorbeeld: we moeten onderscheiden tussen $=_{\langle m,m\rangle}$, te interpreteren als de identiteitsrelatie op D_m, en $=_{\langle t,t\rangle}$, te interpreteren als de identiteitsrelatie op D_t.

Is er nu, met de overgang van enkelsoortige naar meersoortige logica, essentieel iets veranderd? Met andere woorden: worden bepaalde gevolgtrekkingen geldig die dat vroeger niet waren of andersom? Het antwoord is: 'Nee'. Formules van een meersoortige predikatenlogische taal kunnen altijd worden *terugvertaald* naar formules van een enkelsoortige predikatenlogische taal met voor elke soort een eenplaatsige predikaatletter. In het voorbeeld: voer gewoon M weer in voor 'is een mens' en T voor 'is een tijdstip'. De terugvertaal-instructies luiden dan:

$$\forall_m x \ldots \rightsquigarrow \forall x(Mx \rightarrow \ldots)$$

$$\forall_t x \ldots \rightsquigarrow \forall x(Tx \rightarrow \ldots)$$

$$\exists_m x \ldots \rightsquigarrow \exists x(Mx \wedge \ldots)$$

$$\exists_t x \ldots \rightsquigarrow \exists x(Tx \wedge \ldots).$$

Opdracht 7.3 *Vertaal alle voorbeeld-formules uit de meersoortige logica die u in deze paragraaf bent tegengekomen terug naar de gewone, enkelsoortige predikatenlogica, met behulp van bovenstaande terugvertaal-instructies.*

De formules die we zo krijgen kunnen worden geïnterpreteerd in de modellen die ontstaan door de subdomeinen van een meersoortig model samen te voegen en de soort-predikaatletters te interpreteren als de vroegere subdomeinen. Weer voor het voorbeeld: laat $\langle D_m, D_t, I\rangle$ een meersoortig model zijn. Dan is $\langle D_m \cup D_t, I^*\rangle$ het corresponderende enkelsoortige model. Hierbij duidt I^* de interpretatiefunctie aan die is als I, maar uitgebreid met een interpretatie voor M en T, namelijk $I^*(M) = D_m$ en $I^*(T) = D_t$. Als we de terugvertaling van een meersoortige formule φ noteren als φ^*, en het enkelsoortige model dat correspondeert met het meersoortige model $\mathcal{M}$ als $\mathcal{M}^*$, dan hebben we:

$$\mathcal{M} \models \varphi \iff \mathcal{M}^* \models \varphi^*.$$

Hieruit blijkt dat meersoortige predikatenlogica niet meer is dan een *variant* van de gewone predikatenlogica. Het gebruik ervan kan weleens *handiger* zijn dan dat van gewone predikatenlogica, maar dat is dan ook alles.

Een andere manier om een effect te krijgen als van meersoortige logica, maar met een nog kleinere afwijking van de gewone predikatenlogica, is door het invoeren per definitie van *beperkte kwantificatie*. Men neme een predikatenlogische taal en men spreke af: voor elke eenplaatsige predikaatletter A en elke formule φ van de taal is $\forall x \in A : \varphi$ een afkorting voor $\forall x(Ax \to \varphi)$, en $\exists x \in A : \varphi$ een afkorting voor $\exists x(Ax \wedge \varphi)$. Omdat we alleen maar nieuwe afkortingen hebben ingevoerd verandert er niets aan de semantiek. Een paar vertalingen ter illustratie.

(11) *Alle mensen zijn sterfelijk*
$\rightsquigarrow \quad \forall x{\in}M : Sx.$

(12) *Sommige mensen komen altijd ongelegen*
$\rightsquigarrow \quad \exists x{\in}M \forall y{\in}T : Oxy.$

U begrijpt hopelijk dat de keuze 'beperkte kwantificatie of niet' een zuiver *notationele* keuze is. Beperkte kwantificatie is linguïstisch gezien wel handig omdat de bijdrage van nominale constituenten als geheel in de vertalingen nu beter terug te vinden is:

(13) Niemand *is onsterfelijk*
$\rightsquigarrow \quad \neg\exists x{\in}M : \neg Sx.$

(14) Alle mensen *zijn sterfelijk*
$\rightsquigarrow \quad \forall x{\in}M : Sx.$

Opdracht 7.4 *Kies een geschikt discussiedomein, geef een vertaalsleutel, en vertaal de volgende zinnen met behulp van beperkte kwantificatie:*

1. *Niet iedereen is aardig.*

2. *Sommige vrouwen minachten iedere man.*

3. *Als niemand iets weet verklapt niemand iets.*

4. *Iedereen houdt van iemand, en sommigen houden van iedereen.*

De linguïstische voordelen van beperkte kwantificatie zijn echter nogal betrekkelijk. Uniek bepalende beschrijvingen (*de koningin*, *de directeur*, *de dekaan van de faculteit*) zijn, als we ze Russelliaans aanpakken (vergelijk § 6.4), nog steeds wat moeilijk terug te vinden. *De koningin is jarig* wordt nu:

$$\exists x \in K : (\forall y \in K : y = x \wedge Jx).$$

Verder mogen we de ogen niet sluiten voor het feit dat we met de nu voorhanden middelen de volgende nominale constituenten nog helemaal niet kunnen analyseren:

(15) De meeste taalkundigen *houden van goede wijn.*

(16) De helft van de bevolking *is tegen kernenergie (maar wil wel veel en goedkope electriciteit).*

In [Barwise & Cooper 1981] wordt bewezen dat *de meeste A zijn B* en *de helft van de A zijn B* niet in eerste orde predikatenlogica kunnen worden uitgedrukt, gesteld tenminste dat we behalve de eenplaatsige predikaatletters A en B geen predikaatletters ter beschikking hebben. Als er een extra tweeplaatsige predikaatletter R mag worden gebruikt en we ons beperken tot eindige modellen kan het weer wel: zie [Thijsse 1983]. De behandeling van uitdrukkingen als *de helft* en *de meeste* komt in § 7.3 aan de orde. Toch geldt juist voor deze hogere orde kwantoren dat ze *essentieel* beperkt zijn. "De meeste A zijn B" is bij voorbeeld niet weer te geven als "Voor de meeste dingen geldt: als ze A zijn, dan zijn ze ook B."

7.2 Tweede orde logica

De predikaten-logica zoals gepresenteerd in hoofdstuk 6 heet ook wel *eerste orde logica*. De reden: de objecten waarover in de gewone predikatenlogica wordt gekwantificeerd heten, in de terminologie van Bertrand Russell, *objecten van de eerste orde*. Russell deelde objecten in naar de interne verzamelingstheoretische structuur die ze hebben. *Objecten van de eerste orde* zijn objecten die zelf niet weer verzamelingen zijn, of liever: objecten waarvan de eventuele verzamelingstheoretische interne structuur niet door de

taal in kwestie beschreven wordt. Een theelepeltje is dus een object van de eerste orde. Een *verzameling* van theelepeltjes heeft een structuur die een graadje ingewikkelder is; Russell noemt dit daarom een *object van de tweede orde*. Een verzameling die bestaat uit verzamelingen theelepeltjes en verzamelingen suikerklontjes is een *object van de derde orde*, enzovoorts.

Tweede orde logica ontstaat door ook kwantificatie toe te laten over objecten van de tweede orde. Als een domein van (eerste orde) objecten D gegeven is, dan zijn alle deelverzamelingen van D en alle twee- of meerplaatsige relaties op D objecten van de tweede orde.

De predikaatletters uit de eerste orde logica waren (eerste orde) predikaat*constanten*: namen voor predikaten (eigenschappen of relaties) die gedefinieerd zijn voor objecten van de eerste orde. Daarnaast staan we nu ook (eerste orde) predikaat*variabelen* toe. Laat $A, B, C \ldots$ eerste orde predikaatconstanten zijn en $X, Y, Z \ldots$ eerste orde predikaatvariabelen. Nu kunnen we kwantificeren over eerste orde eigenschappen; die eigenschappen worden dan zelf beschouwd als objecten van de tweede orde. Kortom: de interpretatie van een term is een object van de eerste orde, en de interpretatie van een eerste orde predikaat is een object van de tweede orde.

(17) *Jan heeft een eigenschap die Marie niet heeft*
$\rightsquigarrow \quad \exists X(Xj \wedge \neg Xm).$

(18) *Jan en Marie hebben geen enkele eigenschap gemeen*
$\rightsquigarrow \quad \forall X(Xj \rightarrow \neg Xm).$

"Piet heeft alle kenmerken van een machistische man" kan worden geparafraseerd als: "Piet heeft alle kenmerken die elke machistische man heeft", dus de vertaling in tweede orde logica ziet er als volgt uit:

(19) $\forall X(\forall x((Mx \wedge M'x) \rightarrow Xx) \rightarrow Xp).$

Opdracht 7.5 *Vertaal nu zelf: "Pinochet heeft alle kenmerken van een gevreesd dictator, en geen enkel kenmerk van een geliefd staatshoofd."*

De filosoof Leibniz bedacht al in de zeventiende eeuw dat *identiteit* te definiëren valt met behulp van kwantificatie over eigenschappen van dingen. Hij stelde de volgende twee principes op:

- **Principe van de gelijkheid van wat niet te onderscheiden valt**
  $\forall x \forall y(\forall X(Xx \rightarrow Xy) \rightarrow x = y).$
- **Principe van de ononderscheidbaarheid van wat identiek is**
  $\forall x \forall y(x = y \rightarrow \forall X(Xx \rightarrow Xy)).$

Samen leveren deze principes de volgende definitie van $=$ op:

Definitie 7.1 [identiteit] $x = y$ *betekent* $\forall X(Xx \rightarrow Xy)$.

Dat dit mag volgt uit de combinatie van de twee principes van Leibniz: $\forall x \forall y(x = y \leftrightarrow \forall X(Xx \rightarrow Xy))$.

Opdracht 7.6 *Geef een parafrase van de zin "De koning toornt" in tweede orde logica, zonder gebruik te maken van het identiteitssymbool.*

Opdracht 7.7 *In bovenstaande identiteitsprincipes ligt het misschien meer voor de hand* $Xx \leftrightarrow Xy$ *te gebruiken in plaats van* $Xx \rightarrow Xy$. *Laat evenwel zien dat* $\forall X(Xx \rightarrow Xy)$ *equivalent is met* $\forall X(Xx \leftrightarrow Xy)$.

Nu, zult u zeggen, wanneer tweede orde predikatenlogica blijkbaar zoveel uitdrukkingskracht heeft dat we zelfs identiteit erin kunnen definiëren, waarom dan niet voor eens en voor altijd overgestapt van eerste- naar tweede orde logica? De kink in de kabel is dat we dan de mooie correspondentie tussen $\vdash$ en $\models$ die we in de eerste orde logica hadden kwijt zouden zijn: er bestaat geen 'mooie' (dat wil zeggen: recursief definieerbare) verzameling axioma's die als stellingen precies de verzameling van universeel geldige tweede orde formules oplevert.

Niettemin speelt kwantificatie over tweede orde objecten (tweede orde logica dus) in semantische theorieën over natuurlijke taal vaak een grote rol. Een belangrijke toepassing is de theorie van de zogenaamde *gegeneraliseerde kwantoren* voor het verantwoorden van de betekenis van nominale constituenten.

7.3 Gegeneraliseerde kwantoren-theorie

We hebben in 7.1 *beperkte kwantificatie* geïntroduceerd als een notatievariant van gewone eerste orde existentiële en universele kwantificatie. Een voorbeeld van een formule met een beperkte universele kwantor is: $\forall x \in A : Bx$. Een formule met een beperkte existentiële kwantor: $\exists x \in A : Bx$. Het idee van de gegeneraliseerde kwantoren-theorie is dat zo'n beperkte kwantor-formule eigenlijk iets zegt over de manier waarop *twee verzamelingen* (twee tweede orde objecten dus) zich tot elkaar verhouden. Bij de voorbeeld-formules zijn dat, gegeven zeker model $\mathcal{M} = \langle D, I \rangle$, de verzamelingen $I(A)$ en $I(B)$.

Heel algemeen zeggen we nu: een kwantor is een relatie tussen tweede orde objecten (verzamelingen). In het geval van de universele kwantor is het de relatie *bevat zijn in*. Immers: "Alle A zijn B" is waar in model $\mathcal{M}$ desda $I(\mathrm{A}) \subseteq I(\mathrm{B})$. In het geval van de existentiële kwantor is het de relatie *een niet-lege doorsnede hebben met*. Gaat u maar na: "Minstens één A is B" is waar desda $I(\mathrm{A}) \cap I(\mathrm{B}) \neq \emptyset$.

Goed, de beperkte universele kwantor en de beperkte existentiële kwantor kunnen dus worden beschouwd als relaties op de machtsverzameling van het domein D van onze modellen. Formeel: de interpretatie van *alle* in model $\mathcal{M} = \langle D, I\rangle$ is de relatie

$$\{\langle A, B\rangle |\ A \subseteq B \subseteq D\}.$$

De interpretatie van *minstens één* in model $\mathcal{M} = \langle D, I\rangle$ is de relatie

$$\{\langle A, B\rangle |\ A \subseteq D \text{ en } B \subseteq D \text{ en } A \cap B \neq \emptyset\}.$$

Gegeven dit perspectief ligt het nu voor de hand om ook *andere* relaties op de machtsverzameling van D te beschouwen dan alleen deze twee. Er blijkt dan meteen dat we nu ook niet-standaard kwantificatie kunnen behandelen, zoals in "minstens twee A zijn B" en "minstens de helft van de A zijn B". Hiertoe maken we gebruik van de notatie $|A|$ voor 'het aantal elementen dat A bevat'. De interpretatie van *minstens twee* wordt nu (gegeven domein D):

$$\{\langle A, B\rangle |\ A \subseteq D \text{ en } B \subseteq D \text{ en } |A \cap B| \geq 2\}.$$

Minstens de helft wordt geïnterpreteerd als de volgende relatie op de machtsverzameling van het individuendomein D:

$$\{\langle A, B\rangle |\ A \subseteq D \text{ en } B \subseteq D \text{ en } |A - B| \leq |A \cap B|\}.$$

Opdracht 7.8 *Welke relatie op $\mathcal{P}(D)$ kan nu dienen als de interpretatie van 'hoogstens vijf'?*

Opdracht 7.9 *Welke relatie op $\mathcal{P}(D)$ kan dienen als interpretatie van 'geen'?*

Opdracht 7.10 *Welke relatie op $\mathcal{P}(D)$ kan dienen als interpretatie van 'niet alle'?*

U ziet aan de opdrachten: we hebben hier een theorie te pakken die ons de middelen in handen geeft om—heel algemeen—de interpretatie van de lidwoordgroep in nominale constituenten ter hand te nemen. Dankzij het gegeneraliseerde kwantoren-perspectief is het blikveld niet langer beperkt tot existentiële en universele kwantificatie, maar komen ook andere kwantoren aan bod. Een lidwoord-groep als *geen*, *minstens twee*, *niet alle*, *precies drie*, *hoogstens vier* wordt wel een *determinator* genoemd (Engels: *determiner*); diezelfde term is trouwens ook in gebruik voor de relatie op $\mathcal{P}(D)$ die dient als *interpretatie* voor zo'n lidwoordgroep. Een kwantificerende determinator wordt een *kwantor* genoemd.

Is het nu zo dat *elke* relatie op $\mathcal{P}(D)$ de interpretatie is van een kwantor? Dat ligt er natuurlijk maar aan wat we willen verstaan onder een kwantor, maar het ligt voor de hand om een paar beperkingen op te leggen. Daartoe zullen we een tweetal *eigenschappen* van determinatoren (determinator-interpretaties) gaan isoleren.

In de eerste plaats willen we de intuïtie uitdrukken dat zinnen als *Alle jongens wandelen*, *Minstens twee jongens hollen*, *Hoogstens drie jongens praten* betrekking hebben op de verzameling *jongens* in het discussiedomein. Alleen het gedrag van de jongens in het domein maakt iets uit voor de waarheidswaarde van deze zinnen. Of de andere individuen in het domein wandelen, lopen of praten doet er niet toe. We hoeven dus, om te kijken of deze zinnen waar zijn, niet *alle* wandelaars, hardlopers of praters te onderzoeken, maar alleen de wandelende jongens, de hollende jongens, of de pratende jongens. Nog anders gezegd: de volgende parafrases veranderen de betekenis van de bovenstaande zinnen niet: *Alle jongens zijn wandelende jongens*; *Minstens twee jongens zijn hollende jongens*; *Hoogstens drie jongens zijn pratende jongens*. Merk verder op dat het voor de waarheidswaarde van deze zinnen niet uitmaakt of we het discussiedomein groter maken of kleiner, zolang we de verzameling jongens maar intact laten. Het is de verzameling jongens die bepalend is voor de vraag of de kwantorrelatie al dan niet geldt. Algemeen kunnen we dus zeggen:

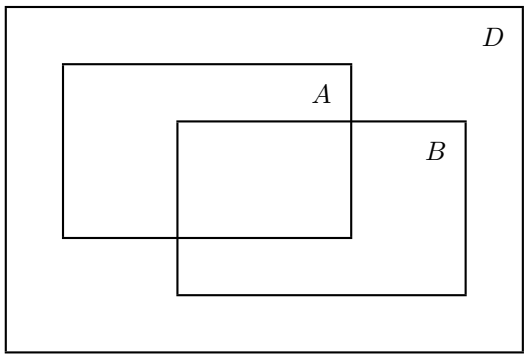
> Als een kwantor-relatie op de machtsverzameling van het domein D geldt tussen een verzameling A en een verzameling B, dan geldt die relatie ook op de machtsverzameling van het *subdomein* A tussen A en de *doorsnede* van A en B, en vice versa.

Deze eigenschap noemen we de *conservativiteitseigenschap* van de relatie. We gebruiken R_D voor een tweeplaatsige relatie op $\mathcal{P}(D)$. We kunnen als volgt uitdrukken dat R_D conservatief is:

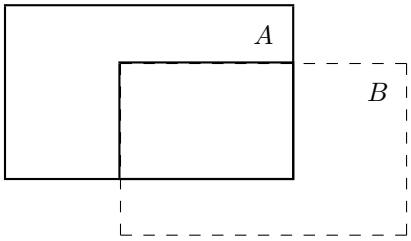
Definitie 7.2 Conservativiteit van R_D

Voor alle $A, B \subseteq D$*:* $R_D AB$ *desda* $R_A A(A \cap B)$.

Om te zien wat *conservativiteit* precies uitdrukt maken we een plaatje van een domein D, met twee verzamelingen A en B.



Stel dat een kwantor-relatie op $\mathcal{P}(D)$ geldt tussen A en B. Dan zegt *conservativiteit* dat die relatie ook geldt tussen A en $A \cap B$, als we het domein van de kwantor-relatie beperken tot $\mathcal{P}(A)$, en vice versa. Dit betekent dat we het bovenstaande plaatje zonder bezwaar mogen vervangen door het volgende plaatje:



Denk bij voorbeeld bij A aan de verzameling jongens, bij B aan de verzameling praters. Dan is $A \cap B$ de verzameling pratende jongens. Buiten de verzameling jongens, het eerste argument van de relatie, hoeven we niet te kijken. We zeggen dat een kwantor-relatie *teert op* zijn eerste argument. In het bovenstaande voorbeeld: de verzameling A (de verzameling jongens). De conservativiteits-eigenschap wordt ook wel de *teren-op eigenschap* (Engels: *the live-on property*) genoemd.

Uit de conservativiteits-eigenschap volgt meteen dat we de index D van een kwantor-relatie R_D gerust mogen weglaten. Voor de interpretatie doet het gedeelte van het domein dat ligt buiten de verzameling waarop de kwantor-relatie teert er immers niet toe.

Een voor de hand liggende tweede eis die we aan kwantor-relaties kunnen stellen is dat ze inderdaad iets te maken hebben met kwantiteit. De conservativiteits-eis zegt ons dat het antwoord op de vraag of R geldt tussen A en B alleen mag afhangen van de verzamelingen A en $A \cap B$. De

kwantiteits-eis voegt daar nog iets aan toe: het gaat niet om deze verzamelingen zonder meer, maar alleen om de *aantallen elementen* die ze bevatten. "Alle jongens wandelen" is waar precies wanneer het *aantal* jongens dat niet wandelt 0 is; "Hoogstens drie jongens praten" is waar wanneer het *aantal* jongens dat praat ≤ 3 is, enzovoorts. De kwantiteits-eis op kwantor-relaties wordt nu:

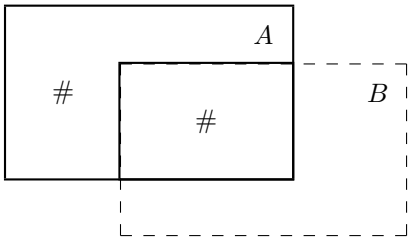
> Als de kwantor-relatie R geldt tussen twee verzamelingen A en B, en we hebben twee andere verzamelingen C en D waarvan we weten dat C evenveel elementen heeft als A, en de doorsnede van C en D evenveel elementen als de doorsnede van A en B, dan weten we dat de kwantor-relatie ook moet gelden tussen C en D.

U gelieve dit zelf te controleren aan de hand van concrete voorbeelden als "Alle jongens wandelen", "Minstens twee jongens hollen", "Hoogstens drie jongens praten", "De helft van de jongens voetbalt". Iets formeler kunnen we de kwantiteits-eigenschap als volgt formuleren:

Definitie 7.3 Kwantiteits-eigenschap van R

Voor alle X, Y, Z, U*:*
als $R(X,Y)$ *en* $|X| = |Z|$ *en* $|X \cap Y| = |Z \cap U|$*, dan* $R(Z,U)$.

Wanneer we nu weer een plaatje gaan tekenen van twee verzamelingen A en B waartussen een kwantor-relatie geldt, dan zien we dat we ons kunnen beperken tot het aangeven van twee kardinaalgetallen, (wanneer we ons beperken tot eindige domeinen: twee natuurlijke getallen). Het eerste getal geeft het aantal elementen in $A - B$ aan, het tweede getal het aantal elementen in $A \cap B$:



De twee #-tekens in het plaatje geven de getallen $|A - B|$ en $|A \cap B|$ aan. We hadden natuurlijk ook de twee getallen $|A|$ en $|A \cap B|$ kunnen nemen: als we $|A|$ hebben weten we ook $|A - B|$ en omgekeerd, vanwege de betrekking

$|A - B| = |A| - |A \cap B|$ (we gaan er nu even van uit dat de verzamelingen A en B eindig zijn).

Aangenomen dat kwantor-relaties voldoen aan *conservativiteit* en *kwantiteit* kunnen we *elke* kwantor-relatie karakteriseren als een patroon van plaatsen in een oneindige getallen-piramide. Ook wanneer we ons beperken tot eindige domeinen is de getallen-piramide oneindig: er is immers geen bovengrens aan de grootte van onze domeinen. Hier volgt het recept voor het construeren van de getallen-piramides. Elke laag van de piramide behandelt de verdeling van $|A - B|$ en $|A \cap B|$, voor een bepaald aantal $|A|$. De top van de piramide bestaat uit het ene getallenpaar $\langle 0, 0\rangle$, want als er 0 A's zijn is dit de enig mogelijke verdeling. De top vormde de eerste laag. De tweede laag behandelt het geval waar er 1 A is; zij bestaat uit de getallenparen $\langle 1, 0\rangle$ ("er is 1 ding dat A is maar niet B, en er zijn geen dingen die zowel A als B zijn") en $\langle 0, 1\rangle$ ("er zijn geen dingen die alleen A zijn maar niet B, en er is 1 ding dat zowel A als B is"). Net zo voor de volgende lagen. De hele piramide ziet er dus zo uit:

$\lvert A\rvert = 0$									$0,0$											
$\lvert A\rvert = 1$								$1,0$			$0,1$									
$\lvert A\rvert = 2$						$2,0$			$1,1$			$0,2$								
$\lvert A\rvert = 3$				$3,0$			$2,1$			$1,2$			$0,3$							
$\lvert A\rvert = 4$			$4,0$			$3,1$			$2,2$			$1,3$			$0,4$					
$\lvert A\rvert = 5$		$5,0$			$4,1$			$3,2$			$2,3$			$1,4$			$0,5$			
$\vdots$									$\vdots$											

Heel fraai is, dat we nu kwantor-relaties kunnen karakteriseren met behulp van een *patroon* van + en − dat ze in deze piramide teweeg brengen: een + verschijnt op de plaatsen van de getallenparen die de aantallen $|A - B|$ en $|A \cap B|$ aangeven van de verzamelingen A en B waarvoor de kwantor-relatie geldt, een − op alle andere plaatsen. Het volgende voorbeeld maakt dit hopelijk duidelijk. "Alle A zijn B" is immers waar als $A \subseteq B$ (eigenlijk: als $I(A) \subseteq I(B)$, maar hierover doen we even niet moeilijk). Maar $A \subseteq B$ is precies het geval als $A - B = \emptyset$, en dit is weer equivalent met $|A - B| = 0$.

alle A zijn B

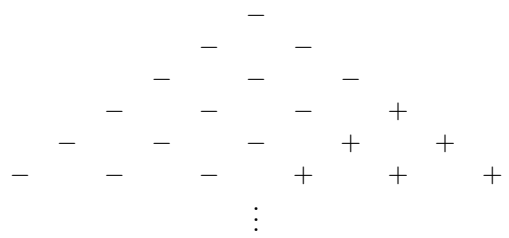
									+											
							−			+										
					−			−			+									
				−			−			−			+							
			−			−			−			−			+					
		−			−			−			−			−			+			
									$\vdots$											

Opdracht 7.11

1. *Geef het piramide-patroon voor* geen.
2. *Geef het piramide-patroon voor* niet alle.
3. *Geef het piramide-patroon voor* minstens één.
4. *Hoe verhouden deze piramide-patronen zich tot elkaar en tot het patroon voor* alle *?*

Hier is het piramide-patroon voor *minstens drie*:

minstens drie A zijn B



Opdracht 7.12

1. *Geef het piramide-patroon voor* hoogstens drie.
2. *Hoe verhoudt dit patroon zich tot dat van* minstens drie*?*
3. *Geef ook het piramide patroon voor* precies drie*?*
4. *Hoe verhoudt dit patroon zich tot die van* hoogstens drie *en* minstens drie*?*
5. *Verklaar het gevonden verband.*

Opdracht 7.13

1. *Geef het piramide-patroon voor* minstens twee en hoogstens vijf.
2. *Geef het piramide-patroon voor* precies twee of minstens vier.
3. *Geef het piramide-patroon voor* precies twee of precies vijf.
4. *Tenslotte dat voor* minstens de helft.

In het gegeneraliseerde kwantor-perspectief zien kwantor-interpretaties eruit als relaties. We kunnen kwantoren dus gaan klassificeren met behulp van relatie-eigenschappen (vergelijk § 2.7). Enkele voorbeelden. De determinator *alle* is reflexief. Immers, de volgende beweringen zijn altijd waar: *alle wandelaars wandelen*, *alle jongens zijn jongens*, *alle meisjes zijn meisjes*, enzovoort. Dus in het algemeen: als $\mathcal{Q}$ geïnterpreteerd wordt als *alle* (dat wil zeggen: als de inclusie-relatie), dan hebben we:

$$\forall X \mathcal{Q} X X.$$

Dit drukt de reflexiviteits-eigenschap van $\mathcal{Q}$ uit.

De determinator *alle* is transitief. Immers, uit *alle jongens voetballen* en *alle voetballers hollen* volgt: *alle jongens hollen*. Uit *alle meisjes huilen* en *alle huilenden treuren* volgt: *alle meisjes treuren*, enzovoort. In het algemeen: als $\mathcal{Q}$ geïnterpreteerd wordt als *alle*, dan hebben we:

$$\forall X \forall Y \forall Z((\mathcal{Q} X Y \wedge \mathcal{Q} Y Z) \rightarrow \mathcal{Q} X Z).$$

Dit drukt de transitiviteits-eigenschap van $\mathcal{Q}$ uit.

De determinator *alle* is antisymmetrisch. Uit *alle jongens voetballen* en *alle voetballers zijn jongens* volgt dat de verzameling voetballers identiek is aan de verzameling jongens. Algemeen: als Q geïnterpreteerd wordt als *alle*, dan hebben we:

$$\forall X \forall Y((\mathcal{Q} X Y \wedge \mathcal{Q} Y X) \rightarrow X = Y).$$

Deze formule drukt uit dat de relatie $\mathcal{Q}$ anti-symmetrisch is.

Opdracht 7.14

1. *Is de determinator* geen *reflexief?*
2. *Irreflexief?*
3. *Symmetrisch?*
4. *Antisymmetrisch?*
5. *Transitief?*

Opdracht 7.15 *Dezelfde vragen voor de determinator* precies twee.

Opdracht 7.16

1. *Zij gegeven dat de determinator $\mathcal{Q}$ reflexief is. Wat kunt u hieruit opmaken voor het piramide-patroon van deze determinator?*

2. *Laat zien dat elke determinator met het gevonden piramide-patroon reflexief is.*

Wanneer u de laatste opdracht heeft begrepen en uitgevoerd kunt u voor alle determinatoren waarvoor u een piramide-patroon heeft getekend in één oogopslag zien of ze reflexief zijn of niet.

Opdracht 7.17 *Zij gegeven dat het piramide-patroon van de determinator $\mathcal{Q}$ aan de rechter-zijkant louter minnen vertoont. Welke relationele eigenschap voor de interpretatie van $\mathcal{Q}$ kunt u hieruit afleiden?*

Wanneer bovenstaande opdrachten u nieuwsgierig hebben gemaakt, is er voor u werk aan de winkel. Vele fraaie vraagstukken in dit genre (zij het soms wat ingewikkelder) liggen nog op u te wachten in de gegeneraliseerde kwantoren theorie.

De theorie der gegeneraliseerde kwantoren speelt impliciet een belangrijke rol in de semantiek van de Montague grammatica. Taalkundig is de theorie van belang omdat zij ons in staat stelt allerlei taalkundige kwantificatieverschijnselen, bij voorbeeld in gekwantificeerde lidwoordgroepen als *sommige*, *de meeste*, *vele*, ..., maar ook in temporele adverbia als *soms*, *meestal*, *vaak*, ..., logisch onder één hoedje te vangen.

De theorie heeft door het verschijnen van [Barwise & Cooper 1981] een nieuwe impuls gekregen. Aan de hoge vlucht die de theorie daarna heeft genomen hebben Nederlandse logici en semantici een belangrijke bijdrage geleverd. Zie voor meer informatie: [Van Benthem & Ter Meulen 1985], en de daar genoemde literatuur. Taalkundige toepassingen zijn te vinden in [Zwarts 1981, Zwarts 1986]. Om het taalkundige belang van gegeneraliseerde kwantorentheorie te illustreren definiëren we een nieuwe eigenschap van kwantoren.

Definitie 7.4 Opwaartse monotonie rechts van een kwantor

Voor alle X, Y, Y': als $\mathcal{Q}XY$ en $Y \subseteq Y'$ dan $\mathcal{Q}XY'$.

Er bestaat een tegenhanger voor opwaartse monotonie:

Definitie 7.5 Neerwaartse monotonie rechts van een kwantor

Voor alle X, Y, Y': als $\mathcal{Q}XY$ en $Y' \subseteq Y$ dan $\mathcal{Q}XY'$.

Voorbeelden van kwantoren die opwaarts monotoon zijn (of kortweg stijgend) in hun rechterargument zijn *alle* en *minstens één*. Dat dit zo is blijkt uit het feit dat de volgende redeneringen geldig zijn:

Alle jongens zijn aardig.

Alle jongens zijn aardig of verlegen.

Minstens één meisje is aardig.

Minstens één meisje is mooi of aardig.

Voorbeelden van kwantoren die neerwaarts monotoon zijn (of kortweg dalend) in hun rechterargument zijn *geen* en *weinig*.

Opdracht 7.18 *Geef voorbeelden waaruit dit blijkt.*

De monotonie eigenschappen van kwantoren zijn zeer nuttig voor het beschrijven van zogenaamde *polariteitsverschijnselen* in natuurlijke taal. Het Nederlandse werkwoord *hoeven* is een voorbeeld van een werkwoord dat alleen kan worden gebruikt in een 'negatieve context'. In taalkundig jargon: het is een woord met *negatieve polariteit*. Beschouw nu de volgende voorbeelden:

(20) *Hoogstens honderd strijders hoeven de wacht te houden.*

(21) **Minstens honderd strijders hoeven de wacht te houden.*

Ga na dat de kwantor *hoogstens honderd* neerwaarts monotoon is in zijn rechterargument. De kwantor *minstens honderd* is dat niet. De nominale constituenten die bij combinatie met een verbale constituent een woord met negatieve polariteit in die verbale constituent toelaten blijken precies de kwantoren te zijn die neerwaarts monotoon zijn in hun tweede argument. Het begrip *neerwaartse monotonie* levert een bruikbare precisering op van het veel vagere begrip 'negatieve context'.

Opdracht 7.19 *Beschouw de volgende voorbeelden:*

- *Alle mannen waren allerminst tevreden.*
- *Precies vijf mannen waren allerminst tevreden.*
- **Niemand was allerminst tevreden.*

Het Nederlandse woord allerminst *is een woord met* positieve polariteit. *Formuleer in termen van monotonie een principe waaraan het gebruik van dit woord moet voldoen.*

Opdracht 7.20 *Formuleer twee monotonie principes voor het linkerargument van een kwantor. Beschouw nu de volgende kwantoren:*

1. *alle*
2. *niet alle*
3. *minstens een*

4. *hoogstens honderd*

5. *precies vijf*

6. *geen.*

Ga na wat hun monotonie-eigenschappen zijn in het linkerargument.

7.4 Extensionele typenlogica

In 7.2 hebben we variabelen voor eerste orde predikaten (en kwantificatie over die variabelen) ingevoerd. Wanneer we ook nog constanten voor tweede orde predikaten toestaan kunnen we predikatie over eerste orde predikaten uitdrukken, en bij voorbeeld een vertaling geven voor "Liefde is blind". Namen voor tweede orde predikaten hebben we ook nodig in de vertaling van "Jan heeft alleen goede eigenschappen". Dit wordt:

$$\forall X(Xj \rightarrow \mathbf{G}X),$$

waarbij X een eerste orde predikaatvariabele is, en $\mathbf{G}$ een tweede orde predikaatconstante. "Ieder mens heeft wel een goede eigenschap" wordt nu:

$$\forall z(Mz \rightarrow \exists X(Xz \wedge \mathbf{G}X)).$$

Voortgaande op het pad dat we in § 7.2 zijn ingeslagen zouden we ook kunnen gaan kwantificeren over predikaten van predikaten. Zo ontstaat derde orde logica. Om systematische redenen is het nu aantrekkelijk om een *algemene* theorie te ontwikkelen die kwantificatie over steeds ingewikkelder objecten toestaat.

Zo'n theorie is gepresenteerd door Russell. Het is de zogenaamde typenlogica, waarin kwantificatie over alle eindige ordes is toegestaan. Russell had zijn theorie bedacht om de door hem ontdekte paradox in de verzamelingenleer (vergelijk § 3.5) te omzeilen, maar in dat verband is ze eigenlijk nooit populair geworden: het bleek dat je die paradox (en verwante paradoxen) ook al kwijt kon raken door het kiezen van geschikte axioma's in de taal van de eerste orde predikatenlogica (zie § 6.7).

De typenlogica is echter wel zeer geschikt voor het analyseren van natuurlijke taal. Het blijkt een zeer natuurlijke pendant te zijn van de zogenaamde *categoriale syntaxis*. We zullen hier iets vertellen over typenlogica in zijn allersimpelste vorm, waarin alle typen opgebouwd zijn uit de basistypen *objecten* en *waarheidswaarden*. Deze versie van de typenlogica heet *extensionele typenlogica*.

In de extensionele typenlogica is het mogelijk te verwijzen naar predikaten van willekeurige complexiteit (dat wil zeggen: van een willekeurige

eindige orde, of: van een willekeurig *type*). Dat dit voor de analyse van natuurlijke taal handig is, is duidelijk te zien in voorbeelden waarin de predikaten waarvan iets gezegd wordt zelf ook nog eens *interne structuur* vertonen:

(22) Niet roddelen *is verstandig.*

(23) Iemand bestelen *is strafbaar.*

(24) Opstaan voor iemand *misstaat niemand.*

(25) Zijn naasten liefhebben *is ieders plicht.*

In de typenlogica wordt de verwijzing naar een predikaat tot stand gebracht door middel van zogenaamde *labda-abstractie* (of: λ-abstractie). Laat een uitdrukking φ waarin een constante a voorkomt gegeven zijn. We noteren dit als $\varphi(a)$. Deze uitdrukking valt te beschouwen als een zin waarin een predikaat aan a wordt toegekend. Dat predikaat willen we er nu uit peuteren door middel van λ-abstractie. Daartoe vervangen we overal in $\varphi(a)$ de constante a door de variabele x. (We nemen aan dat x een variabele is die niet vrij in $\varphi(a)$ voorkomt.) Dit kunnen we schrijven als $[x/a]\varphi(a)$ of als $\varphi(x)$. Vervolgens markeren we de variabele x met een zogenaamde λ-operator. Het resultaat is: $\lambda x[\varphi(x)]$. De vierkante haken geven hier het bereik van de λ-operator aan. Neem even aan dat $\varphi(a)$ een *zin* was (een gesloten formule). Die zin drukt uit dat object a zekere eigenschap heeft. Nu staat $\lambda x[\varphi(x)]$ voor een karakteristieke functie, namelijk de *functie* die objecten d afbeeldt op 1 of op 0, al naar gelang ze al of niet de eigenschap hebben die $\varphi(a)$ aan a toeschreef. Enkele concrete voorbeelden kunnen dit duidelijker maken.

- Abstractie op a in Pa levert $\lambda x[Px]$ ('de eigenschap P hebben').
- Abstractie op a in Rab levert $\lambda x[Rxb]$ ('in relatie R staan tot b', 'b R-en').
- Abstractie op b in Rab levert $\lambda x[Rax]$ ('door a ge-R-d worden').
- Abstractie op a in Raa levert: $\lambda x[Rxx]$ ('tot zichzelf in relatie R staan'; 'zichzelf R-en').
- Abstractie op a in $\neg Pa$ levert: $\lambda x[\neg Px]$ ('de eigenschap niet-P hebben').

Met behulp van de λ-operator kunnen we nu de hierboven gegeven voorbeelden uit de natuurlijke taal gaan vertalen. We maken weer gebruik van hogere orde predikaatconstanten, waarvoor we vette letters zullen nemen. "Niet roddelen is verstandig" krijgt als vertaling: $\mathbf{V}(\lambda x[\neg Rx])$. De vertaling

van "Iemand bestelen is strafbaar" kan nu worden: $\mathbf{S}(\lambda x[\exists y Bxy])$. "Opstaan voor iemand misstaat niemand": $\neg\exists z\mathbf{M}(\lambda x[\exists y Oxy], z)$. Toelichting: **M** is een hogere orde predikaatconstante die als *eerste* argument een eerste orde predikaat neemt en als *tweede* argument een object. **M** staat dus voor: *eigenschap* zus hebben misstaat aan *object* zo. Tenslotte de vertaling voor "(Al) zijn naasten liefhebben is ieders plicht": $\forall z\mathbf{P}(\lambda x[\forall y(Nyx \rightarrow Lxy)], z)$. Toelichting: **P** staat voor: *eigenschap* zus hebben is plicht voor *object* zo.

Het bovenstaande was niet meer dan een eerste stap. Behalve λ-abstractie uit zinnen is ook abstractie uit predikaten mogelijk. Abstractie op a in $\lambda x[Rxa]$ levert $\lambda y[\lambda x[Rxy]]$, hetgeen uitdrukt: '$R$-en', ofwel: 'in relatie R staan'. We kunnen ook abstractie plegen op a in $\lambda x[Rax]$. Dit levert de uitdrukking $\lambda y[\lambda x[Ryx]]$, met een andere betekenis: 'ge-R-d worden'. Over dit verschil tussen de 'actieve' en de 'passieve' vorm van een relatie-uitdrukking volgt straks meer informatie.

Tot nu toe hebben we de λ-operator alleen toegepast op *individuele constanten*. We kunnen ook abstractie toepassen op predikaatconstanten. Abstractie op A in Ab (A is een predikaatconstante) levert op: $\lambda Y[Yb]$. In deze uitdrukking is Y een predikaatvariabele. De betekenis van de hele uitdrukking: 'eigenschap zijn van b'.

In de analyse van natuurlijke taal kunnen we nu dit alles mooi gebruiken. Gegeven dat we de bijdrage van de afzonderlijke woorden aan de betekenis van een zin als "Jan loopt snel" willen achterhalen, dan kan dat met behulp van vertaling in een typenlogische taal als volgt:

- "Jan loopt" wordt vertaald als Lj.

Hieruit volgt meteen:

- 'Jan' wordt vertaald als j, en 'loopt' als L, of met gebruik van λ-notatie, als $\lambda x[Lx]$.
- "Jan loopt snel" betekent zo ongeveer: "Jan heeft de eigenschap snel te lopen", waarbij *snel lopen* een eigenschap is die is afgeleid van *lopen*. De vertaling kan dus zoiets worden als $\mathcal{S}Lj$, waarbij $\mathcal{S}$ een uitdrukking is die van een eenplaatsig eerste orde predikaat een nieuw eenplaatsig eerste orde predikaat maakt.

Korte discussie tussendoor: deze vertaling lijkt misschien nodeloos ingewikkeld, want we zouden immers ook kunnen vertalen in $Lj \wedge Sj$, dat wil zeggen: Jan loopt en hij *is* snel. Dit is echter niet helemaal correct. Immers, stel dat Jan zich te voet met een snelheid van twaalf kilometer per uur voortspoedt en daarbij op gedragen toon een gedicht van Vondel reciteert. Nu is "Jan loopt snel" zeker waar; ook "Jan reciteert" is waar; maar: "Jan reciteert snel" is niet waar. Het voorbeeld geeft aan dat de parafrase van

de eerste zin zoiets zou moeten zijn als "Jan loopt, en dat lopen heeft de eigenschap snel te zijn (vergeleken bij andere loopactiviteiten)". Een andere mogelijkheid is: $Lj \wedge \mathbf{S}L$ (waarbij $\mathbf{S}$ een uitdrukking is die met een eenplaatsig eerste orde predikaat combineert tot een zin. Dit is echter ook niet helemaal juist. Immers, gegeven dit recept zou "Jan loopt snel, en Elias loopt niet snel" als vertaling hebben $Lj \wedge \mathbf{S}L \wedge Le \wedge \neg\mathbf{S}L$. Maar dit is in geen enkele 'redelijke' situatie waar, zodat deze vertaling niet correct kan zijn. De gekozen vertaling lost allebei deze problemen op. Hieruit volgt:

- 'loopt snel' moet als vertaling hebben: $\lambda x[\mathcal{S}Lx]$.

Maar als we dit hebben weten we ook hoe we 'snel' moeten vertalen:

- 'snel' heeft als vertaling: $\lambda Y[\lambda x[\mathcal{S}Yx]]$.

Dit voorbeeld was bedoeld om te illustreren dat de uitdrukkingskracht van een typen-theoretische taal uitstekend van pas komt wanneer het erom gaat de betekenisdragende bouwstenen van de natuurlijke taal elk aan een eigen vertaling te helpen.

Syntactisch gezien is het bijwoord 'snel' een zinsdeel dat samen met de verbale constituent 'loopt' een nieuwe verbale constituent 'loopt snel' kan vormen. De gang van zaken op het niveau van de typenlogische vertaling vertoont nu een volledige parallel: de vertaling van 'loopt', $\lambda y[Ly]$, een eerste orde predikaat, combineert met de vertaling van 'snel', $\lambda Y[\lambda x[\mathcal{S}Yx]]$, tot een *nieuw* eerste orde predikaat dat de vertaling vormt van 'loopt snel': $\lambda x[\mathcal{S}Lx]$.

Hoe kunnen we nu $\lambda Y[\lambda x[\mathcal{S}Yx]]$ met $\lambda y[Ly]$ combineren, zo dat het resultaat $\lambda x[\mathcal{S}Lx]$ is? Dat zit zo. De uitdrukking $\lambda Y[\lambda x[\mathcal{S}Yx]]$ staat voor een functie die eenplaatsige eerste orde eigenschappen als argument neemt, en nieuwe eenplaatsige eerste orde eigenschappen als waarde aflevert. Een eenplaatsige eerste orde eigenschap is in wezen niets anders dan een verzameling individuen (de verzameling lopers, de verzameling jongens, de verzameling snelle lopers, de verzameling snelle jongens, enzovoorts).

De uitdrukking $\lambda y[Ly]$ staat voor een eerste orde eigenschap. Net zoals we de eenplaatsige predikaatletter L kunnen combineren met de individuele constante j tot de formule Lj, kunnen we nu de volgende combinatie maken:

$$\lambda Y[\lambda x[\mathcal{S}Yx]](\lambda y[Ly]).$$

Deze uitdrukking staat voor: de *toepassing* van het tweede orde predikaat $\lambda Y[\lambda x[\mathcal{S}Yx]]$ op het argument $\lambda y[Ly]$.

De typenlogische uitdrukking $\lambda Y[\lambda x[\mathcal{S}Yx]](\lambda y[Ly])$ kan worden vereenvoudigd door middel van zogenaamde λ-*conversie*. Zij gegeven een λ-uitdrukking $\lambda v[\varphi(v)]$. Laat α een argument zijn voor die uitdrukking. α is

een argument voor $\lambda v[\varphi(v)]$ wanneer α en v van hetzelfde type zijn, dat wil zeggen ze staan allebei voor een object, allebei voor een eerste orde predikaat van een bepaalde plaatsigheid, allebei voor een tweede orde predikaat van een bepaalde plaatsigheid, enzovoort. Als α een argument is voor $\lambda v[\varphi(v)]$ mogen we $\lambda v[\varphi(v)](\alpha)$ converteren tot $[\alpha/v]\varphi$, het resultaat van het vervangen van de variabele v in φ door α, overal waar deze variabele voorkomt, terwijl het voorvoegsel λv wegvalt. Voor het gemak schrijven we dit als $\varphi(\alpha)$.

Om duidelijk te maken hoe λ-conversie precies in zijn werk gaat zijn voorbeelden weer erg nuttig:

- $\lambda x[Lx](j)$ wordt geconverteerd tot Lj.

Opmerkingen: aan dit voorbeeld kunt u zien dat L en $\lambda x[Lx]$ op hetzelfde neerkomen; verder mag j een argument zijn bij $\lambda x[Lx]$ omdat x en j van hetzelfde type zijn.

- $\lambda Y[Ya](P)$ wordt geconverteerd tot Pa.

Opmerking: Y en P zijn weer van hetzelfde type (beide letters staan voor eerste orde predikaten); als de twee letters uitdrukkingen van verschillende type zouden zijn geweest zou P geen argument kunnen zijn bij $\lambda Y[Ya]$.

- $\lambda Y[Ya \wedge Yb](P)$ wordt geconverteerd tot $Pa \wedge Pb$.
- $\lambda x[\neg Rxx](a)$ wordt geconverteerd tot $\neg Raa$.

Het argument bij een λ-uitdrukking kan ook een variabele zijn:

- $\lambda x[Px](y)$ wordt geconverteerd tot $\neg Py$.
- $\lambda x[\neg Px](x)$ wordt geconverteerd tot: $\neg Px$.

Moeilijkheden kunnen ontstaan wanneer een argument-variabele bij conversie gebonden dreigt te raken, door een kwantor of door een λ-operator. De argument variabele was dan niet vrij voor de variabele v waarvoor conversie plaatsvindt. Een voorbeeld hebben we in: $\lambda x[\lambda y[Rxy]](y)$. Conversie zou hier een uitdrukking opleveren waarin de argument-variabele y ten onrechte gebonden wordt, namelijk de uitdrukking $\lambda y[Ryy]$. De remedie is standaard: toepassen van λ-conversie op de uitdrukking $\lambda v[\varphi(v)](\alpha)$ is verboden als het argument α een variabele is die niet vrij is voor v in $\varphi(v)$ of variabelen *bevat* die niet vrij zijn voor v in $\varphi(v)$. In dit soort gevallen moet eerst worden overgestapt naar een alfabetische variant van de oorspronkelijke uitdrukking waarin dit probleem niet optreedt. Voorbeeld: $\lambda x[\lambda y[Rxy]](y)$ wordt eerst vervangen door de alfabetische variant $\lambda x[\lambda z[Rxz]](y)$; λ-conversie levert nu geen problemen op en geeft als resultaat $\lambda z[Ryz]$.

Het conversie-recept toepassen op $\lambda Y[\lambda x[\mathcal{S}Yx]](\lambda x[Lx])$ geeft—in twee conversie-stappen—het beoogde resultaat. De eerste conversie-stap levert op: $\lambda x[\mathcal{S}\lambda x[Lx](x)]$. In de volgende stap krijgen we via conversie van een deel-uitdrukking het uiteindelijke conversie-resultaat: $\lambda x[\mathcal{S}Lx]$. Dit is inderdaad de vertaling voor 'snel lopen' die we wilden hebben. Komt deze λ-goochelarij u nog wat vreemd voor? Niet getreurd, want oefening baart kunst:

Opdracht 7.21 *Vereenvoudig de volgende formules zoveel mogelijk, met behulp van λ-conversie:*

1. $\lambda Y[\lambda x[Yx]]\,(P)$
2. $\lambda Y[\lambda x[Yx]]\,(P)(j)$
3. $\lambda Y[\lambda Z[\exists x(Yx \wedge Zx)]]\,(P)$
4. $\lambda Y[\lambda Z[\exists x(Yx \wedge Zx)]]\,(Y)$
5. $\lambda Y[\lambda Z[\exists x(Yx \wedge Zx)]]\,(Z)$
6. $\lambda x[\lambda Y[Yx]]\,(j)$
7. $\lambda Y[Yj](\lambda x[Rax])$
8. $\lambda \mathbf{X}[\mathbf{X}(\lambda y[Py])]\,(\lambda Y[Ya])$
9. $\lambda \mathbf{Z}[\mathbf{Z}(\lambda y[Py \wedge Qy]) \vee \mathbf{Z}(\lambda x[Bx])]\,(\lambda X[Xb])$.

Opdracht 7.22 *Vertaal de volgende zinnen in de extensionele typenlogica:*

1. *Het is verboden zich op het gras te bevinden.*
   (Vertaalsleutel: g: het gras; Bxy: x bevindt zich op y; $\mathbf{V}(Y)$: Y is verboden.)
2. *Het in iets geloven geeft troost.*
   (Vertaalsleutel: Gxy: x gelooft in y; $\mathbf{T}(Y)$: Y geeft troost.)
3. *Niemand vertrouwen is ziekelijk.*
   (Vertaalsleutel: Vxy: x vertrouwt y; $\mathbf{Z}(Y)$: Y is ziekelijk.)
4. *Jezelf niet vertrouwen is ziekelijk.*
   (Vertaalsleutel als boven.)
5. *Niet iedereen vertrouwen is niet ziekelijk.*
   (Vertaalsleutel als boven.)

Opdracht 7.23 *Neem aan dat Gabc de vertaling is van "*A*driaan geeft de* B*akoenin-biografie aan* C*ornelis". Parafraseer nu de volgende uitdrukkingen in het Nederlands:*

1. $\lambda x[Gxbc]$
2. $\lambda x[Gaxc]$
3. $\lambda x[Gabx]$.

Opdracht 7.24 *Vertaal de volgende Nederlandse uitdrukkingen in uitdrukkingen van de typenlogica (zie vorige opdracht voor de vertaalsleutel):*

1. *Aan iemand de Bakoenin-biografie geven.*
2. *Van iemand de Bakoenin-biografie krijgen.*
3. *Iets aan Cornelis geven.*
4. *Iets van Adriaan krijgen.*
5. *Aan iedereen iets geven.*

Wie 'iets wil bereiken' in de semantiek van de natuurlijke taal doet er goed aan zich grondig vertrouwd te maken met de λ-rekenarij (ook wel 'λ-calculus' genoemd). Befaamde linguïsten gingen u reeds voor. Een gevleugeld woord van Barbara Partee, één van hen: "λ has changed my life".

Opdracht 7.25 *Vertaal in een typenlogische uitdrukking: "Iets willen bereiken is prijzenswaardig."*

7.5 Syntaxis en semantiek van de extensionele typenlogica

We zijn tot nu toe wat slordig geweest in het praten over de typen van de verschillende uitdrukkingen in de typenlogica. Let op: de *typen* waar we het nu over hebben zijn *functionele typen*; iets heel anders dan de *soort-typen* die in 7.1 ter sprake zijn geweest. De soort-typen uit 7.1 waren allemaal van het functionele type *entiteit* (dat wil zeggen: individueel object).

Praten over functionele typen kan heel precies. De verzameling van alle typen noemen we TYPE. Deze verzameling wordt als volgt recursief gedefinieerd:

Definitie 7.6 *Definitie van* TYPE.

- *e en t zijn elementen van* TYPE *(e staat voor 'entiteit', Engels: entity; t staat voor 'waarheidswaarde', Engels: truthvalue);*
- *als a en b elementen zijn van* TYPE*, dan is* $\langle a, b\rangle$ *ook een element van* TYPE*;*
- *niets anders is een element van* TYPE*.*

Uitdrukkingen van type $\langle a, b\rangle$, voor zekere typen a en b, staan voor functies: als je zo'n uitdrukking combineert met een uitdrukking van type a is het resultaat een uitdrukking van type b. Enkele voorbeelden:

- individuele variabelen en individuele constanten zijn van type e;
- alle formules uit de eerste orde predikatenlogica zijn van type t;
- eenplaatsige eerste orde predikaatconstanten en -variabelen zijn van type $\langle e, t\rangle$; als je ze combineert met een individuele variabele of constante als argument—dus met een argument van type e—leveren ze formules op: uitdrukkingen van type t;
- tweeplaatsige eerste orde predikaatconstanten en -variabelen zijn van type $\langle e, \langle e, t\rangle\rangle$; je kunt je voorstellen dat zo'n uitdrukking eerst met een type e uitdrukking combineert tot een eenplaatsig predikaat, dat van type $\langle e, t\rangle$ is (net zoals de transitieve werkwoordsvorm *slaat* combineert met *Jan* als lijdend voorwerp tot de verbale constituent *slaat Jan*, een constituent die dezelfde syntactische status heeft als de intransitieve werkwoordsvorm *slaapt*);
- drieplaatsige eerste orde predikaatconstanten en -variabelen zijn van type $\langle e, \langle e, \langle e, t\rangle\rangle\rangle$ (want met een individuele constante of variabele combineren ze tot een uitdrukking van type $\langle e, \langle e, t\rangle\rangle$, net zoals *geeft aan* combineert met *Jan* tot *geeft aan Jan*, met de status van een transitief werkwoord);
- eenplaatsige predikaatconstanten en -variabelen van de tweede orde hebben type $\langle\langle e, t\rangle, t\rangle$ (want ze combineren met een eenplaatsig eerste orde predikaat, een uitdrukking van type $\langle e, t\rangle$, tot een formule, een uitdrukking van type t);
- het connectief $\neg$ heeft type $\langle t, t\rangle$ (want het combineert met een formule tot een nieuwe formule);
- de connectieven $\vee$, $\wedge$, $\rightarrow$, en $\leftrightarrow$ zouden type $\langle t, \langle t, t\rangle\rangle$ kunnen krijgen (u gelieve zelf te bedenken waarom).

De volgende definitie legt de typen vast van uitdrukkingen die worden gevormd met behulp van λ-abstractie.

Definitie 7.7 Labda abstractie

Als β een uitdrukking is van type b, en v is een variabele van type a, dan is $\lambda v[\beta]$ een uitdrukking van type $\langle a, b\rangle$.

We geven weer een aantal voorbeelden:

- $\lambda x[Lx]$ is van type $\langle e, t\rangle$, want Lx is een formule (heeft dus type t), en x een individuele variabele, die type e heeft. Dit klopt met wat we boven gezien hebben: $\lambda x[Lx]$ is een andere schrijfwijze voor L, een eenplaatsige eerste orde predikaatletter.
- $\lambda Y[Ya]$ is van type $\langle\langle e, t\rangle, t\rangle$, want Y is een predikaatvariabele van type $\langle e, t\rangle$, en Ya is van type t.
- $\lambda\mathbf{X}[\mathbf{X}(P)]$ is van type $\langle\langle\langle e, t\rangle, t\rangle, t\rangle$, want $\mathbf{X}$ is een variabele van type $\langle\langle e, t\rangle, t\rangle$, en $\mathbf{X}(P)$ is een uitdrukking van type t.

In termen van de type-indeling die we zojuist hebben gedefinieerd kunnen we nu ook precies vastleggen wanneer een combinatie van een functor (dat wil zeggen: een naam voor een functie) met een argument welgevormd is. Daartoe moeten de typen van de functor en het argument bij elkaar passen: de functor moet geïnterpreteerd worden als een functie die gedefinieerd is voor de interpretatie van de argument-uitdrukking. Om dat te bewerkstelligen dient de volgende regel voor functor-argument combinatie:

Definitie 7.8 Functor-argument combinatie

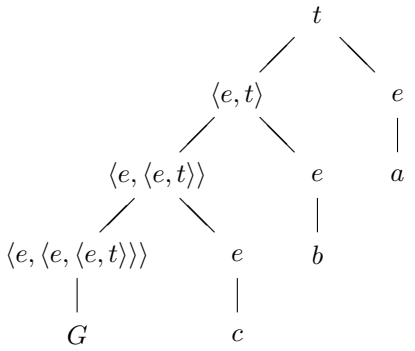
Als β een welgevormde typenlogische uitdrukking is van type $\langle a, b\rangle$, en α is een welgevormde typenlogische uitdrukking van type a, dan is $\beta(\alpha)$ een welgevormde typenlogische uitdrukking van type b.

Volgens deze definitie zijn La en Ya, of zo u wilt $L(a)$ en $Y(a)$, welgevormde uitdrukkingen van type t (want: L en Y hebben type $\langle e, t\rangle$, en a heeft type e), net als $\mathbf{X}(P)$ (want: $\mathbf{X}$ heeft type $\langle\langle e, t\rangle, t\rangle$ en P heeft type $\langle e, t\rangle$).

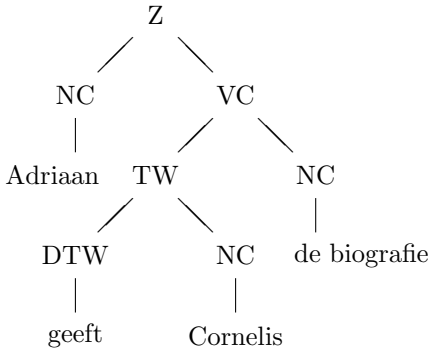
We hebben hierboven gezien dat drieplaatsige eerste orde predikaatletters in de extensionele typenlogica behandeld worden als uitdrukkingen van type $\langle e, \langle e, \langle e, t\rangle\rangle\rangle$. Strikt genomen moeten we nu "*A*driaan geeft de *B*akoenin-biografie aan *C*ornelis" (vergelijk opdrachten 7.23 en 7.24) vertalen als: $G(c)(b)(a)$. Immers, de predikaatletter G combineert eerst met een e-type argument c tot een uitdrukking $G(c)$ van type $\langle\langle e, t\rangle, t\rangle$ (een tweeplaatsige predikaatuitdrukking). De uitdrukking $G(c)$ combineert vervolgens met een e-type argument b tot een uitdrukking $G(c)(b)$ van type

$\langle e, t\rangle$. Tenslotte combineert $G(c)(b)$ met een e-type argument a tot een formule (type t uitdrukking) $G(c)(b)(a)$.

In boom-vorm ziet de structuur van $G(c)(b)(a)$ er nu zo uit (de knopen zijn gemarkeerd met de juiste type-aanduidingen):



Vergelijk dit met:



Hier staat TW voor 'transitief werkwoord' (Engels: TV, transitive verb), en DTW voor 'ditransitief werkwoord' (Engels: DTV, ditransitive verb). Zoals u ziet gaan we ervan uit dat een ditransitief werkwoord met een meewerkend voorwerp combineert tot een transitief werkwoord. Dit klopt ook, want vervanging van de combinatie *geeft Cornelis* door het transitieve werkwoord *leest* levert weer een syntactisch correcte zin op. Merk op dat de typenlogische boom en de syntactische boom, afgezien van de links-rechts volgorde van de onderwerps-NC en de VC, structureel identiek zijn.

Hoewel uit de notatie $G(c)(b)(a)$ de juiste boomstructuur direct valt af te lezen, is het soms lastig dat de volgorde van de argumenten precies omgekeerd is ten opzichte van de volgorde die we in een eerste orde predikatenlogische formule zouden hebben. De remedie is eenvoudig. We voeren een nieuwe afkorting in. We spreken af dat we desgewenst $G(c)(b)(a)$ mogen

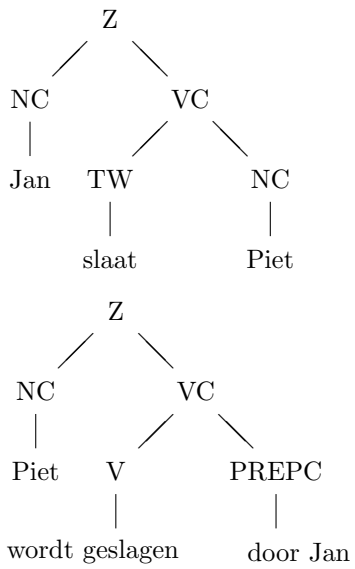
schrijven als $G(a, b, c)$ of als $Gabc$. Net zo voor tweeplaatsige predikaten: $R(b)(a)$ kan worden afgekort als Rab.

In feite hebben we deze manier van afkorten hierboven al gebruikt: atomaire predikatenlogische formules werden zonder haakjes in de typenlogica opgenomen. Ook de vertaling van "Opstaan voor iemand misstaat niemand" die we in § 7.4 hebben gegeven, $\neg\exists z\mathbf{M}(\lambda x[\exists yOxy], z)$, is strikt genomen een afkorting voor: $\neg\exists z\mathbf{M}(z)(\lambda x[\exists yO(y)(x)])$. Merk op: $\lambda x[\exists yOxy]$ is de vertaling van het *onderwerp* van de zin "Opstaan voor iemand misstaat niemand". De weggelaten haakjes kwamen verder, met name bij herhaalde λ-conversie, weer terug om duidelijk te maken wat de nieuwe functie-applicaties waren.

Opdracht 7.26 *Geef de typen van de constanten die we in § 7.4 hebben gebruikt voor de vertaling van* verstandig zijn*,* strafbaar zijn*,* misstaan*, en* plicht zijn*.*

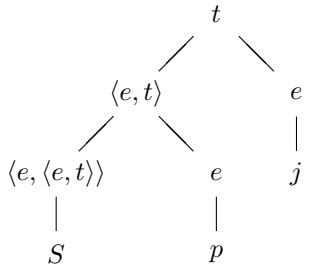
Opdracht 7.27 *Geef het type van de vertaling van* snel *in "Jan loopt snel" (zie de bespreking op bladzijde 212 en volgende).*

We kunnen nu ook laten zien hoe λ-abstractie gebruikt kan worden om argument-plaatsen te verwisselen. In feite is het onderscheid tussen de actieve en passieve vorm van een Nederlandse zin niets anders dan het omdraaien van twee argumenten. Vergelijk de structuurbomen voor "Jan slaat Piet" en "Piet wordt geslagen door Jan":



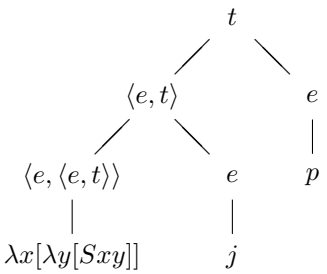
Hier staat V voor *verbum* en PREPC voor *prepositie-constituent* (Engels: PP, prepositional phrase).

Neem aan dat het transitieve werkwoord *slaan* in de typenlogica een vertaling S heeft, van type $\langle e, \langle e, t\rangle\rangle$. De vertaling van "Jan slaat Piet" wordt dan: $S(p)(j)$, met de volgende structuurboom:



Stel nu dat we uit $S(p)(j)$—of, zo u wilt, afgekort: Sjp—door middel van dubbele λ-abstractie een nieuwe predikaatuitdrukking van type $\langle e, \langle e, t\rangle\rangle$ zouden vormen, als volgt. Eerst passen we λ-abstractie toe op het lijdend voorwerp. Dit geeft: $\lambda y[S(y)(j)]$, een uitdrukking van type $\langle e, t\rangle$. We kunnen dit uiteraard ook afgekort opschrijven: $\lambda y[Sjy]$. De uitdrukking betekent *door Jan geslagen worden*. Vervolgens passen we λ-abstractie toe op het onderwerp van de oorspronkelijke zin. Dit geeft $\lambda x[\lambda y[S(y)(x)]]$, afgekort $\lambda x[\lambda y[Sxy]]$. Dit is een uitdrukking van type $\langle e, \langle e, t\rangle\rangle$ (ga dit na), met als betekenis *geslagen worden*.

Met behulp van de uitdrukking die we zojuist hebben gefabriceerd kunnen we nu de passieve zin "Piet wordt geslagen door Jan" vertalen. De vertaling wordt: $\lambda x[\lambda y[Sxy]]\,(j)(p)$. Deze vertaling heeft een structuurboom die correspondeert met de syntactische structuurboom voor "Piet wordt geslagen door Jan":



De typenlogische verantwoording van het feit dat de twee zinnen "Jan slaat Piet" en "Piet wordt geslagen door Jan" hetzelfde betekenen is nu simpel. De vertaling van "Piet wordt geslagen door Jan" kan als volgt worden vereenvoudigd met behulp van λ-conversie: $\lambda x[\lambda y[Sxy]]\,(j)(p)$ wordt geconverteerd tot $\lambda y[Sjy](p)$, en dit wordt weer geconverteerd tot Sjp. De vertaling is dus hetzelfde als die van "Jan slaat Piet".

Opdracht 7.28 *Zij gegeven dat "*A*driaan geeft de* B*akoenin-biografie aan* C*ornelis" de vertaling* $G(c)(b)(a)$ *heeft, of afgekort: Gabc. Vertaal nu in uitdrukkingen van de extensionele typenlogica, en geef van elke typenlogische uitdrukking het type aan:*

1. *de Bakoenin-biografie geven aan Cornelis*
2. *geven aan Cornelis*
3. *de Bakoenin-biografie krijgen van Cornelis*
4. *krijgen van Cornelis*
5. *krijgen van iemand*
6. *aan Cornelis iets geven*
7. *van iedereen iets krijgen.*

Opdracht 7.29 *Het gegeven is als in de vorige opdracht. Geef van elk van de volgende uitdrukkingen het type aan, en geef een parafrase in het Nederlands:*

1. $\lambda x[\lambda y[Gybx]]$
2. $\lambda x[\lambda y[Gxby]]$
3. $\lambda x[\lambda y[\exists z Gyzx]]$
4. $\lambda x[\lambda y[\exists z Gxzy]]$.

Wie een taalkundige toepassing wil zien van deze λ-trucs leze [Dowty 1982].

Het wordt hoog tijd dat we een enkel woord wijden aan de *semantiek* van de extensionele typenlogica. Die semantiek is ingewikkelder dan die van de eerste orde predikatenlogica, omdat we voor elk type a een domein D_a nodig hebben van objecten van type a. Dit lijkt erger dan het is. We hoeven al die domeinen niet expliciet in te voeren, maar we kunnen ze recursief definiëren, uitgaande van de domeinen voor de basistypen e en t. Het domein D_t voor het basistype t is geen probleem: dat bestaat gewoon uit de twee waarheidswaarden ONWAAR en WAAR, en we kunnen er de verzameling $\{0, 1\}$ voor gebruiken. Het enige domein dat expliciet gegeven moet zijn is het domein D_e van de individuele objecten.

We frissen even ons geheugen op wat betreft de verzamelingstheoretische notatie. Als X en Y verzamelingen zijn, dan gebruiken we X^Y voor de verzameling van alle functies f met domein Y en co-domein X. De notatie $f : Y \to X$ stond voor: f is een functie met domein Y en co-domein X.

Dat X het co-domein is van f wil niet zeggen dat elk element van X ook als beeld onder f hoeft op te treden.

De recursieve definitie van de domeinen van de verschillende mogelijke typen gaat nu als volgt:

Definitie 7.9 Domeinen van de typen *in een typenlogische taal met basistypen e en t.*

- $D_t = \{0, 1\}$*;*
- D_e *moet expliciet worden gegeven;*
- *Als de domeinen van de typen a en b respectievelijk* D_a *en* D_b *zijn, dan is het domein voor het type* $\langle a, b\rangle$ *de volgende verzameling:*

  $$D_b^{D_a}$$

  Met andere woorden: de objecten van type $\langle a, b\rangle$ *zijn de functies van* D_a *naar* D_b*.*
- *Niets anders is een domein voor een type.*

We kunnen nu een *model* voor een extensionele typenlogische taal definiëren als een paar $\langle D_e, I\rangle$, waarbij D_e een verzameling individuen is, en I een interpretatie-functie die, voor elk type a, aan alle constanten van type a een element toekent uit het domein D_a dat geconstrueerd is zoals hierboven is aangegeven.

Als u vindt dat dit allemaal nogal abstract klinkt: u staat waarschijnlijk niet alleen, en het wordt u ook niet kwalijk genomen. Verder is het—zoals zoveel dingen in het leven—vooral een kwestie van wennen. Om dat gewenningsproces te stimuleren kijken we nog even naar de domeinen van een paar simpele typen.

Om de gedachten te bepalen: neem aan dat D_e slechts drie elementen heeft: $\{\star, \circ, \bullet\}$. Verder weten we: $D_t = \{0, 1\}$. Hoe ziet $D_{\langle e,t\rangle}$ (dat wil zeggen: het domein van de objecten van type $\langle e, t\rangle$, de eenplaatsige eerste orde predikaten) er nu uit? Volgens de definitie van de domeinen hebben we:

$$D_{\langle e,t\rangle} = D_t^{D_e} = \{0, 1\}^{\{\star,\circ,\bullet\}}$$

Dus: $D_{\langle e,t\rangle}$ is de verzameling *karakteristieke functies* van de verzameling $\{\star, \circ, \bullet\}$. Elk van die functies karakteriseert een deelverzameling van D_e, dus we kunnen zeggen dat $D_{\langle e,t\rangle}$ neerkomt op de verzameling van alle deelverzamelingen van het domein D_e. Een interpretatiefunctie I voor een typenlogische taal kent aan elke constante van type $\langle e, t\rangle$ een element van

$D_{\langle e,t\rangle}$ toe. Eenplaatsige eerste orde predikaatconstanten hebben type $\langle e,t\rangle$. Laat L zo'n predikaatconstante zijn, en veronderstel dat de interpretatiefunctie I aan L de volgende karakteristieke functie toekent:

$$\begin{array}{lcl} \star & \longrightarrow & 0 \\ \circ & \longrightarrow & 1 \\ \bullet & \longrightarrow & 1 \end{array}$$

Deze karakteristieke functie is een element van $D_{\langle e,t\rangle}$, en ze karakteriseert de deelverzameling $\{\circ,\bullet\}$ van D_e. Vergelijk dit met de interpretatiefunctie van een eerste orde model, dat aan eenplaatsige predikaatletters deelverzamelingen van het individuendomein toekent: hier zou L meteen worden geïnterpreteerd als de deelverzameling $\{\circ,\bullet\}$ van het domein. Beide manieren van interpreteren komen natuurlijk op hetzelfde neer.

Nog een voorbeeld. Hoe ziet $D_{\langle e,\langle e,t\rangle\rangle}$ eruit? We gaan weer te werk volgens het boekje (dat wil zeggen: volgens de definitie van de typendomeinen):

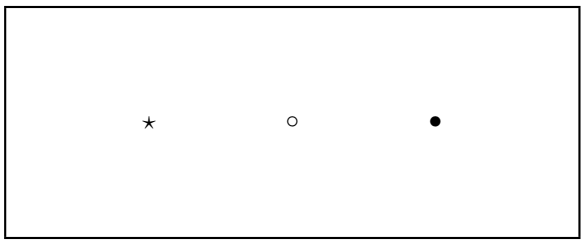
$$D_{\langle e,\langle e,t\rangle\rangle} = (D_{\langle e,t\rangle})^{D_e} = (\{0,1\}^{\{\star,\circ,\bullet\}})^{\{\star,\circ,\bullet\}}$$

De interpretatiefunctie I zou nu aan een constante R van type $\langle e,\langle e,t\rangle\rangle$ de volgende interpretatie kunnen toekennen:

$$\begin{array}{lcl} \star & \longrightarrow & \left(\begin{array}{lcl} \star & \longrightarrow & 0 \\ \circ & \longrightarrow & 1 \\ \bullet & \longrightarrow & 1 \end{array}\right) \\ \circ & \longrightarrow & \left(\begin{array}{lcl} \star & \longrightarrow & 0 \\ \circ & \longrightarrow & 1 \\ \bullet & \longrightarrow & 0 \end{array}\right) \\ \bullet & \longrightarrow & \left(\begin{array}{lcl} \star & \longrightarrow & 1 \\ \circ & \longrightarrow & 1 \\ \bullet & \longrightarrow & 1 \end{array}\right) \end{array}$$

U gelieve zelf na te gaan dat dit inderdaad een plaatje is van een element van $D_{\langle e,\langle e,t\rangle\rangle}$.

Opdracht 7.30 *In een eerste orde model zou R gewoon worden geïnterpreteerd als een tweeplaatsige relatie op $\{\star,\circ,\bullet\}$. Teken die relatie met behulp van pijlen in de volgende figuur:*



Als laatste voorbeeld beschouwen we het domein $D_{\langle\langle e,t\rangle,t\rangle}$. Weer passen we de definitie toe:

$$D_{\langle\langle e,t\rangle,t\rangle} = D_t^{D_{\langle e,t\rangle}} = D_t^{(D_t^{D_e})} = \{0,1\}^{(\{0,1\}^{\{\star,\circ,\bullet\}})}$$

Tekenen van een element van dit domein wordt ons teveel werk, maar we willen wel even met u over zo'n element *praten*. Neem aan dat s een constante is van type e zodanig dat $I(s) = \star$. Dan is $\lambda Y[Y(s)]$ een uitdrukking die moet worden geïnterpreteerd als de karakteristieke functie in $D_{\langle\langle e,t\rangle,t\rangle}$ die elke karakteristieke functie in $D_{\langle e,t\rangle}$ die $\star$ afbeeldt op 1, afbeeldt op 1, en elke andere karakteristieke functie in $D_{\langle e,t\rangle}$ op 0. Moet u hier even op mediteren? Ga gerust uw gang: de meeste stervelingen hebben wel wat tijd nodig voordat ze de Typentheoretische Verlichting bereiken. Anders gezegd: $\lambda Y[Y(s)]$ wordt geïnterpreteerd als (de karakteristieke functie van) de verzameling van alle eerste orde eigenschappen van $\star$. Een eerste orde eigenschap van $\star$ is niets anders dan (de karakteristieke functie van) een deelverzameling van $\{\star, \circ, \bullet\}$ waar $\star$ element van is.

De bedoeling van het bovenstaande verhaal was: u een indruk geven van de semantiek van de extensionele typenlogica. Het verhaal kan (en moet) uiteraard veel systematischer en uitvoeriger worden verteld. Zie voor meer informatie: [Gamut 1982], deel 2. Tenslotte dient nog vermeld dat de λ-calculus ook ten grondslag ligt aan de programmeertaal LISP, die vooral populair is bij onderzoekers op het terrein van de *kunstmatige intelligentie* (Engels: *artificial intelligence*), waar geprobeerd wordt computers 'intelligent gedrag' bij te brengen. Voor informatie over LISP zij verwezen naar [Steels 1983].

7.6 Modale predikatenlogica

Modale predikatenlogica (gangbare afkorting: MPL) ontstaat door $\Box$ en $\Diamond$ toe te voegen aan de verzameling operatoren van de gewone eerste orde predikatenlogica. Ze verhoudt zich dus tot de eerste orde predikatenlogica zoals de modale propositielogica (de gangbare afkorting daarvoor is MpL)

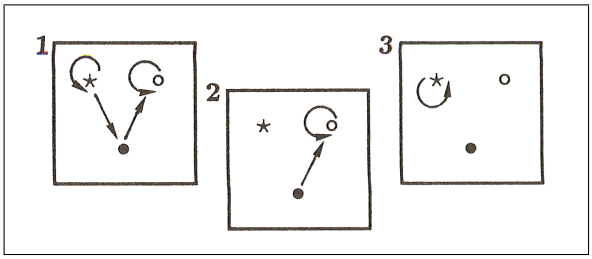
zich verhoudt tot de gewone propositielogica. $\Box$ en $\Diamond$ zijn eenplaatsige operatoren op formules, net als het negatieteken.

Opdracht 7.31 *Geef een recursieve definitie van de verzameling welgevormde formules van de modale predikatenlogica.*

$\Box\varphi$ betekent: φ is noodzakelijk waar. We leggen dit weer uit als: φ is waar *in alle mogelijke werelden*. $\Diamond\varphi$ staat voor: φ is mogelijkerwijs waar, of: het is mogelijk dat φ. De uitleg: φ is waar *in minstens één mogelijke wereld*.

Een *model* voor modale predikatenlogica bestaat nu uit een verzameling *contexten*, *indices*, of *mogelijke werelden* (in dit verband allemaal namen voor hetzelfde), laten we zeggen W, zo dat met elk element $w \in W$ een predikatenlogisch model oude-stijl is geassocieerd. Wanneer we aannemen dat elke wereld hetzelfde individuendomein heeft kunnen we zo'n model definiëren als een drietal $\langle D, W, I\rangle$, waarbij D een individuendomein is, W een verzameling mogelijke werelden, en I een interpretatiefunctie die nu iets ingewikkelder is dan in gewone eerste orde predikatenlogische modellen. I kent aan alle predikaatletters *per wereld* een interpretatie toe. Dus: $I_w(P)$ hoeft niet hetzelfde te zijn als $I_{w'}(P)$ (aangenomen dat w en w' twee verschillende elementen van W zijn).

Aanschouwelijk onderricht werkt weer het beste, dus een plaatje van een zeer eenvoudig model voor een modale predikatenlogische taal is hier op zijn plaats. Het model dat hier is afgebeeld geeft een beeld van de situatie rond drie objecten in drie verschillende mogelijke werelden:



Neem aan dat de getekende pijltjes de interpretatie vormen voor een tweeplaatsige predikaatletter R. Merk op dat de interpretatie *per wereld* kan worden omschreven als een verzameling geordende paren uit het domein (een tweeplaatsige relatie op $D = \{\star, \circ, \bullet\}$).

We kunnen nu formules uit een modale predikatenlogische taal met R als predikaatletter gaan *evalueren* (op waarheid of onwaarheid onderzoeken) in een mogelijke wereld in het hier getekende model. Bij voorbeeld: is $\exists x Rxx$ waar in wereld 3 van het model? Antwoord: ja, want $\star$ staat in die wereld in de pijlrelatie tot zichzelf. Is $\exists x Rxx$ waar in wereld 1? Ja, kijk maar. Nu met

modale operatoren. Is $\Box\exists xRxx$ waar in wereld 1? Ja, want in alle mogelijke werelden (dat wil in dit model zeggen: in wereld 1, wereld 2 en wereld 3) is $\exists xRxx$ waar. Is $\exists x\Box Rxx$ waar in wereld 1? Nee, want er is geen enkel object dat zowel in wereld 1, in wereld 2 en in wereld 3 in de pijlrelatie tot zichzelf staat.

Het model laat zien dat $\Box\exists xRxx$ en $\exists x\Box Rxx$ *niet* equivalent zijn. Intuïtief klopt dat ook wel, want vergelijk het volgende huis, tuin en keuken voorbeeld: "Iemand moet dit spelletje gaan winnen". Dit kan twee dingen betekenen, en elk van die lezingen vraagt om een andere vertaling in de modale predikatenlogica. De twee formules zijn: $\exists x\Box Sx$ en $\Box\exists xSx$ (met S voor 'gaat dit Spelletje winnen').

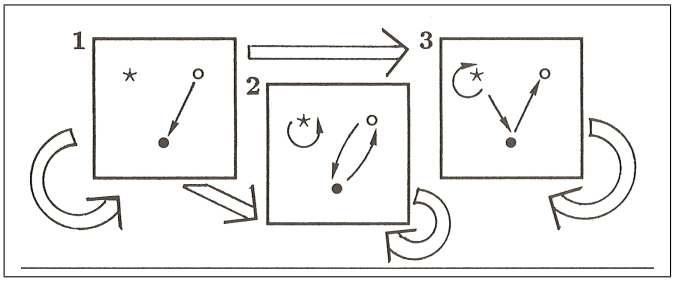
Opdracht 7.32 *Geef een ondubbelzinnige parafrase in het Nederlands van elk van deze twee mogelijke lezingen.*

Opdracht 7.33 *Welke twee lezingen heeft "Iedereen kan dit spelletje verliezen"? Geef voor elk van een beide lezingen een vertaling in de modale predikatenlogica.*

Opdracht 7.34 *Kijk weer naar het hierboven getekende model voor een modale predikatenlogische taal met tweeplaatsige predikaatletter R. Welke van de volgende formules zijn waar in wereld 1 van dit model:*

1. $\Box\forall x\exists yRxy$
2. $\Box\exists x\forall yRxy$
3. $\exists x\Box\forall yRxy$
4. $\forall x\Box\exists yRxy$.

De semantiek voor modale logica wordt interessanter wanneer je ook nog *structuur* aanbrengt in de verhoudingen tussen de verschillende mogelijke werelden. Neem aan dat werelden al dan niet *toegankelijk* kunnen zijn vanuit bepaalde andere werelden. Daartoe voeren we een tweeplaatsige relatie T in op de verzameling werelden W. Een model voor modale predikatenlogica krijgt nu de vorm $\langle D, W, T, I\rangle$. Een plaatje van zo'n model:



De pijl $\Longrightarrow$ staat voor de toegankelijkheidsrelatie T. Zoals u ziet is elke wereld in dit model toegankelijk vanuit zichzelf. Met andere woorden: in dit model is de toegankelijkheidsrelatie *reflexief*. Verder is wereld 2 toegankelijk vanuit wereld 1, maar niet andersom, en wereld 3 is toegankelijk vanuit 1. De waarheidsclausule voor $\Box$ en $\Diamond$ wordt nu:

- $\Box\varphi$ is waar in wereld w desda φ waar is in alle werelden die toegankelijk zijn vanuit w (dus: alle werelden waar vanuit w een pijl $\Longrightarrow$ naar wijst).
- $\Diamond\varphi$ is waar in wereld w desda φ waar is in minstens één wereld die toegankelijk is vanuit w.

Opdracht 7.35 *Beschouw het model dat hierboven gegeven is.*

1. *Is $\Box\exists xRxx$ waar in wereld 1 van dit model?*
2. *En in wereld 2?*
3. *Is $\Box\Box\exists xRxx$ waar in wereld 3?*

Het aardige aan het invoeren van de toegankelijkheidsrelatie T is nu dat we de betekenis van 'noodzakelijk' en 'mogelijk' nader kunnen specificeren door bepaalde *eisen* op te leggen aan de toegankelijkheidsrelatie. Als we bij voorbeeld afspreken dat in alle modellen die we willen beschouwen de toegankelijkheidsrelatie T *reflexief* moet zijn (dat wil zeggen dat in al die modellen geldt dat alle mogelijke werelden er toegankelijk zijn vanuit zichzelf), dan betekent dat dat elke formule van de volgende vorm universeel geldig wordt (dat wil zeggen: waar in elke wereld van elk model):

$$\varphi \rightarrow \Diamond\varphi.$$

Merk op dat dit niet een formule is maar een formule-schema: afhankelijk van wat we voor φ invullen krijgen we telkens een andere formule. Dat het schema geldig is valt als volgt in te zien. Voor elk model en voor elke mogelijke wereld geldt: als φ waar is in w, dan is er een mogelijke wereld die toegankelijk is vanuit w waarin φ waar is, namelijk w zelf, dankzij de reflexiviteit van de toegankelijkheidsrelatie in het model. Dus $\Diamond\varphi$ is ook waar in w.

Opdracht 7.36 *Laat zien: als we de toegankelijkheidsrelatie T* symmetrisch *maken, dan geldt het schema $\varphi \rightarrow \Box\Diamond\varphi$.*

Opdracht 7.37 *Welk schema geldt er als T* transitief *is?*

Over mogelijke eisen op de toegankelijkheidsrelatie valt nog veel meer interessants te zeggen. Wie meer wil weten over de semantiek voor MPL zij verwezen naar [Hughes & Cresswell 1968], en naar [Gamut 1982], deel 2. Beide boeken geven ook allerlei (taal)filosofische achtergronden.

Behalve de semantische kijk op modale propositie- en predikaten-logica is er weer de axiomatische: een gangbaar axioma voor MpL en MPL is

$$\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi).$$

Naast Modus Ponens hebben axiomatische systemen voor MpL en MPL meestal een afleidingsregel als "wanneer φ een axioma is, dan $\Box\varphi$ ook".

Door $\Box$ en $\Diamond$ op een specifieke manier te interpreteren ontstaan allerlei toepassingen van MpL en MPL. Een voorbeeld daarvan, de *tijdslogica* die ontstaat door eenplaatsige tijdsoperatoren te introduceren hebben we in § 5.10 al aangestipt. De modellen maken gebruik van een *eerder dan* relatie tussen tijdstippen. De operator F, voor '*eens* in de toekomst ...' is qua betekenis analoog aan $\Diamond$. $\Box$ heeft ook een analogon, namelijk de operator G, voor '*altijd* in de toekomst ...' (G is een afkorting voor 'it is *g*oing to be the case'). Net zo hebben we een 'universele' en een 'existentiële' tijdsoperator voor verleden tijd: P staat voor '*eens* in het verleden ...'; H voor '*altijd* in het verleden ...' (H is een afkorting voor 'it *h*as been the case'). Verder kunnen modaal-logische systemen bij voorbeeld worden gebruikt om de logica van *weten* en *geloven* bloot te leggen.

Filosofisch is er veel gekrakeel geweest rond de implicaties van de mogelijke werelden metafoor als uitleg voor 'noodzakelijkheid'. Wanneer u weet dat 'noodzakelijkheid' een kernbegrip is in allerlei vormen van metafysica, dan kunt u zich voorstellen dat discussies over wat het *betekent* dat iets 'anders had kunnen zijn dan het in feite is' kunnen voeren tot grote hoogten van filosofische spitsvondigheid. Betekent 'stand van zaken A had anders kunnen zijn' bij voorbeeld hetzelfde als 'A is *a posteriori* waar'? En is dat weer hetzelfde als of iets anders dan 'A is een *synthetische bewering*'?

7.7 Intensionele typenlogica

Intensionele logica is een andere benaming voor modale logica (of eigenlijk: voor vormen van logica waarin de semantiek gebruik maakt van mogelijke werelden). *Intensionele typenlogica* ontstaat door het combineren van typenlogica en modale logica. Het gebruik van deze combinatie voor het weergeven van de betekenis van uitdrukkingen uit de natuurlijke taal is voorgesteld door de logicus Richard Montague. Zie [Montague 1974], maar veel toegankelijker is [Gamut 1982], deel 2. Montague's voorstellen hebben

geleid tot een aparte school in het onderzoek van natuurlijke taal, de zogenaamde Montague-grammatica.

Montague maakt de stap van extensionele naar intensionele typenlogica omdat hij de zogenaamde 'intensionele verschijnselen' in de natuurlijke taal wil kunnen vangen in de logische vertalingen voor uitdrukkingen uit de natuurlijke taal. Die logische vertalingen worden dan vervolgens geïnterpreteerd in modellen voor intensionele typenlogica.

Wellicht het beroemdste voorbeeld van een intensioneel verschijnsel is Frege's *ochtendster-avondster paradox*. In de Oudheid hadden de ochtendster (dat wil zeggen het laatst verdwijnende lichtpuntje in het Oosten aan de ochtendhemel) en de avondster (dat wil zeggen het eerst verschijnende lichtpuntje in het Westen aan de avondhemel) verschillende namen. Toen ontdekten de Babyloniërs dat het hier om een en hetzelfde hemellichaam gaat (namelijk de planeet Venus). Dus:

(26) *De Babyloniërs ontdekten dat de ochtendster de avondster is.*

Dat klopt ook:

(27) *De ochtendster is de avondster.*

Maar als de twee termen *de ochtendster* en *de avondster* niets anders zijn dan verwijzingen naar een en hetzelfde object, dan zou je ze met behoud van waarheidswaarde voor elkaar moeten kunnen substitueren. En dan zou je moeten kunnen zeggen:

(28) *De Babyloniërs ontdekten dan de avondster de avondster is.*

Maar deze zin is *niet* waar, want dat de avondster identiek is aan zichzelf wisten de Babyloniërs natuurlijk altijd al. De context *De Babyloniërs ontdekten dat...* is blijkbaar een omgeving waarbinnen het principe van substitutie met behoud van waarheidswaarde voor termen die naar hetzelfde object verwijzen *niet* meer opgaat. Zo'n context noemen we een *intensionele context* (Engels: intensional context), of ook wel: een *referentieel ondoorzichtige context* (Engels: referentially opaque context).

Intensioneel is de tegenhanger van *extensioneel*: een context waarbinnen het substitutie-principe als boven wel opgaat heet *extensioneel*. Een andere benaming is: *referentieel doorzichtige context* (Engels: referentially transparent context). Ook de contexten *Het is noodzakelijk dat...* en *Het is mogelijk dat...* zijn intensioneel. Immers, uit:

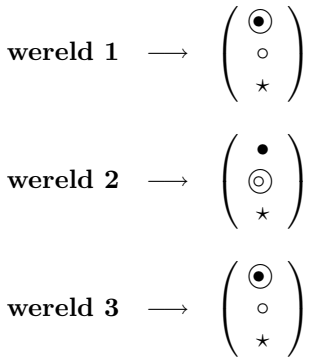
(29) *Het is noodzakelijk dat de logicadocenten de stof goed uitleggen.*

volgt nog niet:

(30) *Het is noodzakelijk dat Jan en Elias de stof goed uitleggen.*

ook al is het in de mogelijke wereld waarin wij ons bevinden (de werkelijke wereld) zo dat Jan en Elias de logica-docenten zijn. Immers, in andere mogelijke werelden (in andere voorstelbare standen van zaken) hoeft het niet zo te zijn dat Jan en Elias de logica-docenten zijn. In *die* mogelijke werelden moeten niet Jan en Elias de stof goed uitleggen, maar degenen die *daar* de logica-docenten zijn.

De bovenstaande parafrase geeft al aan in welke richting we de oplossing moeten zoeken. Blijkbaar moeten we met *de avondster*, *de ochtendster*, *de logica-docent* niet zonder meer een individu associëren, maar veeleer een *functie* die in elke mogelijke wereld een individu oplevert. Dus: *binnen* een mogelijke wereld is de verwijzing van deze termen een individu (bij voorbeeld: de planeet Venus, of Jan), maar *globaal* is het een functie van de verzameling W van mogelijke werelden naar individuen in die werelden. De verwijzing-in-engere-zin noemen we de *extensie* van de term. De 'globale' verwijzing noemen we de *intensie*. Frege's ochtendster-avondster paradox kan nu worden opgelost door over te stappen van extensies naar intensies. Weer is een plaatje duizend woorden waard. Een plaatje voor de *intensie* van *de logica-docent*:



Zoals u ziet: een functie van de verzameling mogelijke werelden $W = \{$wereld 1, wereld 2, wereld 3$\}$ naar individuen in die werelden. De *extensie* van *de logica-docent* is in wereld 1 de waarde die de *intensie*-functie daar oplevert; in wereld 2 en in wereld 3 evenzo.

Het intensie-extensie onderscheid kunnen we ook goed gebruiken voor de analyse van zinnen als:

(31) *Marietje zoekt de logica-docent.*

Het zou kunnen wezen dat Marietje helemaal niet weet dat Jan de logica-docent is. Ze loopt gewoon te zoeken naar de persoon die het vak logica verzorgt, wie dat ook moge zijn. Hieraan zien we dat we uit bovenstaande

zin en

(32) *Jan is de logica-docent.*

niet mogen concluderen tot:

(33) *Marietje zoekt Jan.*

Conclusie: ook de context *Marietje zoekt...* is intensioneel. De technische oplossing die door Montague is voorgesteld maakt weer gebruik van mogelijke werelden. Neem aan dat *zoeken* wordt geïnterpreteerd als een relatie tussen individuen (de zoekers) en *intensies van individuen* (die dan staan voor 'concepties' van wat wordt gezocht). Slag om de arm: hier komt het globaal op neer; Montague maakt het allemaal nog veel ingewikkelder, maar wij beperken ons hier tot de Hoofdgedachten van de Meester.

Dat het onderscheid tussen een *intensionele* en een *extensionele* lezing van *zoeken* taalkundig van belang is moge blijken uit het feit dat sommige talen er twee verschillende constructies voor hebben. Vergelijk:

(34) *Jean cherche une amie qui* est *blonde.*

(35) *Jean cherche une amie qui* soit *blonde.*

Het zoeken in de eerste zin is extensioneel: er wordt een individu gezocht. In de tweede voorbeeldzin is het zoeken intensioneel: er wordt een 'conceptie' gezocht, en misschien—wie zal het zeggen—een hersenschim nagejaagd. In het Nederlands hebben we geen *subjonctif*, maar we kunnen het verschil met wat kunst-en vliegwerk ook wel uitdrukken in de vertalingen: "Jan zoekt een vriendin, en ze is blond" versus "Jan zoekt een vriendin die blond moet zijn". Op de technische details van de semantiek voor intensionele typenlogica gaan we hier niet in. Het komt neer op combineren van de ingrediënten voor de semantiek van modale logica met die voor de semantiek voor extensionele typenlogica, iets wat niet al te moeilijk is voor een kok die al weet hoe hij deze twee gerechten afzonderlijk moet toebereiden.

Hoofdstuk 8

Informatica-toepassingen van de logica

8.1 PROLOG: programmeren met logica

Een van de meest praktische toepassingen van logica in de informatica is het gebruik van de logica zelf als programmeertaal. Heel in het algemeen gaat programmeren met logica zo. We nemen een eindige *consistente* verzameling formules in een geschikte taal; dit is ons programma Π. We voegen daar een nieuwe formule $\neg\varphi$ aan toe en we laten het systeem uitmaken of $\Pi \cup \{\neg\varphi\}$ nog steeds consistent is. Als dat niet zo is dan is φ kennelijk waar in alle modellen van Π, ofwel: φ volgt logisch uit Π. Als het wel zo is dan volgt φ *niet* uit Π.

Om een en ander te laten werken moeten de formules uit Π een vorm hebben die garandeert dat eindige consistentie voor dergelijke formules beslisbaar is. De poging om verzameling Π inconsistent te maken door toevoegen van een nieuwe formule heeft immers alleen zin wanneer we weten dat Π zelf consistent is. Verder moeten de formules $\neg\varphi$ die we toevoegen een vorm hebben die garandeert dat de vraag of ze consistent zijn met eindige verzamelingen programma-formules beslisbaar is.

Propositielogica is te arm om als programmeertaal te worden gebruikt; predikatenlogica is daarentegen juist te rijk, want een minimum-eis is dat onze programma's consistent zijn, en de vraag of een eindige verzameling predikatenlogische zinnen consistent is, is in het algemeen onbeslisbaar. Het is dus zaak om een deel van de predikatenlogica af te zonderen met de gewenste eigenschappen.

De ingrediënten van PROLOG programma's zijn predikatenlogische zin-

nen van een speciale vorm.

Definitie 8.1 *Een* **Horn zin** *is een formule van de vorm*

$$\forall x_1 \cdots \forall x_m((A_1 \wedge \cdots \wedge A_n) \rightarrow A)$$

of van de vorm

$$\forall x_1 \cdots \forall x_m(\neg A_1 \vee \cdots \vee \neg A_n),$$

waarbij $A, A_1, \ldots, A_n$ atomaire predikatenlogische formules zijn waarbinnen hoogstens de variabelen $x_1, \ldots, x_m$ vrij voorkomen.

De eigenschappen van Horn zinnen werden bestudeerd (door de logicus A. Horn) lang voor de uitvinding van PROLOG. In PROLOG is een speciale notatie voor Horn zinnen in zwang.

(1) $\forall x_1 \cdots \forall x_m((A_1 \wedge \cdots \wedge A_n) \rightarrow A)$

wordt genoteerd als

(2) A :- $A_1, \ldots, A_n$.

Een Horn zin van de vorm (2) wordt een *programma-clausule* genoemd. De logische formule

(3) $\forall x_1 \cdots \forall x_m(\neg A_1 \vee \cdots \vee \neg A_n)$

wordt genoteerd als

(4) :- $A_1, \ldots, A_n$.

Een Horn zin van de vorm (4) heet een *doelclausule*. De onderdelen A_1 tot en met A_n van een doelclausule heten de *subdoelen* van de doelclausule.

Als we een Horn zin in het algemeen opvatten als een implicatie, dan kan een doelclausule worden gezien als een implicatie met een lege consequent:

(5) $\forall x_1 \cdots \forall x_m(\neg A_1 \vee \cdots \vee \neg A_n)$

is immers equivalent met

(6) $\forall x_1 \cdots \forall x_m(\neg A_1 \vee \cdots \vee \neg A_n \vee \bot)$,

wat weer kan worden geschreven als:

(7) $\forall x_1 \cdots \forall x_m((A_1 \wedge \cdots \wedge A_n) \rightarrow \bot)$.

Hierbij staat $\bot$ voor een contradictie; een altijd onwaar disjunct kan immers zonder bezwaar worden weggelaten. Een andere schrijfwijze voor (5) is:

(8) $\neg\exists x_1 \cdots \exists x_m(A_1 \wedge \cdots \wedge A_n)$.

Aan deze notatie kunnen we zien dat doelclausules de vorm hebben van *negaties*. Wanneer $n = 0$ in een doelclausule dan noteren we dit als

(9) :-

of

(10) $\square$.

Deze formule heet de *lege clausule*, en zij staat eveneens voor een contradictie: het is de PROLOG notatie voor $\bot$. De consequent en de antecedent van de lege clausule zijn beide leeg, hetgeen betekent dat er niets volgt uit de lege antecedent; dit is altijd onwaar.

Wanneer $n = 0$ in een programma-clausule hebben we het volgende speciale geval:

(11) A :-.

Dit wordt ook wel genoteerd als:

(12) A.

Een programma-clausule van deze vorm heet een *programma-feit*. Een feit is een programma-clausule met een lege antecedent. Terugvertaald naar standaardnotatie ziet (11) er zo uit:

(13) $\forall x_1 \cdots \forall x_m A$.

Een programma-clausule van de vorm (2) met $n \neq 0$ (een programma-clausule met een niet-lege antecedent) heet een *programma-regel*. In de programma-regel (2) vormt A de *kop* en $A_1, \ldots, A_n$ de *staart*. Een feit is een programma-clausule met een lege staart.

We kunnen nu een definitie geven van wat we verstaan onder een PROLOG programma.

Definitie 8.2 *Een PROLOG programma is een eindige verzameling van programma-clausules.*

Voorbeeld 8.1 Het volgende PROLOG programma bevat informatie over ons koninklijk huis (in standaard PROLOG notatie beginnen individuele variabelen met hoofdletters of het onderstrepings-teken, en individuele constanten met kleine letters):

```
man(willem_alexander).
man(claus).
man(bernhard).
```

```
vrouw(beatrix).
vrouw(juliana).

ouder_van(claus,willem_alexander).
ouder_van(beatrix,willem_alexander).
ouder_van(juliana,beatrix).
ouder_van(bernhard,beatrix).

vader_van(X,Y) :- man(X), ouder_van(X,Y).

moeder_van(X,Y) :- vrouw(X), ouder_van(X,Y).
```

De eerste negen programma-clausules in dit voorbeeld zijn feiten, de twee overige zijn regels. Een eindige verzameling van PROLOG feiten is niets anders dan een zogenaamd *relationeel gegevensbestand*; een efficiënt PROLOG systeem is geknipt voor het manipuleren van relationele gegevensbestanden.

Wanneer het programma uit voorbeeld 8.1 is geladen kunnen we vragen stellen met behulp van doelclausules. Een voorbeeld van een ja–nee vraag:

```
| ?- man(willem_alexander).

yes
| ?-
```

Hier is `| ?-` de PROLOG prompt. PROLOG interpreteert alles wat de gebruiker intikt als doelclausule.

Vraagwoord-vragen kunnen worden gesteld met behulp van doelclausules die variabelen bevatten. Een mogelijke vraag is bij voorbeeld: *Wie zijn mannen?*

```
| ?- man(X).

X = willem_alexander
```

Het systeem wacht na het geven van de eerste passende `X`; de gebruiker kan opdracht geven tot verder zoeken door intikken van `;`. Dit levert:

```
| ?- man(X).

X = willem_alexander ;
```

```
X = claus ;

X = bernhard ;

no
| ?-
```

Na het derde verzoek om verder speurwerk geeft het systeem te kennen dat er geen nieuwe antwoord-substituties te vinden zijn.

Merk op dat de regels in voorbeeld 8.1 enkel variabelen bevatten. In een regel kunnen ook individuele constanten voorkomen. Bij voorbeeld:

```
kind_van_bernhard(X) :- ouder_van(bernhard,X).
```

Regels van de meer algemene vorm zijn echter nuttiger. De vraag naar de kinderen van Bernhard kunnen we immers ook als volgt stellen:

```
| ?- ouder-van(bernhard,X).
```

Complexe vraagwoord-vragen kunnen worden gesteld door gebruik te maken van meerledige doelclausules.

```
| ?- moeder_van(juliana,X), ouder_van(X,Y).

X = beatrix,
Y = willem_alexander ;

no
| ?-
```

Deze vraag suggereert de definitie van het begrip *grootmoeder* met behulp van de volgende regel:

```
grootmoeder_van(X,Y) :- moeder_van(X,Z), ouder_van(Z,Y).
```

Terugvertaald naar standaard-notatie:

$$\forall x \forall y \forall z((Mxz \wedge Ozy) \rightarrow Gxy).$$

Dit is equivalent met:

$$\forall x \forall y(\exists z(Mxz \wedge Ozy) \rightarrow Gxy).$$

U ziet: bij het programmeren in PROLOG komen de vaardigheden in het vertalen van Nederlands naar predikatenlogica die u in hoofdstuk 6 heeft aangeleerd goed van pas.

Na uitbreiding van het programma uit voorbeeld 8.1 met de nieuwe regel reageert het systeem als volgt op het bevragen van de grootmoeder-relatie:

```
| ?- grootmoeder_van(X,Y).

X = juliana,
Y = willem_alexander ;

no
| ?-
```

Het systeem gebruikt regels door hun kop te 'unificeren' met de gestelde vraag, en dan de zo ontstane nieuwe doelen te onderzoeken. Unificeren is gelijk maken door het vinden van een passende substitutie voor de variabelen. Het doel

```
| ?- grootmoeder_van(X,Y).
```

wordt geünificeerd met de kop van de regel

```
grootmoeder_van(X,Y) :- moeder_van(X,Z), ouder_van(Z,Y).
```

Dit levert de volgende nieuwe subdoelen:

```
?- moeder_van(X,Z), ouder_van(Z,Y).
```

Het eerste subdoel wordt geünificeerd met de regel

```
moeder_van(X,Y) :- vrouw(X), ouder_van(X,Y).
```

Dit levert de volgende nieuwe subdoelen:

```
?- vrouw(X), ouder_van(X,Z), ouder_van(Z,Y).
```

Vervolgens worden de feiten voor *vrouw* en *ouder* successievelijk uitgeprobeerd tot een substitutie voor de individuele variabelen gevonden is die aan alle condities voldoet.

Opdracht 8.1 *Breid het programma uit voorbeeld 8.1 uit met meer feiten in termen van* `man`*,* `vrouw` *en* `ouder_van`*, en met regels die de begrippen* broer*,* zus*,* oom *en* tante *definiëren.*

Opdracht 8.2 *Breid het programma uit voorbeeld 8.1 uit met een predikaat* `voorouder_van`*.*

Opdracht 8.3 *Programmeer de inhoud van de volgende onsterfelijke tekst: "Al die willen uit varen gaan, moeten mannen met baarden zijn. Jan, Piet, Joris en Korneel, die hebben baarden, zij varen mee." Gebruik predikaten* `man`*,* `vrouw`*,* `bebaard` *en* `vaart_mee`*. Hoe wordt de vraag naar wie er meevaart gesteld, en wat is het antwoord?*

8.2 PROLOG met gestructureerde data

Zolang we alleen individuele constanten en variabelen gebruiken zal het domein waar we over praten eindig zijn: omdat een PROLOG programma eindig is kunnen er slechts eindig veel objecten bij name worden genoemd. PROLOG programma's worden interessanter wanneer we functieconstanten gebruiken om gestructureerde objecten weer te geven. De functieconstanten maken het mogelijk om een oneindig domein van objecten op te bouwen.

Voorbeeld 8.2 Dit programma illustreert het gebruik van functie-constanten:

```
:- op(600, fy, s).

plus(X, 0, X).
plus(X, s Y, s Z) :- plus(X, Y, Z).
```

De doelclausule met `op` is nodig om `s` te declareren als een prefix functieconstante, eenplaatsig en met 'precedentie' 600. Meer informatie over `op` is te vinden in elk PROLOG leerboek. We gebruiken `s` als naam voor de opvolger functie. De representatie voor de natuurlijke getallen wordt dus: $\{0, s0, ss0, sss0, \ldots\}$.

Het voorbeeldprogramma geeft een definitie van optelling voor de natuurlijke getallen. De relatie `plus` geldt tussen drie getallen wanneer optellen van het eerste getal bij het tweede getal het derde getal oplevert. Terugvertalen naar standaard-notatie is weer instructief (s staat voor de opvolger-functie):

$$\forall x Px0x \wedge \forall x \forall y \forall z (Pxyz \rightarrow Pxsysz).$$

Vragen hoeveel twee plus twee is gaat nu als volgt:

```
| ?- plus(s s 0, s s 0, X).

X = s s s s 0 ;

no
```

```
| ?-
```

Vragen naar alle mogelijkheden om door middel van optelling van natuurlijke getallen het getal *drie* te krijgen:

```
| ?- plus(X, Y, s s s 0).

X = s s s 0,
Y = 0 ;

X = s s 0,
Y = s 0 ;

X = s 0,
Y = s s 0 ;

X = 0,
Y = s s s 0 ;

no
| ?-
```

Voorbeeld 8.3 We kunnen het programma uit voorbeeld 8.2 verder uitbreiden door clausules voor vermenigvuldiging toe te voegen:

```
maal(_, 0, 0).
maal(X, s Y, U) :- maal(X, Y, Z), plus(X, Z, U).
```

In de eerste clausule staat `_` voor een variabele. Variabelen in PROLOG beginnen met een hoofdletter of een _-teken. Het verschil tussen deze twee is dat _-variabelen uniek zijn in een clausule: ze spelen om zo te zeggen slechts een bijrol; het eigenlijke werk wordt verricht door variabelen die meerdere keren in één clausule voorkomen.

Het programma uit voorbeeld 8.3 kan als volgt worden bevraagd:

```
| ?- maal(X,X,Y).

X = Y = 0 ;

X = Y = s 0 ;

X = s s 0,
```

```
Y = s s s s 0 ;

X = s s s 0,
Y = s s s s s s s s s 0 ;

X = s s s s 0,
Y = s s s s s s s s s s s s s s s s 0
```

Het is duidelijk dat elke nieuwe ;–aansporing een nieuw antwoord oplevert. Dit gaat door totdat de gebruiker er genoeg van heeft of de zaak wordt afgebroken met een foutmelding (*'out of memory'* of iets dergelijks). Dit gedrag is natuurlijk correct: de gestelde vraag heeft nu eenmaal een oneindig aantal goede antwoorden. De vraagstelling suggereert meteen de definitie van een nieuw predikaat voor kwadrateren:

```
kwadr(X, Y) :- maal(X, X, Y).
```

Er zijn overigens ook voorbeelden waar PROLOG 'ten onrechte' in een lus terechtkomt:

Voorbeeld 8.4 Beschouw dit programma:

```
getrouwd_met(claus, beatrix).
getrouwd_met(X,Y) :- getrouwd_met(Y,X).
```

Logisch gesproken is een en ander correct: de regel drukt uit dat de *getrouwd zijn met* relatie symmetrisch is. Het PROLOG systeem verslikt zich hierin: geconfronteerd met de vraag `:- getrouwd_met(X,Y)` zal het eindeloos de antwoorden `X = claus, Y = beatrix` en `X = beatrix, Y = claus` blijven herhalen. Overigens is hier in dit geval iets aan te doen door preciezer formuleren. Het volgende programma vermijdt de vicieuze cirkel:

```
getrouwd_met_mv(claus, beatrix).
getrouwd_met_mv(bernhard, juliana).

getrouwd_met(X,Y) :- getrouwd_met_mv(X,Y).
getrouwd_met(X,Y) :- getrouwd_met_mv(Y,X).
```

Het voorbeeld illustreert dat de PROLOG programmeur niet alleen rekening moet houden met de logische betekenis van de programma-clausules maar ook met de manier waarop PROLOG het programma verwerkt. PROLOG programma's hebben niet alleen een *logische* of *declaratieve* interpretatie maar ook een *procedurele* interpretatie. De logische en de procedurele kijk leveren helaas niet altijd hetzelfde plaatje op.

Opdracht 8.4 *Breid de clausules uit voorbeelden 8.2 en 8.3 uit met een predikaat* `fac(X,Y)` *voor het uitrekenen van de faculteitsfunctie. De recursieve definitie van deze functie luidt als volgt:*

- $0! = 1$*;*
- $(n+1)! = n! \cdot (n+1) \quad (n \geq 0)$.

Voorbeeld 8.5 De natuurlijke getallen worden in de programma's uit voorbeelden 8.2 en 8.3 gerepresenteerd als gestructureerde objecten. Een ander voorbeeld van het gebruik van functiesymbolen voor het aanduiden van gestructureerde objecten is het gebruik van functies voor het construeren van lijsten. Lijsten of eindige rijtjes kunnen worden opgebouwd uit de lege lijst, die we `nil` zullen dopen, en een verzameling van willekeurige objecten, met behulp van een constructie-functie `.` die een nieuw element voor aan een lijst plakt. Een BNF regel voor de opbouw van lijsten is

lijst ::= **nil** | *element* `.` *lijst*

en in PROLOG ziet dit er als volgt uit:

```
:- op(600, xfy, .).

lijst(nil).
lijst(_Kop.Staart) :- lijst(Staart).
```

Weer is de doelclausule met `op` bedoeld om een functiesymbool te declareren: tweeplaatsig, infix-notatie, rechts-vertakkend en met 'precedentie' 600. De PROLOG leerboeken geven meer informatie.

De vraag of `a.b.c.nil` een lijst is wordt als volgt gesteld en beantwoord:

```
| ?- lijst(a.b.c.nil).

yes
| ?-
```

Voorbeeld 8.6 De volgende uitbreiding van het programma in voorbeeld 8.5 maakt het mogelijk om te praten over de elementen van een lijst.

```
elem(X,X.Y) :- lijst(Y).
elem(X,_Kop.Staart) :- elem(X,Staart).
```

Een object is element van een lijst L als dat object (i) de kop is van lijst L of (ii) een element is de staart van lijst L.

Hier ziet u het predikaat in actie:

```
| ?- elem(b,a.b.c.nil).

yes
| ?- elem(d,a.b.c.nil).

no
| ?- elem(X,a.b.c.nil).

X = a ;

X = b ;

X = c ;

no
| ?-
```

In feite heeft PROLOG een iets handiger notatie voor lijsten ingebakken. De lijst `nil` wordt genoteerd als `[]`, de lijst `a.b.c.nil` als `[a,b,c]`, en in het algemeen, de lijst met kop `a` en een willekeurige staart als `[a|X]` of `[a|_]`. De notatie met `|` en variabelen specificeert in feite *lijstpatronen*. Het patroon `[a,b|_]` specificeert een lijst met eerste element `a`, tweede element `b`, en een willekeurige staart. Let op: elementen van lijsten kunnen zelf weer lijsten zijn: `[a]` is een lijst met één element, te weten het object `a`, en `[[a]|_]` is het patroon van alle lijsten die met de lijst `[a]` beginnen. Merk op dat `[a|[]]` een andere aanduiding is voor de lijst `[a]`.

Opdracht 8.5 *Omschrijf de volgende lijstpatronen:*

1. `[a,_,b|_]`.
2. `[[a,_,b]|_]`.
3. `[[]|_]`.
4. `[[],_]`.
5. `[_,_,a]`.

Het algemene patroon van een niet-lege lijst is: `[_|_]` of `[Kop|Staart]`.

Voorbeeld 8.7 Door gebruik te maken van de PROLOG lijst notatie kan het bovenstaande lijstprogramma vervangen worden door de volgende clausules:

```
element(X,[X|_Staart]).
element(X,[_Kop|Staart]) :- element(X,Staart).
```

Het apart definiëren van het begrip *lijst* is nu overbodig geworden: het gebruik van de haken `[` en `]` dwingt af dat het tweede argument van `element` een lijst is.

Als we vragen wanneer `a` een element is van een lijst krijgen we de volgende antwoorden:

```
| ?- element(a,X).

X = [a|_224] ;

X = [_223,a|_226] ;

X = [_223,_225,a|_228] ;

X = [_223,_225,_227,a|_230]
```

Dit is wat er letterlijk op het scherm verschijnt; de getalsaanduidingen van de variabelen (`_223`, `_224`, enzovoorts) zijn 'toevallig'; het PROLOG systeem kiest deze aanduidingen op grond van de geheugenlocaties waar de variabelen zich bevinden.

Parafrase: `a` is element van `X` wanneer `X` een lijst is met `a` als eerste element, of een lijst met `a` als tweede element, of een lijst met `a` als derde element, of een lijst met `a` als vierde element, enzovoorts. Weer is het aantal mogelijke antwoorden oneindig. De antwoordenreeks kan worden afgebroken door een *'return'* te geven in plaats van een druk op de ';' toets.

Opdracht 8.6 *Welke lijst-eigenschap wordt gedefinieerd door het volgende predikaat:*

```
xyz([]).
xyz([_,_|Staart]) :- xyz(Staart).
```

Opdracht 8.7 *Welke lijst-eigenschap wordt gedefinieerd door het volgende predikaat:*

```
zyx(X,Y,[X,Y|_]).
zyx(X,Y,[_|Staart]) :- zyx(X,Y,Staart).
```

Voorbeeld 8.8 Twee lijsten aan elkaar plakken ('concateneren') gebeurt met het volgende programma:

```
concat([],X,X).
concat([Kop|Staart],X,[Kop|Y]) :- concat(Staart,X,Y).
```

Hier ziet u het programma in actie:

```
| ?- concat([a,b],[c,d],X).

X = [a,b,c,d] ;

no
| ?-
```

De volgende interactie illustreert het gebruik van `concat` om te vragen naar het 'verschil' van twee lijsten:

```
| ?- concat([a],X,[a,b,c,d]).

X = [b,c,d] ;

no
| ?-
```

Ook kunnen we vragen naar alle mogelijke manieren om een lijst op te splitsen in een prefix en een suffix:

```
| ?- concat(X,Y,[a,b,c]).

X = [],
Y = [a,b,c] ;

X = [a],
Y = [b,c] ;

X = [a,b],
Y = [c] ;

X = [a,b,c],
Y = [] ;

no
| ?-
```

Opdracht 8.8 *Definieer een predikaat* `laatste(X,Y)` *dat waar is desda* `X` *het laatste element is van een niet-lege lijst* `Y`.

Opdracht 8.9 *Definieer een predikaat* `keerom(X,Y)` *dat een lijst* `X` *omkeert, met het resultaat in* `Y`*. Met andere woorden:* `keerom(X,Y)` *is waar desda* `Y` *de omkering van lijst* `X` *bevat. Aanwijzing: begin met het opschrijven van een recursieve definitie van de omkeer-procedure en formuleer die definitie daarna in PROLOG.*

De bovenstaande discussie is bedoeld om u een zeker gevoel bij te brengen voor programmeren in PROLOG. Goede bronnen voor meer informatie zijn [Bradko 1986] en [Pereira & Shieber 1987]; het laatste boek geeft toepassingen van PROLOG voor de analyse van natuurlijke taal.

8.3 De PROLOG bewijsstrategie

Zoals reeds vermeld hebben PROLOG programma's een declaratieve en een procedurele interpretatie. In de procedurele interpretatie is een programmaclausule die de vorm heeft van een regel een definitie van een *procedure*: om twee lijsten `X` en `Y` aan elkaar te plakken moet je eerst de staart van lijst `X` concateneren met lijst `Y`, en vervolgens de kop van lijst `X` voor het resultaat zetten. Een doelclausule kan procedureel worden opgevat als een opdracht om een procedure uit te voeren. Het lege doel $\square$ kan worden opgevat als de lege procedure, ofwel: de opdracht om niets te doen, dat wil zeggen de halt-opdracht.

Voor een goed begrip van PROLOG is inzicht in het bewijsproces dat eraan ten grondslag ligt van belang, want de logische en de procedurele interpretatie van PROLOG programma's gaan niet altijd gelijk op. Als PROLOG puur declaratief zou zijn zou de programmeur zich niet hoeven bekommeren om de 'geest in de machine'. De redeneerregel waar alles om draait is zogenaamde *resolutie met unificatie*. Eerst een voorbeeld van resolutie zonder unificatie. Neem aan dat A, B, C, D, E, F predikaatletters zonder variabelen zijn (nulplaatsige predikaatletters of predikaatletters gevolgd door constanten):

$$\frac{A \text{ :- } B, C, D \qquad C \text{ :- } E, F}{A \text{ :- } B, E, F, D.}$$

De resolutie vindt hier plaats op C. Omdat de predikaten geen variabelen bevatten zijn er geen substituties nodig.

Hier is een voorbeeld van resolutie met unificatie; om de resolutie te doen slagen moeten er nieuwe waarden voor de variabelen worden gesubstitueerd.

$$\frac{A(x, y) \text{ :- } B(x), C(x, fy) \qquad C(a, u) \text{ :- } D(u), E(gu)}{A(a, fy) \text{ :- } B(a), D(fy), E(gfy).}$$

De resolutie vindt plaats op C; de unificerende substitutie—dat wil zeggen de gelijktijdige substitutie die de atomaire formules waarop de resolutie plaats vindt aan elkaar gelijk maakt—is $\{a/x, fy/u\}$, of in PROLOG notatie: `X = a`, `U = fY`.

In een PROLOG systeem wordt de resolutie-met-unificatie techniek als volgt gebruikt. Het systeem tracht een tegenspraak af te leiden uit $\neg\exists x_1 \cdots \exists x_m(A_1 \wedge \cdots \wedge A_m)$, genoteerd als :- $A_1, \ldots, A_m$, en een verzameling programma-clausules Π. Daartoe wordt het eerste (meest linkse) subdoel geünificeerd met de bovenste in aanmerking komende clausule in de lijst van programma-clausules. PROLOG maakt daarbij essentieel gebruik van het feit dat de verzameling Π geordend is als een lijst. Resolutie levert vervolgens een nieuwe lijst van subdoelen op: wanneer de gebruikte programma-clausule een feit is zal de nieuwe lijst een item minder bevatten; wanneer het een regel is kan de lijst van subdoelen groeien. De gevonden waarden van de variabelen worden opgeslagen; in PROLOG jargon: de *bindingen* van de variabelen blijven bewaard. Als er een vraagwoord-vraag is gesteld gaan die opgeslagen waarden de antwoorden vormen. Vervolgens wordt het nieuwe eerste subdoel vergeleken met de in aanmerking komende programma-clausules, weer van boven naar beneden, enzovoorts.

Wanneer dit proces vastloopt omdat een subdoel S niet kan worden geünificeerd gaat het systeem *terugkrabbelen* (de Engelse benaming voor dit proces is *backtracking*). Het systeem 'keert dan terug in de zoekruimte'. De laatst geselecteerde programma-clausule P (die S ergens in zijn staart heeft, en die gebruikt was voor de resolutie van doel D) wordt verworpen, het systeem maakt de substitutie die D oploste tegen de kop van P ongedaan en neemt D opnieuw in beschouwing door te proberen D te unificeren met een van de clausules die op P volgen. Wanneer dit proces uiteindelijk een lege doelclausule oplevert zeggen we dat het falsum $\Box$ is afgeleid; hiermee is $\neg\exists x_1 \cdots \exists x_m(A_1 \wedge \cdots \wedge A_m)$ weerlegd, en de gevonden bindingen voor de variabelen leveren de antwoord-substituties.

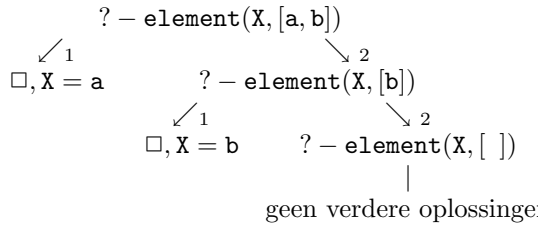
Ook na het vinden van een weerlegging (het afleiden van de lege doelclausule $\Box$) krabbelt het systeem terug om nieuwe antwoorden te vinden. De unificatie die het laatste subdoel S heeft weggewerkt door middel van resolutie tegen een feit P in de lijst van programma-clausules wordt ongedaan gemaakt, en er wordt gepoogd een substitutie te vinden die S unificeert met de kop van een van de programma-clausules die op P volgen. Dit kan leiden tot nieuwe antwoord-substituties, als $\Box$ vanaf dit punt op andere manieren kan worden afgeleid, of tot verder terugkrabbelen op de wijze die in de vorige alinea is uiteengezet.

In het PROLOG zoek- en reken-algoritme zitten twee keuze-elementen, te weten:

- **PROLOG rekenregel:** pak altijd eerst het meest linkse subdoel in de doel-clausule aan.
- **PROLOG zoekregel:** doorzoek de lijst van programma-clausules altijd van boven naar beneden.

Tezamen levert dit een 'eerst links en omlaag' bewijsstrategie op. Voordeel van deze strategie is de efficiëntie. Nadeel is dat de bewijsstrategie niet *volledig* is. Er is geen garantie dat elke weerlegging wordt gevonden omdat het systeem terecht kan komen in een oneindig zoekpad 'links-omlaag' in de boom van mogelijke oplossingen, terwijl er rechts nog niet geëxploreerde eindige paden zijn die naar een oplossing leiden.

Het is illustratief om voor een gegeven voorbeeldprogramma en doelclausule de 'boom van mogelijke oplossingen' uit te tekenen. Neem weer het programma uit voorbeeld 8.7, en beschouw de vraag `?- element(X,[a,b])`. We nummeren de twee programma-clausules als 1 en 2. De bewijsboom ziet er nu zo uit (alleen bevraagde variabelen zijn in de boom opgenomen):

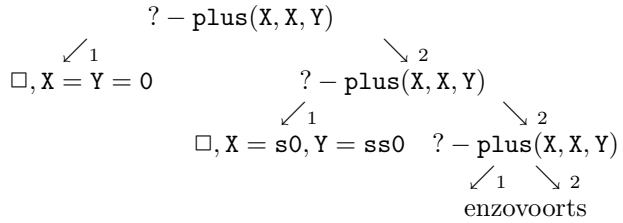


Deze boom wordt volgens de 'eerst links en naar beneden' strategie doorzocht. Merk op dat de bewijsboom voor dit voorbeeld eindig is: doorzoeken volgens een andere strategie (bij voorbeeld: 'eerst rechts en in de breedte') zou dezelfde antwoorden opleveren.

Beschouw nu het programma uit voorbeeld 8.2. We nummeren de twee programma-clausules weer als 1 en 2. De bewijsboom voor de vraag

```
?- plus(X,X,Y)
```

ziet er als volgt uit:

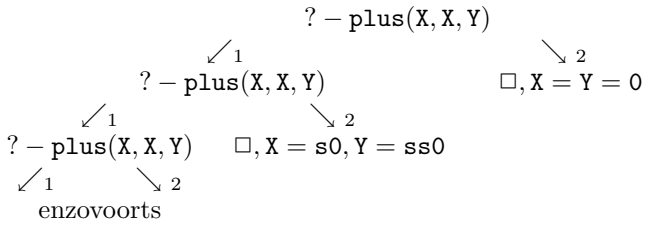


Deze bewijsboom heeft een oneindige tak; bij doorlopen van de boom volgens de PROLOG bewijsstrategie komen er oneindig veel antwoordsubstituties te voorschijn.

Veronderstel nu dat we de volgorde van de twee programma-clausules omkeren:

```
plus(X, s Y, s Z) :- plus(X, Y, Z).
plus(X, 0, X).
```

Weer nummeren we de twee clausules als 1 en 2. Een bewijsboom voor de vraag `?- plus(X,X,Y)` ziet er nu zo uit:



PROLOG zal deze boom weer volgens de 'eerst links en naar beneden' strategie doorlopen, en ... het systeem raakt daarbij in een oneindige lus waarbij geen antwoorden worden gegenereerd.

De moraal: bij het schrijven van PROLOG programma's en bij het bevragen van die programma's moeten we er voor zien te zorgen dat een eventuele oneindige tak in de bewijsboom voor dat programma met die vraag altijd de meest rechtse tak in de boom is. Vaak lukt dit door ordening van de programma-clausules (feiten altijd vóór regels), maar vaak ook niet. In dit laatste geval is de enige uitweg het gebruik van de zogenaamde *snoei-operator* (Engels: *cut operator*) om een gedeelte van de bewijsboom weg te snoeien. De snoei-operator is een zuiver procedureel werktuig: het is nodig om van PROLOG een volwassen procedurele programmeertaal te maken, maar het verstoort de illusie van PROLOG als puur declaratieve taal. Bespreking van het gebruik van de snoei-operator valt buiten het bestek van dit boek.

8.4 Modellen voor PROLOG programma's

We schakelen terug naar het puur logisch/declaratieve perspectief op PROLOG programma's voor een paar semantische vragen.

Definitie 8.3 *Het* **Herbrand universum** *van een PROLOG programma* Π*, notatie* H_Π*, is de verzameling van alle variabel-vrije termen in de taal*

van Π*; als zulke termen niet bestaan omdat de taal van* Π *geen individuele constanten bevat dan nemen we een constante a als voorgift.*

Een eenvoudig Herbrand universum zag u al in voorbeeld 8.2: de verzameling **N**. De volgende stelling is elementair maar cruciaal.

Stelling 8.1 *Elk PROLOG programma is consistent.*

Bewijs: Laat Π een verzameling programma-clausules zijn in de predikatenlogische taal $\mathcal{T}$. We construeren een model voor Π. We nemen het Herbrand universum H_Π als domein van het model. We interpreteren een individuele constante c als die constante zelf. Net zo: de interpretatie van een functie-constante f toegepast op een n-tal termen $t_1, \ldots, t_n$ is $f(t_1, \ldots, t_n)$ zelf. Neem voor predikaatletters de ruimst mogelijke interpretatie: voor 0-plaatsige predikaatletters (dat wil zeggen: propositieletters) is dat de waarde 1, voor 1-plaatsige de verzameling objecten H_Π, voor 2-plaatsige H_Π^2 en in het algemeen voor n-plaatsige predikaten de verzameling H_Π^n. Omdat elke atomaire formule in Π dan waar is, zijn ook alle programma-feiten (universele generalisaties van atomen) waar. Ook zijn alle conjuncties van atomen, implicaties van zulke conjuncties naar atomen, en universele generalisaties over zulke implicaties waar (ga na!), en daarmee zijn de programma-regels in Π waar. ∎

Modellen zoals in stelling 8.1 geconstrueerd staan bekend als *Herbrand modellen* (naar de Franse logicus Jacques Herbrand). Niet alle Herbrand modellen zijn even 'zuinig': de methode uit het bewijs van de stelling maakt elk n-plaatsig predikaat dat in Π voorkomt waar voor alle n-tallen van individuen in het Herbrand universum. Deze interpretatie maakt de clausules uit Π zeker waar, maar daarnaast nog veel meer. De volgende definitie maakt het mogelijk over te stappen op zuiniger Herbrand modellen.

Definitie 8.4 *Laat* $\{\mathcal{M}_i \mid i \in I\}$ *een familie van predikatenlogische structuren zijn voor taal* $\mathcal{T}$ *met allemaal hetzelfde domein en dezelfde operaties. De* **doorsnede** *van deze familie, notatie* $\bigcap_{i\in I} \mathcal{M}_i$*, is de structuur met hetzelfde domein en dezelfde operaties, en met de interpretatie* I *van een* n*-plaatsige predikaatletter* P *als volgt gedefinieerd:*

$$\langle d_1, \ldots, d_n\rangle \in I(P)$$

desda de interpretatie van P *waar is van* $\langle d_1, \ldots, d_n\rangle$ *in elke* $\mathcal{M}_i$*.*

Stelling 8.2 *Als een PROLOG programma* Π *waar is in een familie*

$$\{\mathcal{M}_i \mid i \in I\}$$

van predikatenlogische structuren voor taal $\mathcal{T}$ van Π *met allemaal hetzelfde domein en dezelfde operaties, dan is* Π *waar in de doorsnede van deze familie* $\bigcap_{i\in I}\mathcal{M}_i$.

Bewijs: Neem een willekeurige $\varphi \in \Pi$. Als φ van de vorm $\forall x_1 \cdots \forall x_m A$ is, met A atomair, en φ is waar in elk van de $\mathcal{M}_i$, dan volgt het direct uit de definitie van 'doorsnede' voor structuren dat φ waar is in $\bigcap_{i\in I}\mathcal{M}_i$. Veronderstel nu dat φ de vorm heeft van een regel:

$$\forall x_1 \cdots \forall x_m((A_1 \wedge \cdots \wedge A_n) \rightarrow B).$$

Neem aan dat er een bedeling b is zo dat

$$\bigcap_{i\in I}\mathcal{M}_i \models A_1 \wedge \cdots \wedge A_n \ [b].$$

Dan vervult b elke A_j $(1 \leq j \leq n)$ in elk van de $\mathcal{M}_i$ $(i \in I)$. Omdat gegeven is dat $\mathcal{M}_i \models \varphi$ mogen we concluderen dat b het predikaat B vervult in $\mathcal{M}_i$, voor elke $i \in I$. Dus:

$$\bigcap_{i\in I}\mathcal{M}_i \models B \ [b].$$

Hieruit volgt dat $\bigcap_{i\in I}\mathcal{M}_i \models \varphi$. ∎

Stelling 8.2 geeft ons een methode om een kleinste Herbrand model voor een programma Π te construeren.

Definitie 8.5 *Het* **kleinste Herbrand model** *van een programma* Π *is de doorsnede van alle Herbrand modellen van* Π.

Een Herbrand model voor het programma uit voorbeeld 8.1 heeft het volgende individuendomein: `willem_alexander`, `claus`, `bernhard`, `beatrix` en `juliana`. Een mogelijke interpretatie die alle programma-clausules waar maakt is: elk individu is een man, elk individu is een vrouw, elk individu is ouder van iedereen. Dit is niet de meest zuinige interpretatie. In het kleinste Herbrand model voor het programma zijn `willem_alexander`, `claus` en `bernhard` de mannen, `beatrix` en `juliana` de vrouwen, en is de relatie `ouder_van` beperkt tot

$$\{\langle \texttt{juliana}, \texttt{beatrix}\rangle,$$
$$\langle \texttt{bernhard}, \texttt{beatrix}\rangle,$$
$$\langle \texttt{beatrix}, \texttt{willem\_alexander}\rangle,$$
$$\langle \texttt{claus}, \texttt{willem\_alexander}\rangle\}.$$

Alle Herbrand modellen voor het rekenprogramma uit voorbeeld 8.2 en 8.3 hebben het volgende aftelbaar oneindige domein:

$$\{\texttt{0}, \texttt{s0}, \texttt{ss0}, \texttt{sss0}, \ldots\}.$$

Het grootste van deze modellen maakt *plus* en *maal* waar voor alle drietallen van termen (dat wil zeggen, objecten in het domein van het model). Er zijn ook allerlei zuiniger mogelijkheden. De doorsnede constructie gooit al het overbodige overboord en geeft ons precies wat we willen: de natuurlijke getallen met de standaard interpretatie van *plus* en *maal*.

Herbrand modellen voor het programma uit voorbeeld 8.7 hebben als domein de verzameling van alle lijsten die zijn opgebouwd uit `[]` (immers, `[]` is een constante). Dus: `[]`, `[[]]`, `[[],[]]`, enzovoorts. In het kleinste Herbrand model geldt de *element* relatie van precies die paren met de eigenschap dat de eerste lijst een element is van de tweede.

Om iets meer te kunnen zeggen over kleinste Herbrand modellen hebben we een definitie en een tweetal hulpstellingen nodig.

Definitie 8.6 *Een predikatenlogisch model $\mathcal{M}_1 = \langle D_1, I_1 \rangle$ is een* **substructuur** *van een model $\mathcal{M}_2 = \langle D_2, I_2 \rangle$, notatie $\mathcal{M}_1 \subseteq \mathcal{M}_2$, als het volgende geldt:*

- *$D_1 \subseteq D_2$;*
- *op D_1 stemmen de $\mathcal{M}_2$ functies overeen met de $\mathcal{M}_1$ functies, en D_1 is afgesloten onder die functies;*
- *de interpretaties van individuele constanten worden gezien als 0-plaatsige functies: al die interpretaties bevinden zich in D_1;*
- *op D_1 stemmen de $\mathcal{M}_2$ relaties overeen met die van $\mathcal{M}_1$.*

Intuïtief gesproken is een substructuur van een model voor een taal $\mathcal{T}$ een model voor $\mathcal{T}$ dat mogelijkerwijs een kleiner domein heeft, maar dat zich verder zoveel mogelijk gedraagt als het oorspronkelijke model.

Voorbeelden: de natuurlijke getallen met nul, de opvolger functie en de kleiner dan relatie vormen een substructuur van de reële getallen met nul, opvolger en kleiner dan. Net zo: de even natuurlijke getallen met nul en de kleiner dan relatie vormen een substructuur van de natuurlijke getallen met nul en kleiner dan. In dit laatste geval is het niet mogelijk de opvolger functie toe te voegen, want de even natuurlijke getallen zijn daaronder niet afgesloten.

Opdracht 8.10 *Leg uit waarom de natuurlijke getallen met nul en de operaties voor optellen en aftrekken geen substructuur vormen van de gehele getallen met nul en de operaties voor optellen en aftrekken.*

Stelling 8.3 *Voor elke universele predikatenlogische zin φ, dat wil zeggen voor elke zin φ die equivalent met een formule bestaande uit een rijtje van universele kwantoren gevolgd door een kwantorvrij deel, geldt: als $\mathcal{M}_1 \subseteq \mathcal{M}_2$ en $\mathcal{M}_2 \models \varphi$, dan is $\mathcal{M}_1 \models \varphi$.*

Bewijs: We mogen aannemen dat φ de vorm $\forall x_1 \cdots \forall x_m \psi(x_1, \ldots, x_m)$ heeft, waarbij $\psi(x_1, \ldots, x_m)$ staat voor een kwantorvrije formule waarin de variabelen $x_1, \ldots, x_m$ vrij voorkomen. Neem aan dat $\mathcal{M}_2 \models \varphi$. Dan geldt voor elke bedeling b voor $x_1, \ldots, x_m$ in D_2 dat

$$\mathcal{M}_2 \models \psi(x_1, \ldots, x_m) \ [b].$$

Maar dan vervult ook elke bedeling b' die aan alle variabelen objecten uit D_1 toebedeelt $\psi(x_1, \ldots, x_m)$ in $\mathcal{M}_2$. En omdat $\mathcal{M}_1$ en $\mathcal{M}_2$ op het domein van $\mathcal{M}_1$ met elkaar overeenstemmen wat de interpretatie van functie-constanten en predikaatletters betreft hebben we, voor elke bedeling b' voor $\mathcal{M}_1$:

$$\mathcal{M}_1 \models \psi(x_1, \ldots, x_m) \ [b'].$$

Met andere woorden: $\mathcal{M}_1 \models \varphi$. ∎

Stelling 8.4 *Elke consistente verzameling Γ van universele zinnen heeft een Herbrand model $\mathcal{H}$.*

Bewijs: Omdat Γ consistent is is er een $\mathcal{M} = \langle D, I \rangle$ met $\mathcal{M} \models \Gamma$. We construeren op basis van $\mathcal{M}$ een Herbrand model voor Γ, als volgt. De functie W die de variabelvrije termen van de taal afbeeldt op de waarden die ze volgens de interpretatiefunctie I van $\mathcal{M}$ moeten krijgen, beeldt het Herbrand universum H_Γ af in het domein van $\mathcal{M}$. Als f een m-plaatsige functieconstante is die in Γ wordt gebruikt, dan geldt het volgende:

$$W(f(t_1, \ldots, t_m)) = W(f)(W(t_1), \ldots, W(t_m)).$$

Met andere woorden: W beeldt elke complexe variabelvrije term af op het 'juiste' object in het domein van $\mathcal{M}$.

De definitie van $\mathcal{H}$ is nu simpel. Voor elke predikaatletter P die in Γ wordt gebruikt stipuleren we dat $P(t_1, \ldots, t_n)$ waar is in $\mathcal{H}$ desda

$$\langle I(t_1), \ldots, I(t_n) \rangle \in I(P).$$

Dit legt het Herbrand model $\mathcal{H}$ volledig vast.

We moeten nu nog laten zien dat $\mathcal{H} \models \Gamma$. Omdat alle formules uit Γ universele predikatenlogische zinnen zijn blijft de waarheid ervan bewaard onder het vormen van substructuren (stelling 8.3). Het beeld van H_Γ onder W is een substructuur van $\mathcal{M}$. Alle leden van Γ zijn dus waar in deze substructuur. Uit de definitie van de interpretatiefunctie voor $\mathcal{H}$ volgt nu

dat $\mathcal{H} \models \Gamma$. ∎

De stelling waarmee we deze paragraaf besluiten illustreert het belang van het kleinste Herbrand model van een PROLOG programma Π.

Stelling 8.5 *Voor elke atomaire formule φ en elk PROLOG programma Π geldt: $\Pi \models \varphi$ desda φ waar is in het kleinste Herbrand model voor Π.*

Bewijs: Als $\Pi \models \varphi$ dan is φ zeker waar in het kleinste Herbrand model voor Π, want dit model maakt alle clausules uit Π waar.

Om het omgekeerde te laten zien veronderstellen we dat $\Pi \not\models \varphi$. Dit wil zeggen: er is een model $\mathcal{M}$ voor Π met $\mathcal{M} \models \neg\varphi$. We gebruiken nu de methode van stelling 8.4 om op basis van $\mathcal{M}$ een Herbrand model $\mathcal{H}$ te construeren. Uit de constructie volgt dat $\mathcal{H} \models \neg\varphi$, dat wil zeggen: $\mathcal{H}$ maakt φ niet waar. Maar dan zal het kleinste Herbrand model voor Π (dat alleen de atomaire formules waar maakt die in *elk* Herbrand model voor Π waar zijn), φ onwaar maken. ∎

Deze stelling rechtvaardigt een alternatieve definitie van het kleinste Herbrand model van een PROLOG programma Π als het Herbrand model waarvoor de interpretatie van elke predikaatletter P als volgt gegeven is:

$$\langle t_1, \ldots, t_n \rangle \in I(P) \quad \text{desda} \quad \Pi \models P(t_1, \ldots, t_n).$$

Dit model is bevat in alle Herbrand modellen voor Π.

Het is niet al te moeilijk om te zien dat stelling 8.5 niet alleen opgaat voor atomaire formules, maar ook voor conjuncties van atomaire formules en voor existentiële kwantificaties over zulke conjuncties. Daarmee is onze cirkel rond. We hebben immers in § 8.1 gezien dat doelclausules in PROLOG de volgende algemene vorm hebben:

$$\neg\exists x_1 \cdots \exists x_m (A_1 \wedge \cdots \wedge A_n). \tag{14}$$

PROLOG probeert zo'n doel te weerleggen, met andere woorden: als de weerleggingspoging slaagt is de PROLOG conclusie van de volgende algemene vorm:

$$\exists x_1 \cdots \exists x_m (A_1 \wedge \cdots \wedge A_n). \tag{15}$$

Nog anders gezegd: PROLOG conclusies hebben de vorm van atomaire formules, conjuncties van atomaire formules, of existentiële kwantificaties over atomaire formules of hun conjuncties. We hebben in stelling 8.5 dus bewezen dat voor PROLOG conclusies de twee begrippen *logisch volgen uit programma* Π en *waar zijn in het kleinste Herbrand model van* Π op hetzelfde neerkomen.

Hopelijk heeft deze paragraaf u een indruk gegeven van het theoretische fundament waarop PROLOG rust. Voor meer informatie over de theoretische achtergronden van programmeren met logica verwijzen we naar [Apt 1988].

8.5 Predikatenlogica en correctheid van programma's

Omdat PROLOG programma's een declaratieve lezing hebben wordt programmeren in PROLOG wel aangeduid als *declaratief programmeren*: een programma is in feite een verzameling *definities*. In meer traditionele programmeertalen is een programma een verzameling *opdrachten*; programmeren op deze meer traditionele manier wordt *imperatief programmeren* genoemd.

Met name voor het onderzoek naar correctheid van (imperatieve) programma's is de predikatenlogica van groot belang. Om het gebruik van predikatenlogica bij het leveren van correctheidsbewijzen voor imperatieve programma's te demonstreren, definiëren we een heel simpele imperatieve programmeertaal in BNF notatie. Voor het gemak breiden we de BNF notatie uit met accolades. We spreken af dat $\{A\}$ betekent dat A 0 of meer malen mag voorkomen. Deze uitgebreide BNF notatie wordt EBNF ('Extended Backus Naur Form') genoemd. De definitie van onze programmeertaal ziet er nu zo uit (elke opdracht in de taal kan worden beschouwd als een programma):

Definitie 8.7 *De programmeertaal* **Impertaal**.

⟨*opdracht*⟩ ::= ⟨*variable*⟩ **:=** ⟨*term*⟩
| **begin** ⟨*opdracht*⟩ { **;** ⟨*opdracht*⟩ } **einde**
| **(** **als** ⟨*uitdrukking*⟩ **dan** ⟨*opdracht*⟩ **)**
| **(** **als** ⟨*uitdrukking*⟩ **dan** ⟨*opdracht*⟩ **anders** ⟨*opdracht*⟩ **)**
| **zolang** ⟨*uitdrukking*⟩ **doe** ⟨*opdracht*⟩

⟨*variabele*⟩ ::= ⟨*letter*⟩ | ⟨*letter*⟩ ⟨*getal*⟩

⟨*letter*⟩ ::= **a** | **b** | ⋯ | **x** | **y** | **z**

⟨*getal*⟩ ::= ⟨*cijfer*⟩ { ⟨*cijfer*⟩ }

⟨*cijfer*⟩ ::= **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9**

⟨*term*⟩ ::= ⟨*variable*⟩ | ⟨*getal*⟩ | − ⟨*getal*⟩
| ⟨*term*⟩ + ⟨*term*⟩ | ⟨*term*⟩ − ⟨*term*⟩ | **(** ⟨*term*⟩ ∗ ⟨*term*⟩ **)**

⟨*uitdrukking*⟩ ::= ⟨*term*⟩ = ⟨*term*⟩ | ⟨*term*⟩ < ⟨*term*⟩

| ⟨*term*⟩ > ⟨*term*⟩ | ⟨*term*⟩ <= ⟨*term*⟩
| ⟨*term*⟩ >= ⟨*term*⟩
| **niet** ⟨*uitdrukking*⟩
| **(** ⟨*uitdrukking*⟩ **en** ⟨*uitdrukking*⟩ **)**
| **(** ⟨*uitdrukking*⟩ **of** ⟨*uitdrukking*⟩ **)**

De tweeplaatsige term-operatoren <= en >= staan respectievelijk voor 'kleiner dan of gelijk aan' en 'groter dan of gelijk aan'. Ga na hoe de accolade-notatie die we boven hebben ingevoerd wordt gebruikt bij het herschrijven van ⟨*opdracht*⟩ en ⟨*getal*⟩.

Opdracht 8.11 *Geef een BNF definitie van* ⟨*getal*⟩ *die geen gebruik maakt van accolade-notatie.*

Hier zijn een paar voorbeelden van programma's in **Impertaal**.

- `x := x + 1`
- `(als x < 0 dan x := -x)`
- `(als x > y dan m := x anders m := y)`
- `begin z := x; x := y; y := z einde`
- `begin x := y; y := x einde`
- ```
  begin
    i := 0;
    p := 0;
    zolang i < x doe
      begin p := p + y; i := i + 1 einde
  einde
  ```
- ```
  begin
    i := 0;
    f := 1;
    zolang i < x doe
      begin i := i + 1; f := (i * f) einde
  einde
  ```

De taal **Impertaal** (voor: 'imperatieve taal') die we hebben gedefinieerd is een sterk vereenvoudigde versie van **Pascal**. Variabelen en hun typen hoeven niet te worden gedeclareerd (in feite zijn alle variabelen van het type *geheel getal*). Merk op dat de BNF regels een dubbelzinnigheid uit de definitie van **Pascal** vermijden. De BNF regels voor **Pascal** maken niet duidelijk hoe de opdracht

if U_1 **then if** U_2 **then** O_1 **else** O_2

moet worden gelezen. De twee mogelijkheden zijn:

if U_1 **then (** **if** U_2 **then** O_1 **else** O_2**)**

en

if U_1 **then (** **if** U_2 **then** O_1**)** **else** O_2.

In **Impertaal** treedt de dubbelzinnigheid niet op, omdat de desambiguerende ronde haken verplicht aanwezig zijn.

Goed, we hebben nu de syntaxis van een programmeertaal. Wat we nog moeten vastleggen is wat het *effect* is dat onze programma's hebben. Met andere woorden: we moeten de semantiek van **Impertaal** nog specificeren. Daarvoor gaan we, in navolging van C.A.R. Hoare, predikatenlogica gebruiken. Omdat **Impertaal** zo simpel is, is het enige effect dat een programma kan hebben een verandering van de *toestand* van de machine waarop het programma wordt uitgevoerd. Een machinetoestand is niets anders is dan een specificatie van de waarden die de programma-variabelen hebben. Als bepaalde geheugenplaatsen waar variabelen zijn opgeslagen gevuld zijn met zekere waarden dan is het effect van een programma de verandering die dat programma teweeg brengt in die geheugenplaatsen.

Neem bij voorbeeld het programma `x := x + 1`. Dit programma bestaat uit een enkele *toekenningsopdracht*, een opdracht om een nieuwe waarde toe te kennen aan een variabele. In dit geval wordt de waarde van het getal dat in geheugenlocatie x zit opgeslagen met 1 verhoogd. Wanneer de geheugenlocatie x (de plaats in de computer waarin de waarde voor de variabele x is opgeslagen) voordat het programma gedraaid wordt de waarde 3 heeft, dan moet die locatie na afloop van het programma de waarde 4 hebben. Wat het programma doet is deze toestandsverandering teweeg brengen.

Opdracht 8.12 *Ga na welk effect de bovenstaande voorbeeldprogramma's hebben op de waarden van de variabelen die erin worden gebruikt.*

Bij het uitvoeren van de bovenstaande opdracht hebt u in feite een informele specificatie gegeven van de semantiek van de voorbeeldprogramma's. We gaan dat nu formeler maken: we gaan de voorwaarden waaraan een machinetoestand voldoet uitdrukken met behulp van predikatenlogische formules. We onderscheiden daarbij tussen *beginvoorwaarden* en *eindvoorwaarden*, of in jargon: tussen *precondities* en *postcondities*. Notatie voor de correctheidsbeweringen die we gaan doen:

$\{\varphi\}\ \pi\ \{\psi\}$.

Hierbij is π een programma in **Impertaal** (of een andere imperatieve taal waarover we correctheidsbeweringen willen doen), en zijn φ en ψ respectievelijk een beginvoorwaarde en een eindvoorwaarde voor de variabelen die in het programma gebruikt worden. Let op: de correctheidsbewering $\{\varphi\}\ \pi\ \{\psi\}$ garandeert niet dat het programma π afloopt. Nagaan of een programma termineert (dat wil zeggen: niet in een oneindige lus raakt) is een probleem op zichzelf. Dit probleem is veel moeilijker dan het op het eerste gezicht lijkt, en daarom laten we het hier buiten beschouwing. Omdat onze programmaspecificaties op zichzelf niet garanderen dat de programma's termineren wordt de correctheidsbewering $\{\varphi\}\ \pi\ \{\psi\}$ wel een *partiële correctheidsbewering* genoemd. De parafrase van $\{\varphi\}\ \pi\ \{\psi\}$ luidt:

> Als π een programma is dat termineert, en φ is een conditie die geldt voordat het programma wordt gedraaid, dan geldt conditie ψ wanneer het programma is afgelopen.

Voorbeeld 8.9 $\{x = 3\}$ `x := x + 1` $\{x = 4\}$.

Deze specificatie is juist: het is inderdaad het geval dat wanneer variabele x voor het draaien van programma `x := x + 1` de waarde 3 bevat, die variabele na afloop van het programma de waarde 4 zal bevatten.

Voorbeeld 8.10

$\{x = X, y = Y\}$
`begin z := x; x := y; y := z einde`
$\{x = Y, y = X\}$.

Dit programma termineert, en het is niet moeilijk om in te zien dat het de waarden van x en y verwisselt. Om dit effect aan te kunnen geven ongeacht de beginwaarden van x en y gebruiken we hoofdletters X en Y voor die beginwaarden. De letters X en Y worden *schaduwvariabelen* genoemd (Engels: *ghost variables*).

Voorbeeld 8.11

$\{x = X, y = Y\}$
`begin x := y; y := x einde`
$\{x = Y, y = X\}$.

Deze specificatie is *niet* juist (ga na waarom).

Opdracht 8.13 *Vervang de begin- en eindvoorwaarden in dit voorbeeld door voorwaarden die juist zijn.*

Voorbeeld 8.12 $\{\top\}\ \pi\ \{\psi\}$.
We gebruiken $\top$ voor een conditie die altijd waar is. De specificatie zegt dat ψ waar is als π termineert.

Opdracht 8.14 *Geef aan onder welke voorwaarden de specificatie* $\{\varphi\}\ \pi\ \{\top\}$ *juist is.*

Voorbeeld 8.13

$\{\top\}$
```
    begin
      x := 0;
      zolang x >= 0 doe x := x + 1
    einde
```
$\{x = 100000\}$.

Deze specificatie is correct. Immers, het programma termineert niet, dus de bewering dat ALS het programma termineert de eindwaarde van x gelijk zal zijn aan 100000 is juist. Het voorbeeld maakt duidelijk dat we er voortdurend op bedacht moeten zijn dat de specificaties *partiële correctheidsbeweringen* geven.

Voorbeeld 8.14

$\{x \geq 0\}$
```
   begin
     i := 0;
     p := 0;
     zolang i < x doe
       begin p := p + y; i := i + 1 einde
   einde
```
$\{p = x \cdot y\}$.

De specificatie zegt dat als de waarde van x voordat het programma begint niet negatief is en het programma termineert, de variabele p na afloop het product van x en y zal bevatten. Ga na dat deze specificatie juist is.

Opdracht 8.15 *Is de specificatie nog steeds correct wanneer we de beginvoorwaarde vervangen door* $\{\top\}$*? Waarom (niet)?*

We kunnen de probleemstelling ook omkeren: als een beginvoorwaarde en een eindvoorwaarde gegeven zijn, schrijf dan een programma in **Impertaal** dat aan deze specificatie voldoet. Zie de nu volgende opdrachten.

Opdracht 8.16 *Laat de volgende specificatie gegeven zijn:*

$$\{x > 0\}\ \pi\ \{y = 1 + 2 + \cdots + x\}.$$

Geef een programma π dat gebruik maakt van de variabelen x en y, en dat aan deze specificatie voldoet.

Opdracht 8.17 *Laat de volgende specificatie gegeven zijn:*

$$\{x > 0\}\ \pi\ \{y = 1^3 + 2^3 + \cdots + x^3\}.$$

Geef een programma π dat aan deze specificatie voldoet.

Nu we gezien hebben hoe we predikatenlogica kunnen gebruiken om correctheidsbeweringen te doen over programma's geven we een korte schets van hoe we formeel in de specificatie-logica (ook wel *Floyd–Hoare logica* genoemd naar de uitvinders) kunnen redeneren. De Floyd–Hoare logica voor onze voorbeeldtaal is een axiomatisch systeem met één axioma en een aantal afleidingsregels.

Het axioma heeft betrekking op de toekenningsopdracht, en heet daarom het *toekenningsaxioma*. Het luidt als volgt:

Axioma 8.1 (Toekenningsaxioma) $\{[t/v]\varphi\}\ v := t\ \{\varphi\}.$

Hierbij is v een willekeurige variabele, t een willekeurige term, en φ een willekeurige predikatenlogische formule. De definitie van $[t/v]$ hebt u in opdracht 6.23 zelf gegeven. Enkele voorbeelden:

Voorbeeld 8.15 $\vdash\ \{y = 3\}$ `x := 3` $\{y = x\}$.
Ga na dat deze specificatie een voorbeeld is van een toepassing van het toekenningsaxioma. Elk axioma is een stelling; dit geven we aan met behulp van $\vdash$.

Voorbeeld 8.16 $\vdash\ \{x - 1 = N\}$ `x := x - 1` $\{x = N\}$.
Wat deze specificatie zegt is: als x na de toekenning een waarde N heeft, dan moet de waarde van $x - 1$ voor de toekenning gelijk zijn geweest aan N. Met andere woorden: als x na de toekenning de waarde N heeft, dan moet de waarde daarvoor $N + 1$ zijn geweest.

Wanneer u het toekenningsaxioma niet erg plausibel vindt (omdat de substitutie 'van achter naar voren' plaatsvindt) staat u niet alleen. Echter, $\{\varphi\}\ v := t\ \{[t/v]\varphi\}$ is *niet* correct, zoals uit het volgende voorbeeld blijkt[1]:

[1] Dit voorbeeld is ontleend aan [Gordon 1988].

Voorbeeld 8.17 $\vdash \ \{x = 0\}$ `x := 1` $\{1 = 0\}$
Deze specificatie, die direct volgt uit $\{\varphi\}\ v := t\ \{[t/v]\varphi\}$, is uiteraard onjuist. Het begripsprobleem rond het toekenningsaxioma maakt duidelijk dat de notie *toekennen van een nieuwe waarde aan een variabele* minder simpel is dan op het eerste gezicht lijkt.

De afleidingsregels van Floyd–Hoare logica hebben de volgende vorm:

$$\frac{\begin{array}{l}\vdash S_1 \\ \vdots \\ \vdash S_n\end{array}}{\vdash S.}$$

Wat dit zegt is dat $\vdash S$ kan worden afgeleid uit $\vdash S_1$ tot en met $\vdash S_n$. We geven een aantal voorbeelden.

Afleidingsregel 8.1 (Versterking van de beginvoorwaarde)

$$\frac{\begin{array}{ll}\vdash & \varphi \to \psi \\ \vdash & \{\psi\}\ \pi\ \{\chi\}\end{array}}{\vdash \ \{\varphi\}\ \pi\ \{\chi\}.}$$

In feite staat $\vdash \varphi \to \psi$ hier voor een willekeurige implicatie die een stelling is in een theorie die betrekking heeft op het toepassingsdomein van de programma's (in het geval van **Impertaal**: de theorie van de gehele getallen). Een heel simpel voorbeeld: $\vdash x = N + 1 \to x - 1 = N$. We kunnen de regel bij voorbeeld als volgt gebruiken in een afleiding:

$$\frac{\begin{array}{ll}\vdash & x = N + 1 \to x - 1 = N \\ \vdash & \{x - 1 = N\}\ \texttt{x} := \texttt{x} - \texttt{1}\ \{x = N\}\end{array}}{\vdash \ \{x = N + 1\}\ \texttt{x} := \texttt{x} - \texttt{1}\ \{x = N\}.}$$

Hier is een volgende afleidingsregel:

Afleidingsregel 8.2 (Afzwakking van de eindvoorwaarde)

$$\frac{\begin{array}{ll}\vdash & \{\varphi\}\ \pi\ \{\psi\} \\ \vdash & \psi \to \chi\end{array}}{\vdash \ \{\varphi\}\ \pi\ \{\psi\}.}$$

Voor elke formule ψ hebben we de volgende predikatenlogische stelling: $\vdash \psi \to \top$. Gecombineerd met de zojuist gegeven regel betekent dit dat we voor willekeurige beginvoorwaarde φ en willekeurig programma π af kunnen leiden:

$$\vdash \{\varphi\}\ \pi\ \{\top\}.$$

In het algemeen maken de twee afleidingsregels *versterking van de beginvoorwaarde* en *afzwakking van de eindvoorwaarde* het mogelijk om stellingen over het toepassingsdomein te verweven in correctheidsredeneringen.

De nu volgende regels hebben meer specifiek betrekking op eigenschappen van de **Impertaal** programma's zelf.

Afleidingsregel 8.3 (Opeenvolgingsregel)

$$\frac{\vdash \{\varphi\}\ \pi_1\ \{\psi\} \qquad \vdash \{\psi\}\ \pi_2\ \{\chi\}}{\vdash \{\varphi\}\ \pi_1; \pi_2\ \{\chi\}.}$$

Deze regel geeft aan wat het effect is van het na elkaar uitvoeren van **Impertaal** opdrachten (of: programma's). Herhaald gebruik van de opeenvolgingsregel produceert correctheidsbeweringen voor **Impertaal** opeenvolgingsopdrachten (daarbij worden `begin` en `einde` respectievelijk links en rechts aan de opdrachtenreeks toegevoegd).

Afleidingsregel 8.4 ('Als–dan' regel)

$$\frac{\vdash \{\varphi \wedge \sigma\}\ \pi\ \{\chi\} \qquad \vdash (\varphi \wedge \neg\sigma) \rightarrow \chi}{\vdash \{\varphi\}\ (\textbf{als}\ \ \sigma\ \textbf{dan}\ \pi)\ \{\chi\}.}$$

Deze regel geeft aan wat het effect is van het uitvoeren van een **als dan** opdracht.

Opdracht 8.18 *Gebruik de* **'Als–dan'** *regel om het voorbeeldprogramma* `(als x < 0 dan x := -x)` *van een specificatie te voorzien.*

Afleidingsregel 8.5 ('Als–dan–anders' regel)

$$\frac{\vdash \{\varphi \wedge \sigma\}\ \pi_1\ \{\psi\} \qquad \vdash \{\varphi \wedge \neg\sigma\}\ \pi_2\ \{\psi\}}{\vdash \{\varphi\}\ (\textbf{als}\ \sigma\ \textbf{dan}\ \pi_1\ \textbf{anders}\ \pi_2)\ \{\psi\}.}$$

Opdracht 8.19 *Gebruik de* **'Als–dan–anders'** *regel om het voorbeeldprogramma* `(als x > y dan m := x anders m := y)` *van een specificatie te voorzien.*

Afleidingsregel 8.6 ('Zolang' regel)

$$\frac{\vdash \{\varphi \wedge \sigma\}\ \pi\ \{\varphi\}}{\vdash \{\varphi\}\ \textbf{zolang}\ \sigma\ \textbf{doe}\ \pi\ \{\varphi \wedge \neg\sigma\}.}$$

Voorbeeld 8.18 Als toepassing van de '**Zolang**' regel voorzien we de volgende lus-opdracht van een specificatie:

```
zolang i < x doe i := i + 1
```

Eerst geven we een specificatie voor `i := i + 1` met identieke begin- en eindvoorwaarde. U gelieve zelf na te gaan dat het toekenningsaxioma ons toestaat om het volgende te zeggen:

$\{\top\}$ `i := i + 1` $\{\top\}$.

Dit levert ons de zogenaamde *lusinvariant* die we nodig hebben om de **zolang** regel toe te passen. Een lusinvariant is een predikatenlogische formule waarvan de waarheid behouden blijft bij het uitvoeren van een herhalingslus. In de **zolang** regel is φ de lusinvariant. De volledige specificatie voor de lus-opdracht uit ons voorbeeld volgt nu direct uit de **zolang** regel:

$\{\top \wedge i < x\}$
`zolang i < x doe i := i + 1`
$\{\top \wedge i \geq x\}$.

Voorbeeld 8.19 Beschouw de volgende **zolang** opdracht:

```
zolang niet i = x doe begin k := k + x; i := i + 1 einde
```

Teneinde hiervoor een specificatie te vinden geven we eerst een specificatie voor het `doe` gedeelte. De **'zolang'** regel suggereert dat dat een specificatie moet zijn met de eigenschap dat de beginvoorwaarde identiek is aan de eindvoorwaarde. Zo'n specificatie geeft dan de *lusinvariant* die we moeten hebben. Om de lusinvariant te vinden beginnen we met een geschikte eindvoorwaarde (die we verkrijgen door simpelweg op te schrijven wat het *beoogde* effect is van ons programma), om vervolgens de opdracht van achter naar voren door te lopen en het toekenningsaxioma en de opeenvolgingsregel toe te passen.

Als eindvoorwaarde kiezen we $k = i \cdot x$. Volgens het toekenningsaxioma geldt het volgende:

$\{k = (i+1) \cdot x\}$
`i := i + 1`
$\{k = i \cdot x\}$

We nemen nu de nieuw gevonden beginvoorwaarde als eindvoorwaarde van de voorafgaande toekenningsopdracht. Nogmaals het toekenningsaxioma toepassen levert:

$\{k + x = (i+1) \cdot x\}$
`k := k + x`
$\{k = (i+1) \cdot x\}$

Elementaire algebra leert dat we dit als volgt mogen vereenvoudigen (in feite komt hier een toepassing van de regel voor versterking van de beginvoorwaarde aan te pas):

$\{k = i \cdot x\}$
```
    k := k + x
```
$\{k = (i + 1) \cdot x\}$

Combinatie met behulp van de opeenvolgingsregel levert nu de gewenste lusinvariant:

$\{k = i \cdot x\}$
```
    begin k := k + x; i := i + 1 einde
```
$\{k = i \cdot x\}$

Hier passen we de **'zolang'** regel op toe en we krijgen:

$\{k = i \cdot x \ \wedge \ i \neq x\}$
```
    zolang niet i = x doe begin k := k + x; i := i + 1 einde
```
$\{k = i \cdot x \ \wedge \ i = x\}$

Hiermee is bewezen dat de **zolang** opdracht het kwadraat van een getal x berekent (mits de variabelen k, i en x juist geïnitialiseerd zijn).

Opdracht 8.20 *Voorzie de volgende* **zolang doe** *opdracht van een specificatie:*

```
zolang niet i = x doe begin i := i + 1; f := (i * f) einde
```

Als u het stap voor stap toepassen van de Floyd–Hoare axioma's en regels nogal geestdodend vindt dan bewijst dat dat u geen computer bent. Het inzicht dat regeltoepassingen van het slag dat we zojuist hebben gedemonstreerd geknipt zijn voor de computer leidt tot het idee om correctheidsbewijzen voor imperatieve programma's te gaan *mechaniseren*. We stoppen een geannoteerd programma in de computer en we laten het vervolgens aan de machine over om de condities die met behulp van Floyd–Hoare logica uit de annotaties volgen door te rekenen. De annotaties waarmee we beginnen specificeren wat ons bij het schrijven van het programma voor ogen stond; van die oogmerken kan een computer uiteraard geen weet hebben.

Uiteraard valt over Floyd–Hoare logica veel meer te zeggen dan wij hier gedaan hebben. Voor meer informatie verwijzen we naar [Van Amstel 1984] (voor correctheidsbewijzen met betrekking tot **Pascal**) en [Gordon 1988] (hoofdstukken 1 en 2 geven een goede inleiding in Floyd–Hoare logica; het boek gaat ook in op het mechaniseren van correctheidsbewijzen). Een standaardwerk over correctheidsredeneringen voor imperatieve programma's is [Gries 1983].

8.6 Dynamische logica

De Floyd–Hoare logica uit de vorige paragraaf was tamelijk geschikt voor het bestuderen van de correctheid van **zolang**-programma's. Over een aantal andere zaken kan deze logica echter bijzonder weinig zeggen:

- *Termineert* een gegeven programma?
- Zijn twee verschillende programma's *equivalent*?
- Is een willekeurig programma *deterministisch* of niet?

De laatste vraag heeft betrekking op mogelijke keuzevrijheid binnen een programma (in zekere zin: oude metafysische problemen in een nieuw jasje). Het programma kan zich bij gelijke data verschillend gedragen wanneer het meerdere keren "gedraaid" wordt; in dat geval noemen we het programma *indeterministisch*. Dit kan zich niet voordoen bij **Impertaal**: **zolang**-programma's zijn deterministisch.

PROLOG programma's met een gefixeerde bewijsstrategie (bij voorbeeld 'eerst links en naar beneden') zijn ook deterministisch: het verwerkingsproces ligt dan geheel vast. Wanneer we een PROLOG programma—bij voorbeeld het programma dat de familierelaties binnen ons koningshuis beschrijft—door een procedurele bril beschouwen, en bovendien de PROLOG bewijsstrategie even uitzetten, dan kunnen we zeggen dat zo'n programma indeterministisch is wanneer het opdrachten toelaat die op meer dan één manier kunnen worden uitgevoerd. De vraag 'Is X een zus van Beatrix?' ziet er procedureel uit als een opdracht: 'Noem een zus van Beatrix!' Wanneer er meerdere mogelijke antwoorden zijn ('Margriet', 'Irene', 'Christina') dan noemen we het programma *indeterministisch*. We vergeten dan dus even dat de PROLOG bewijsstrategie in feite vastlegt welk antwoord het eerst wordt gegeven. Op dezelfde manier worden PROLOG predikaten die op meer dan een manier waar kunnen worden gemaakt *indeterministisch* genoemd; de PROLOG predikaten waarbij dat niet zo is heten *deterministisch*. Een en ander maakt duidelijk dat het verband met het metafysische begrip 'indeterminisme' nu ook weer niet zo direct is.

Maar er zijn moderne toepassingen zoals verspreide berekeningen—op een stel aan elkaar verbonden computers die nooit alles van elkaar kunnen weten—die naar hun aard indeterministisch zijn. We zullen ons hier echter beperken tot **Impertaal** programma's, die zoals gezegd altijd deterministisch zijn.

De vraag naar de equivalentie van programma's is met de nodige extra theorievorming nog wel gedeeltelijk binnen de correctheidslogica te beantwoorden, maar de eerste vraag (naar mogelijke terminatie) onttrekt zich

daar geheel aan. De Floyd–Hoare logica is te weinig flexibel om behalve correctheid ook andere wezenlijke kenmerken van het gedrag van programma's te onderzoeken. Daarom presenteren we in deze paragraaf een rijkere logische taal waarin we de programma's zelf kunnen representeren.

De zogenaamde dynamische logica (DL) beoogt de veranderingen te beschrijven die een computerprogramma bewerkstelligt: na het draaien van een programma ziet de wereld er anders uit dan daarvoor. Hierin speelt het begrip *toestand* van de machine weer een belangrijke rol: een programma of opdracht kan de machinetoestand veranderen. Begin- en eindtoestand zullen bij het uitvoeren van een programma dus in de regel verschillen. Met andere woorden, het programma maakt de eindtoestand *bereikbaar* vanuit de begintoestand. Een model voor een programma π bevat dus een verzameling toestanden en een toegankelijkheidsrelatie T_π daartussen. In begin- en eindtoestand zullen uiteraard verschillende formules waar gemaakt worden; vergelijk de begin- en eindvoorwaarden uit de Floyd–Hoare logica.

Dit doet u wellicht denken aan de semantiek van de modale logica (zie § 7.6), en dat was precies de bedoeling. De toestanden zijn in feite wat de werelden waren in de intensionele modellen. We kunnen nu dus spreken over beweringen die *noodzakelijk waar* zijn volgens het programma. 'Noodzakelijk waar' wil hier zeggen: waar in elke toestand die ontstaat als het programma gestopt is. Er is wel een kleine notationele complicatie: bij elk programma π hoort nu een verschillende noodzakelijksoperatie $\Box$. Het vierkantje indiceren als $\Box_\pi$ of $\boxed{\pi}$ wordt echter lastig als het programma een zekere omvang krijgt. Daarom wordt het vierkantje in de regel opengebroken: de notatie wordt dan $[\pi]$. En net zo voor de tegenhanger van 'mogelijkerwijs' in dynamische logica: $\langle\pi\rangle$. De definitie van 'mogelijk volgens π' is als vanouds: $\langle\pi\rangle\varphi = \neg[\pi]\neg\varphi$. Ofwel: in minstens één van de toestandsovergangen die π kan bewerkstelligen is φ waar.

De waarheidsdefinitie van dynamische formules is overeenkomstig de semantiek van de modale logica. ψ is waar in b wordt nu genoteerd als $b \models \psi$, vergelijk de predikatenlogische notaties $V_b(\psi) = 1$ of $\models \psi[b]$. De cruciale waarheidsvoorwaarden van dynamische formules luiden (b en e zijn willekeurige toestanden):

- $b \models [\pi]\varphi \Leftrightarrow$ voor elke e zodat $bT_\pi e$ geldt dat $e \models \varphi$;
- $b \models \langle\pi\rangle\varphi \Leftrightarrow$ er is een e waarvoor $bT_\pi e$ en $e \models \varphi$.

We kunnen nu correctheidsbeweringen binnen de logica zelf doen. Dat is anders dan in de Floyd–Hoare logica die weliswaar predikaatlogische formules gebruikt, maar zelf geen predikaatlogische theorie is: de correctheidsbewering $\vdash \{\varphi\}\pi\{\psi\}$ is een *metalogische* formulering; het kan nog het best beschouwd worden als een speciale gevolgtrekkingsrelatie $\overset{\pi}{\Rightarrow}$ tussen φ en ψ.

Dezelfde bewering kunnen we zonder verdere poespas weergeven in dynamische logica: $\varphi \rightarrow [\pi]\psi$.

Voorbeeld 8.20 Beschouw het volgende programma π:

```
begin
  k := 0;
  j := x;
  zolang j > 0 doe
    begin k := k + x; j:= j - 1 einde
einde
```

Om te zien wat het programma doet kiezen we een bepaalde waarde van x, zeg $x = 3$ en "draaien" het programma waardoor de computer achtereenvolgens de onderstaande serie toestanden (weergegeven door tripels $\langle x, j, k\rangle$) doorloopt:

$$\langle 3, 3, 0\rangle, \langle 3, 2, 3\rangle, \langle 3, 1, 6\rangle, \langle 3, 0, 9\rangle.$$

Het programma telt voor elke aan x gegeven waarde $x - 1$ keer x bij x op, kortom het kwadrateert x, mits x niet negatief is. In de Floyd–Hoare calculus zouden we dit als $\{x \geq 0\}\pi\{k = x^2\}$ noteren; in dynamische logica luidt dezelfde specificatie: $x \geq 0 \rightarrow [\pi](k = x^2)$.

De zojuist geformuleerde correctheidsbewering steunt echter nog steeds op de aanname dat het programma termineert. Immers $[\pi]\varphi$ is waar in een bepaalde toestand als er helemaal geen π-toegankelijke toestanden zijn. Dit is niet het geval voor een formule als $\langle\pi\rangle\varphi$: deze wordt alleen waargemaakt als er een overgang is naar een toestand waarin φ waar is. Dus drukt in het bovenstaande voorbeeld de formule $x \geq 0 \rightarrow \langle\pi\rangle(k = x^2)$ zowel terminatie als correctheid (zogenaamde *totale correctheid*) uit. Meer in het algemeen kunnen we stellen dat de formule $\langle\pi\rangle\top$ *terminatie* van een programma π weergeeft. Kortom:

$\langle\pi\rangle\top$ is waar $\Leftrightarrow$ T_π is voortzettend $\Leftrightarrow$ π termineert.

We zien aan dit voorbeeld eveneens dat $\langle\pi\rangle\varphi$ een sterkere bewering kan zijn dan $[\pi]\varphi$. Het is zelfs zo dat $\langle\pi\rangle\varphi \rightarrow [\pi]\varphi$ voor **Impertaal**-programma's π een *geldige* formule is. Deze (voor modale logica uitzonderlijke) situatie wordt veroorzaakt door het feit dat zulke programma's *deterministisch* zijn: als π termineert dan is de eindtoestand bepaald door de begintoestand. Anders gezegd: vanuit elke toestand is er steeds hoogstens één toestand toegankelijk. Samengevat:

$\langle\pi\rangle\varphi \rightarrow [\pi]\varphi$ is geldig voor elke φ $\Leftrightarrow$ T_π is een partiële functie
$\Leftrightarrow$ π is deterministisch.

Ook over equivalentie van programma's kunnen we nu uitspraken doen. Hier zijn meerdere zienswijzen mogelijk. Een heel stringente eis aan equivalentie van programma's is te eisen dat ze altijd een gelijke interpretatie (dat wil zeggen: toegankelijkheidsrelatie) krijgen.

> $[\pi_1]\varphi$ en $[\pi_2]\varphi$ zijn logisch equivalent voor elke $\varphi \Leftrightarrow T_{\pi_1} = T_{\pi_2}$
> in elk model $\Leftrightarrow$ π_1 en π_2 zijn (sterk) equivalent.

Deze eis kan echter overdreven sterk blijken te zijn, getuige het volgende voorbeeld.

Voorbeeld 8.21 Laat π het programma zijn uit voorbeeld 8.20 dat positieve getallen kwadrateert. We amenderen het programma enigszins om ook negatieve getallen te kwadrateren. Laat daarom π_1 het volgende programma zijn:

```
begin
  (als x < 0 doe x := -x);
  k := 0;
  j := x;
  zolang j > 0 doe
    begin k := k + x; j:= j - 1 einde
einde
```

Omdat π en π_1 voor negatieve x een ander resultaat leveren zullen we deze programma's eigenlijk niet equivalent willen noemen. π_1 is daarentegen intuïtief gesproken wel equivalent met het eenvoudige programma π_2:

```
k := x * x
```

Echter π_1 en π_2 zijn volgens de definitie niet sterk equivalent: ze maken niet dezelfde formules waar; bijvoorbeeld $[\pi_1](j = 0)$ is geldig, maar $[\pi_2](j = 0)$ zeker niet.

Zoals voorbeeld 8.21 duidelijk maakt is het beter het begrip *equivalentie* wat af te zwakken. Voor een willekeurige formule φ stellen we:

> $[\pi_1]\varphi$ en $[\pi_2]\varphi$ zijn logisch equivalent $\Leftrightarrow$ π_1 en π_2 zijn equivalent
> ten opzichte van φ.

We moeten nog een laatste hobbel nemen voordat we daadwerkelijk programma's kunnen interpreteren. De situatie is nu namelijk dat er voor bijna elk programma π een verschillende toegankelijkheidsrelatie T_π moet komen, en hoe kunnen we zo ooit inzicht verwerven in de werking van programma's? Dit probleem is echter makkelijk oplosbaar: we kunnen T_π opbouwen uit deelrelaties die corresponderen met de opdrachten waaruit π bestaat. Voor **Impertaal** gaat dat zo:

- als $\pi =$ **begin** $\pi_1; \cdots ; \pi_n$ **einde**, dan $T_\pi = T_{\pi_1} \bullet \cdots \bullet T_{\pi_n}$.
- als $\pi =$ **(als** φ **dan** α**)**, dan
  $T_\pi = \{\langle b, e\rangle \mid b \models \varphi \Rightarrow \langle b, e\rangle \in T_\alpha \wedge b \not\models \varphi \Rightarrow b = e\}$.
- als $\pi =$ **(als** φ **dan** α **anders** β**)**, dan
  $T_\pi = \{\langle b, e\rangle \mid b \models \varphi \Rightarrow \langle b, e\rangle \in T_\alpha \wedge b \not\models \varphi \Rightarrow \langle b, e\rangle \in T_\beta\}$.
- als $\pi =$ **zolang** φ **doe** α, dan
  $T_\pi = \{\langle b, e\rangle \mid e \not\models \varphi$ en óf $b = e$ óf er bestaan toestanden $c_1, \ldots, c_k$ waarvoor φ waar is, $c_1 = b$ en voor iedere $i : \langle c_i, c_{i+1}\rangle, \langle c_k, e\rangle \in T_\alpha\}$.

Toelichting: de eerste clausule gebruikt de operatie $\bullet$ van *samenstelling van relaties*, die gedefinieerd wordt door:

$$T_\alpha \bullet T_\beta = \{\langle b, e\rangle \mid \text{ er is een } c \text{ zodat } \langle b, c\rangle \in T_\alpha \wedge \langle c, e\rangle \in T_\beta\}.$$

Merk op deze samenstelling een generalisatie is van *compositie* van functies wanneer we functies als relaties opvatten, zij het dat de notatie andersom werkt (dus $f \bullet g = g \circ f$). Merk verder op dat $\bullet$ een *associatieve* operatie is: de volgorde waarin we het "product" berekenen doet er niet toe, we hoeven dus geen haakjes te gebruiken.

De middelste clausules behoeven weinig commentaar, maar de laatste des te meer. De interpretatie van de **zolang**-opdracht dient als volgt gelezen te worden: of φ is onwaar in een begintoestand, α wordt niet toegepast, en de toestand van de automaat blijft onveranderd, of φ is wel waar en α wordt een aantal keren toegepast totdat φ onwaar wordt.

Opdracht 8.21 *Wat gaat er mis als we voor de bovenstaande* **als dan** *clausule de volgende voor de hand liggende definitie hadden gekozen:*

$$T_\pi = \{\langle b, e\rangle \mid b \models \varphi \Rightarrow \langle b, e\rangle \in T_\alpha\}.$$

Overigens zien we bij bovenstaande clausules de wisselwerking tussen logica en programmeertaal: logische formules verschijnen als Boolese uitdrukkingen in de condities van de programmeeropdrachten, en programma's treden op binnen dynamische operatoren. De notatie van uitdrukkingen of formules zal daarom enige variatie vertonen: binnen het programma-deel van de taal wordt gebuik gemaakt van `niet`, `en` en `of`, binnen het logica-deel van $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$. In semantische interpretaties en logische axioma's verdonkeremanen we dit verschil vaak.

Wat bij al deze interpretaties (behalve misschien de eerste) verder in mindere of meerdere mate verwaarloosd wordt is het procesmatige karakter van het draaiende programma. De semantiek van een opdracht geeft een

beschrijving alsof de hele opdracht verwerkt is, en niet hoe het dit stukje bij beetje doet. Voor het resultaat maakt dit natuurlijk niets uit.

We weten nu hoe we T_π kunnen opbouwen uit relaties die aan afzonderlijke opdrachten verbonden zijn, maar nog niet wat de basisstap van deze recursieve definitie is. Hiervoor zijn meerdere mogelijkheden. In de zogenaamde propositionele Dynamische Logica (pDL) werkt men zoals uit de naam blijkt met de taal van de propositielogica verrijkt met een programmeertaal die binnen de modale operatoren kan verschijnen. Omdat er in het puur logische gedeelte van de taal dan alleen met atomaire formules en connectieven en niet met variabelen en kwantificatie gewerkt wordt, ligt het voor de hand iets dergelijks binnen het programmeerdeel te doen. Men kiest dan naar analogie met de verzameling propositieletters een verzameling *atomaire programma's* $a, a', a'', \ldots$ ("programmaletters"). In de semantiek krijgt elk atomair programma a een willekeurige (deterministische) relatie T_a toegekend.

Eerder hebben we opgemerkt dat **Impertaal** deterministisch werkt en gesuggereerd dat het dan ook aanbeveling verdient de semantiek hiernaar in te richten.[2] We dienen nu nog wel te controleren dat de voorgestelde modellen inderdaad deterministisch zijn, dat wil zeggen dat er voor elk programma π en toestand b hoogstens één toestand e kan bestaan zodat $bT_\pi e$, uitgaande van het gegeven dat dit voor atomaire programma's per definitie het geval is. De sleutelvraag wordt nu: bewaren de interpretatieclausules determinisme? Uiteraard schreeuwt deze vraag om een inductief bewijs. We controleren een paar stappen expliciet.

Voorbeeld 8.22

opeenvolging Laat $\pi =$ **begin** $\pi_1; \cdots; \pi_n$ **einde**, en stel dat $T_{\pi_1}, \ldots, T_{\pi_n}$ alle deterministisch zijn. Dan ook $T_\pi = T_{\pi_1} \bullet \cdots \bullet T_{\pi_n}$. Want kies een willekeurige toestand b, dan is er maar hoogstens één b_1 zodat $bT_{\pi_1}b_1$. Vanuit die eventuele b_1 is weer 0 of 1 b_2 met $b_1T_{\pi_2}b_2$, enzovoorts. Dus kan er voor gegeven π en b maar hoogstens 1 T_π-bereikbare toestand b_n zijn.

als–dan Laat $\pi =$ **(als** φ **dan** α**)**, en stel T_α is deterministisch. Kies een willekeurige toestand b. Dan ofwel $b \models \varphi$ en dan is er maximaal 1 T_π-bereikbare e wanneer $\langle b, e\rangle \in T_\alpha$, ofwel $b \not\models \varphi$ en dan is $e = b$ per definitie de enige T_π-bereikbare toestand.

Opdracht 8.22 *Bewijs zelf behoud van determinisme voor de* **als–dan–anders**- *en de* **zolang**-*stap.*

[2] Logisch gezien is een indeterministische semantiek overigens best mogelijk, maar voor een eerste kennismaking lijkt dit didactisch minder gewenst.

Er zijn in de pDL prachtige theoretische resultaten geboekt (zie bijvoorbeeld [Harel 1984] voor het bewijs van volledigheid en beslisbaarheid) maar voor de meeste toepassingen is pDL toch ongeschikt. Er zijn nauwelijks echte programma's te bedenken die adequaat worden weergegeven in pDL; een pDL-formule als

$$[\texttt{zolang p doe a}]q$$

zal weinig realiteitswaarde bezitten omdat er voor echte programma's in de regel wel een koppeling moet zijn tussen de `p` en de `a`; kijk er de **Impertaal** voorbeelden maar eens op na.

Het alternatief is uiteraard om binnen atomaire programma's individuele variabelen en constanten (en eventueel ook functieconstanten) toe te laten. Het gebruik van *termen* (in logische zin) is volstrekt natuurlijk omdat ze ook—soms zelfs onder dezelfde naam—als type in de programmeertaal voorkomen. Een atomaire opdracht is daarmee een *toekenningsopdracht* geworden. Deze combinatie van dynamische operatoren en predikatenlogica wordt wel gekwantificeerde dynamische logica genoemd; wij zullen deze combinatie gewoon aanduiden als 'dynamische logica'.

Logisch betekent een toekenningsopdracht dat er een nieuwe waarde aan een variabele wordt toebedeeld. Het enige wat er bij een toekenning mogelijk in de machinetoestand verandert is dus de *bedeling* die aangepast moet worden opdat de variabele de nieuwe waarde krijgt; precies wat we in § 6.3 deden om de kwantoren te interpreteren. Maar als bij atomaire opdrachten alleen de bedeling verandert, dan geldt dit vanwege de gegeven recursieve opbouw ook voor willekeurige opdrachten en programma's. Om die reden worden de toestanden in het model vanaf hier gezien als bedelingen. De interpretatie van de toekenningsopdracht is dus:

- als π de opdracht $v := t$ is dan $R_\pi = \{\langle b, e\rangle \mid e = b(v|W_b(t))\}$.

Verder brengen we in herinnering dat W_b gedefinieerd wordt door:

- $W_b(u) = b(u)$ voor variabelen u;
- $W_b(c) = I(c)$ voor constanten c;
- $W_b(ft_1 \ldots t_n) = I(f)(W_b(t_1), \ldots, W_b(t_n))$ voor n-plaatsige functieconstanten f.

Voor het gegeven fragment van **Impertaal** betekent dit dat we de rekenkundige bewerkingen ($+$,$-$, $*$) en relaties ($=, <, >, <=, >=$) in het model moeten verdisconteren. Dat kan natuurlijk op allerlei niet bedoelde manieren, maar we beperken ons hier tot zogenaamde *rekenkundige modellen*. In zo'n standaardmodel kiezen we een geschikt rekenkundig domein

(meestal **N**, **Z** of **Q**) en een vaste interpretatie I van getallen (bij voorbeeld $I(\mathbf{10}) = 10$), operaties ($I(*) = \cdot$) en relaties ($I(<=) = \leq$).

Voorbeeld 8.23 Neem als domein van het model de verzameling gehele getallen **Z**. Laat I de standaardinterpretatie zijn. Dan zijn bijvoorbeeld de volgende beweringen waar in dit model:

$(x = 4) \rightarrow \langle$`x := x + 1`$\rangle(x = 5)$
$\langle$`zolang x > 0 doe x := x - 1`$\rangle(x \leq 0)$
$[$`zolang x > 0 doe x := x + 1`$](x \leq 0)$
$\forall x \forall y [$`zolang niet x = y doe x := x + 1`$](x = y)$.

We controleren de eerste bewering. Stel de bewering is niet waar, dan is er een bedeling b zodat $V_b(x = 4) = 1$ en $V_b(\langle$`x := x + 1`$\rangle(x = 5)) = 0$. Uit het eerste volgt $b(x) = 4$. Als $e = b(x|b(x) + 1)$, en dus $e = b(x|W_b(x + 1))$, dan $\langle b, e\rangle \in T_{\mathtt{x:=x+1}}$ en $e(x) = 5$, maar dan ook $V_e(x = 5) = 1$, wat een tegenspraak oplevert.

De waarheid van de tweede bewering is ook eenvoudig na te rekenen (bedelingen die aan x een positieve waarde toekennen worden door de **zolang**-opdracht zo aangepast dat x de waarde 0 krijgt, en bedelingen waarvoor x negatief of 0 is blijven onveranderd).

De derde bewering is minder vanzelfsprekend. Ze lijkt zelfs onwaar als we een bedeling kiezen die aan x een positieve waarde toekent. Maar in dat geval levert de interpretatie van de **zolang**-opdracht geen toegankelijke toestand op, en in het andere geval blijft de bedeling weer onveranderd, en dus is $x \leq 0$ waar in iedere toegankelijke toestand.

Opdracht 8.23 *Bewijs de waarheid van de laatste van de vier beweringen in bovenstaand voorbeeld.*

Hoewel de dynamische logica vooral semantisch is gemotiveerd kunnen we haar ook vanuit axiomatisch gezichtspunt bekijken: welke afleidingsprincipes liggen eraan ten grondslag? Allereerst gebruiken we alle axioma's, afleidingsregels (en daarmee alle stellingen en afgeleide regels) van de predikatenlogica. Daarnaast herhalen we de principes die ook geldig waren in de (zogenaamde normale) modale predikatenlogica die we in dit boek gepresenteerd hebben:

Axioma 8.2 (K, "verdeling") $\vdash [\pi](\varphi \rightarrow \psi) \rightarrow ([\pi]\varphi \rightarrow [\pi]\psi)$.

Afleidingsregel 8.7 (N, "necessitatie") $\vdash \varphi \Rightarrow \vdash [\pi]\varphi$.

Deze principes zijn immers ook semantisch geldig, omdat ze op Kripke modellen altijd waar zijn. Daarnaast zijn er principes die kenmerkend zijn

voor dynamische logica, en voor een deel zelfs afhangen van de gekozen programmeertaal. Voor een gestructureerde, deterministische taal als **Impertaal** geldt zoals gezegd het axioma:

Axioma 8.3 (determinisme) $\vdash \langle\pi\rangle\varphi \to [\pi]\varphi$.

Vergeleken met de Floyd–Hoare calculus is de toekenningsopdracht op zichzelf onproblematisch:

Axioma 8.4 (toekenning) $\vdash [v := t]\varphi \to [t/v]\varphi$.

Overigens zorgt de dynamische toekenningsoperator wel voor een andere complicatie: ze bindt net als kwantoren en de labda-operator variabelen die binnen haar bereik vallen. Wat dat betreft gedraagt $[v := t]$ zich precies als bij voorbeeld $\exists v$. Dat er echt sprake is van binding kunnen we onder meer zien aan het feit dat substitutie binnen het bereik van de toekenningsoperator aan dezelfde gevaren onderhevig is als binnen het bereik van een kwantor. Analoog aan § 6.7 merken we op dat

(16) $\forall y[\mathtt{x} := \mathtt{y} + \mathtt{1}](x \neq y) \to [x/y][\mathtt{x} := \mathtt{y} + \mathtt{1}](x \neq y)$

geen geldige formule is.

Opdracht 8.24 *Voer de substitutie uit en toon vervolgens aan dat formule (16) niet semantisch geldig is.*

Een verschil tussen $[x := t]$ en $\exists x$ is dat een toekenningsopdracht zelf weer variabelen kan introduceren en die voorkomens mogen best vrij zijn. Ter illustratie beschouwen we de ware DL-formule

(17) $\forall x(x \geq 0 \to \langle \mathtt{x} := \mathtt{x} + \mathtt{1}\rangle(x > 0)$.

De universele kwantor bindt hier de voorkomens van x in $x \geq 0$ en in `x + 1`, de toekenning bindt de x in $x > 0$. Dat dit zo kunnen we inzien door herbenoemen van gebonden variabelen. Formule (17) is immers equivalent met:

(18) $\forall x(x \geq 0 \to \langle \mathtt{y} := \mathtt{x} + \mathtt{1}\rangle(y > 0)$.

Merk echter op dat "dubbel gebruik" van variabelen niet te elimineren is wanneer de ophoging $\mathtt{x} := \mathtt{x} + \mathtt{1}$ plaats vindt binnen een **zolang**-opdracht.

Opeenvolging van opdrachten wordt uiteraard als volgt weergeven:

Axioma 8.5 (opeenvolging)

$$\vdash [\textbf{begin } \pi_1; \cdots; \pi_n \textbf{ einde}]\varphi \leftrightarrow [\pi_1] \cdots [\pi_n]\varphi.$$

Opdracht 8.25 *Laat zien dat dit opeenvolgingsprincipe semantisch geldig is.*

De **als-dan-**(**anders**) opdrachten worden beschreven door de volgende axioma's, die onmiddellijk uit de semantische clausules zijn af te lezen:

Axioma 8.6 (als–dan)

$$\vdash [(\textbf{als } \psi \textbf{ dan } \alpha)]\varphi \leftrightarrow ((\psi \rightarrow [\alpha]\varphi) \wedge (\neg\psi \rightarrow \varphi)).$$

Axioma 8.7 (als–dan–anders)

$$\vdash [(\textbf{als } \psi \textbf{ dan } \alpha \textbf{ anders } \beta)]\varphi \leftrightarrow ((\psi \rightarrow [\alpha]\varphi) \wedge (\neg\psi \rightarrow [\beta]\varphi)).$$

Voor **zolang**-opdrachten ligt de zaak minder eenvoudig. Weliswaar is het niet zo moeilijk een geldig postulaat te verzinnen, maar daarmee is de kous nog niet af.

Axioma 8.8 (zolang-iteratie)

$$\begin{array}{l}\vdash [\textbf{zolang } \psi \textbf{ doe } \alpha]\varphi \leftrightarrow \\ \quad ((\psi \rightarrow [\alpha][\ \textbf{zolang } \psi \textbf{ doe } \alpha]\varphi) \wedge (\neg\psi \rightarrow \varphi)).\end{array}$$

Opdracht 8.26 *Laat zien dat het bovenstaande postulaat ook als volgt geformuleerd kan worden:*

$$[\textbf{zolang } \psi \textbf{ doe } \alpha]\varphi \leftrightarrow [(\textbf{als } \psi \textbf{ dan begin } \alpha;\ \textbf{zolang } \psi \textbf{ doe } \alpha \textbf{ einde})]\varphi.$$

Het recursieve effect wordt door dit schema wel verantwoord, maar op deze manier zullen we allerlei andere waarheden zoals een formule met slechts één voorkomen van een **zolang**-opdracht vrijwel nooit kunnen afleiden. We herinneren daarom aan een observatie die eigenlijk al in voorbeeld 8.23 gedaan is:

Axioma 8.9 (deconditionering) $\vdash [\textbf{zolang } \psi \textbf{ doe } \alpha]\neg\psi.$

Onmiddellijk nadat een programma een **zolang**-opdracht gepasseerd is zal de conditie uit die opdracht niet (meer) gelden. Daarnaast willen we ook de beschikking hebben over formules die hun waarheid juist behouden bij het passeren van een **zolang**-opdracht:

Afleidingsregel 8.8 (conservatie)

$$\frac{\vdash \quad \varphi \rightarrow (\psi \rightarrow [\alpha]\varphi)}{\vdash \quad \varphi \rightarrow [\textbf{zolang } \psi \textbf{ doe } \alpha]\varphi.}$$

Om te zien hoe we redeneren in een dergelijk modaal systeem zullen we enige stellingen en regels afleiden. Daarna besteden we in het bijzonder aandacht aan de Hoare-correctheidsregels, die immers ook binnen het ruimere kader van de dynamische logica moeten gelden.

We bewijzen eerst een eenvoudige maar nuttige regel die zelfs in normale modale logica afleidbaar is:

Stelling 8.6 $\vdash \varphi \to \psi \Rightarrow \vdash [\pi]\varphi \to [\pi]\psi$.

Bewijs:

1	$\varphi \to \psi$	[gegeven stelling]
2	$[\pi](\varphi \to \psi)$	[uit 1 met regel N]
3	$[\pi](\varphi \to \psi) \to ([\pi]\varphi \to [\pi]\psi)$	[axioma K]
4	$[\pi]\varphi \to [\pi]\psi$.	[met MP uit 2 en 3]

■

Zonder bewijs vermelden we ook de volgende bekende eigenschap van normale modale logica's:

Stelling 8.7 $\vdash [\pi](\varphi \wedge \psi) \leftrightarrow ([\pi]\varphi \wedge [\pi]\psi)$.

Een resultaat dat de specifieke eigenschappen van de dynamische axiomatiek illustreert is een versterking van het toekenningsaxioma:

Stelling 8.8 $\vdash [v := t]\varphi \leftrightarrow [t/v]\varphi$.

Bewijs:

1	$[v := t]\varphi \to [t/v]\varphi$	[toekenningsaxioma]
2	$[v := t]\neg\varphi \to [t/v]\neg\varphi$	[toekenningsaxioma]
3	$[v := t]\neg\varphi \to \neg[t/v]\varphi$	[uit 2 met substitutie]
4	$[t/v]\varphi \to \neg[v := t]\neg\varphi$	[met contrapositie uit 3]
5	$[t/v]\varphi \to \langle v := t\rangle\varphi$	[met definitie $\langle\pi\rangle$ uit 4]
6	$\langle v := t\rangle\varphi \to [v := t]\varphi$	[determinisme]
7	$[t/v]\varphi \to [v := t]\varphi$	[met propositielogica uit 5 en 6]
8	$[v := t]\varphi \leftrightarrow [t/v]\varphi$.	[met propositielogica uit 1 en 7]

■

Enigszins paradoxaal is stelling 8.8 in het licht van de eerdere opmerking dat de toekenningopdracht variabelen bindt: aan de rechterkant van de equivalentie bij de stelling hoeft helemaal geen sprake te zijn van een gebonden variabele. Enige reflectie leert dat hier geen werkelijke tegenspraak bestaat.

Opdracht 8.27 *Verklaar de bovenstaande schijnbare paradox.*

Toegerust met het bovenstaande kunnen we een resultaat van groter belang voor de dynamische logica aantonen:

Stelling 8.9 *De axiomatiek van Floyd–Hoare is afleidbaar binnen de DL.*

Bewijs: We zullen niet alles uitputtend bewijzen, maar volstaan soms met enige opmerkingen en laten de rest aan de lezer.

- Het *toekenningsaxioma* uit de correctheidslogica krijgt dynamisch de vorm $[t/v]\varphi \to [v := t]\varphi$ en dat is gewoon één kant van de eerder in stelling 8.8 bewezen versterking van het dynamische toekenningsaxioma.
- *Versterking van beginvoorwaarde* is nu zelfs een propositioneel geldige gevolgtrekking.
- *Afzwakking van eindvoorwaarde* leiden we wel even af:

1	$\varphi \to [\pi]\psi$	[gegeven stelling]
2	$\psi \to \chi$	[gegeven stelling]
3	$[\pi]\psi \to [\pi]\chi$	[uit 2 met stelling 8.6]
4	$\varphi \to [\pi]\chi$	[uit 1 en 3 met propositielogica].

- De *opeenvolgingsregel* laten we over aan de lezer.
- De *als–dan regel* en de *als–dan–anders regel* laten zich ook tamelijk rechtstreeks afleiden.
- De *zolang regel* heeft iets meer voeten in de aarde:

1	$(\varphi \wedge \sigma) \to [\pi]\varphi$	[gegeven stelling]
2	$\varphi \to (\sigma \to [\pi]\varphi)$	[uit 1 met propositielogica]
3	$\varphi \to [\textbf{zolang } \sigma \textbf{ doe } \pi]\varphi$	[uit 2 met conservatie]
4	$[\textbf{zolang } \sigma \textbf{ doe } \pi]\neg\sigma$	[deconditionering]
5	$\varphi \to [\textbf{zolang } \sigma \textbf{ doe } \pi]\neg\sigma$	[uit 4 met pL]
6	$\varphi \to ([\textbf{zolang } \sigma \textbf{ doe } \pi]\varphi \wedge$ $[\textbf{zolang } \sigma \textbf{ doe } \pi]\neg\sigma)$	[uit 5 en 6 met pL]
7	$\varphi \to [\textbf{zolang } \sigma \textbf{ doe } \pi](\varphi \wedge \neg\sigma)$	[uit 6 met stelling 8.7].

■

Opdracht 8.28 *Leid de dynamische weergave van de Floyd–Hoare opeenvolgingsregel af; dat wil zeggen: bewijs de volgende regel*

$$\frac{\vdash \varphi \to [\pi_1]\psi \qquad \vdash \psi \to [\pi_2]\chi}{\vdash \varphi \to [\textbf{begin } \pi_1; \pi_2 \textbf{ einde}]\chi.}$$

Over dit axiomastelsel valt nog veel meer te zeggen maar we laten het hierbij. Voor verwante resultaten, ook ten aanzien van een krachtiger (zogenaamd reguliere) programmeertaal verwijzen we wederom naar [Harel 1984]. Een aantal op zichzelf schijnbaar eenvoudige noties zoals iteratie en terminatie blijken dan overigens anders geïnterpreteerd te moeten worden; zie voor deze subtiliteiten ook [Harel 1979].

Floyd–Hoare logica en dynamische logica vormen interessante toepassingen van logische technieken op het gebied van het programmeren. Hoewel de meeste programmeurs de tijd of de formele achtergrond missen om dit heilzame logische middel te gebruiken, zijn wijzelf ten volle overtuigd van het nut van deze toepassingen.

Afgezien van het onmiskenbare belang van de dynamische logica voor de informatica, is de dynamische logica de laatste jaren een belangrijke inspiratiebron geworden voor onderzoekers in de semantiek van natuurlijke talen. Sinds het begin van de jaren 80 zijn er al semantische theorieën opgesteld die uitgaan van een "van links naar rechts" aangroeien van de betekenis van een zin of tekst; zie bij voorbeeld [Kamp 1981] en [Heim 1982]. Het verband met DL wordt echter duidelijker gelegd in [Barwise 1987] en, nog explicieter, in [Groenendijk & Stokhof 1987]. Net zoals een werkend programma een verandering in de machinetoestand teweegbrengt, zorgt de uiting van een zin voor een verandering in de informatie waarover de toehoorder beschikt. Stokhof en Groenendijk trekken de analogie met de semantiek van DL nog verder door: de informatietoestanden—althans de onderdelen daarvan waarop zij zich concentreren—zijn weer *bedelingen*. Voor de taalkundige semantiek zijn bedelingen van groot belang omdat een aantal hardnekkige problemen nu juist te maken hebben met het binden van variabelen. Meer taalkundig: hoe kunnen in bepaalde gevallen pronomina die buiten het bereik van een kwantificerende uitdrukking vallen daar toch door gebonden worden? Beschouw de volgende voorbeelden:

(19) *Er loopt een man op straat. Hij fluit.*

(20) *Iedere boer die een ezel heeft slaat hem.*

[Groenendijk & Stokhof 1987] merken op dat we de anaforische lezingen van (19) en (20) toch eenvoudig in predikatenlogica kunnen weergeven:

(21) $\exists x(Mx \wedge Lx \wedge Fx)$

(22) $\forall x \forall y((Bx \wedge Ey \wedge Hxy) \rightarrow Sxy)$.

Hiermee is de directe correspondentie tussen de opbouw van de tekst respectievelijk de zin enerzijds en de logische representatie anderzijds echter om zeep geholpen. Dat is op zichzelf nog geen ramp, zou u kunnen denken: in § 6.4 hebben we gezien dat bij voorbeeld Russell's analyse van het

bepaald lidwoord ook de zinsstructuur niet volgt. Er is echter een verschil: Russell's descriptie-theorie kan namelijk ook worden weergegeven in typenlogica, en dan bestaat er weer wel een directe correspondentie tussen syntactische structuur en logische representatie. Zo niet bij (21) en (22). In jargon: deze formules zijn niet *compositioneel* verkregen. Een vervelend gevolg daarvan is dat het niet meer duidelijk is hoe we de betekenisrepresentaties automatisch uit de zinnen kunnen verkrijgen. Nu kan men wel een niet-compositionele procedure definiëren die toch bij voorbeeld (22) uit (20) afleidt. Zo'n procedure kan als volgt luiden:

1. Vertaal de zin compositioneel in een formule waarin de anafoor door een vrije variabele, zeg x, wordt weergegeven.

2. Breng de formule in zogenaamde prenex normaalvorm, dat wil zeggen: geef een equivalente formule waarbij alle kwantoren voorop staan.

3. Vervang x door een willekeurige andere variabele.

Het zal echter lastig zijn een dergelijke procedure zo in te perken dat zij niet teveel lezingen genereert. Hoe dan ook, [Groenendijk & Stokhof 1987] komen met de volgende elegante oplossing: stel dat we de zinnen toch weergeven door een compositioneel te verkrijgen representatie, namelijk voor bovenstaande voorbeelden:

(23) $\exists x(Mx \wedge Lx) \wedge Fx$

(24) $\forall x((Bx \wedge \exists y(Ey \wedge Hxy)) \rightarrow Sxy).$

Op het eerste gezicht lijken deze formules de verkeerde betekenissen weer te geven van (19) en (20), namelijk de weinig voor de hand liggende lezingen waarin het pronomen (*hij*, *hem*) vrij (deiktisch) is. Maar nu komt de aap uit de mouw: deze doodgewone formules krijgen een *dynamische* interpretatie, als volgt:

- Niet alleen opdrachten, maar ook logische formules veranderen de waarden van variabelen. In essentie worden conjunctie en concatenatie van zinnen in de semantiek als opeenvolgingsopdrachten geïnterpreteerd.

- Existentiële kwantificatie wordt opgevat als een speciale vorm van een toekenningsopdracht.

De dynamische kijk op de existentiële kwantor wordt aannemelijk als u denkt aan de overeenkomsten tussen het 'bindend effect' van de logische kwantoren en dat van de (modaal geïnterpreteerde) toekenningsopdrachten.

Het gezamelijke effect van de twee dynamiserings-stappen is dat existentiële kwantoren bedelingen 'aanpassen', en dat op zo'n manier dat die aangepaste bedelingen daarna bewaard blijven voor de interpretatie van vrije variabelen buiten het bereik van de kwantor (in onze voorbeelden: voor de interpretatie van de vrije variabele die de anafoor vertegenwoordigt). Voor verder details verwijzen we naar [Groenendijk & Stokhof 1987]. Het is buiten kijf dat er aan deze toepassing van dynamische logica op de semantiek van natuurlijke taal nog druk moet worden gesleuteld om de empirische feiten te verantwoorden. Dat neemt echter niet weg dat we hier een aardige demonstratie hebben van de bruikbaarheid van de 'dynamische kijk' op informatie, en bovendien een illustratie van de manier waarop logica, taalkunde en informatica elkaar positief kunnen beïnvloeden.

Literatuurverwijzingen

[Van Amstel 1984] J.J. van Amstel, *Programmeren: het ontwerpen van algoritmen met Pascal*, Academic Service, Den Haag.

[Apt 1988] K.R. Apt, "Introduction to Logic Programming", Rapport, Centrum voor Wiskunde en Informatica, Amsterdam. Verschijnt in: J. van Leeuwen (ed.), *Handbook of Theoretical Computer Science*, North Holland, Amsterdam.

[Barwise 1987] J. Barwise, 'Noun Phrases, Generalized Quantifiers and Anaphora', in: P. Gärdenfors (ed.), *Generalized Quantifiers: Linguistic and Logical Approaches*, Reidel, Dordrecht.

[Barwise & Cooper 1981] J. Barwise & R. Cooper, 'Generalized Quantifiers and Natural Language', *Linguistics and Philosophy* 4, 159–219.

[Van Benthem 1986] J. van Benthem 1986, *Essays in Logical Semantics*, Reidel, Dordrecht.

[Van Benthem & Ter Meulen 1985] J. van Benthem & A. ter Meulen (eds.), *Generalized Quantifiers*, Foris, Dordrecht.

[Bradko 1986] I. Bradko, *Prolog Programming for Artificial Intelligence*, International Computer Science Series, Addison-Wesley, Wokingham, England etc.

[Clocksin & Mellish 1981] W.F. Clocksin & C.S. Mellish, *Programming in Prolog*, Springer, Berlin etc.

[Van Dalen 1978] D. van Dalen, *Filosofische grondslagen van de wiskunde*, Van Gorcum, Assen.

[Van Dalen 1983] D. van Dalen, *Logic and Structure*, Springer Verlag, Berlin etc., 1983 (tweede druk).

[Van Dalen e.a. 1975] D. van Dalen, H.C. Doets, H.C.M. de Swart, *Verzamelingen; naïef, axiomatisch, toegepast*, Oosthoek, Scheltema & Holkema, Utrecht.

[Dowty 1982] D. Dowty, 'Grammatical Relations and Montague Grammar', in: P. Jacobson & G.K. Pullum (eds.), *The Nature of Syntactic Representation*, Reidel, Dordrecht.

[Dummett 1973] M. Dummett, *Frege, Philosophy of Language*, Duckworth, London.

[Van Eijck 1982] J. van Eijck, *Filosofie: een inleiding*, Boom, Meppel.

[Van Eijck 1987] J. van Eijck, *Programmeren in Turbo Pascal*, Academic Service, Schoonhoven.

[Emmet 1969] E.R. Emmet, *Logisch denken*, Het Spectrum, Utrecht & Antwerpen (Aula 73).

[Enderton 1972] H.B. Enderton, *A Mathematical Introduction to Logic*, Academic Press, New York etc.

[Frege 1879] G. Frege, *Begriffsschrift*, Nebert, Halle.

[Gamut 1982] L.T.F. Gamut, *Logica, taal en betekenis*, Het Spectrum, Utrecht & Antwerpen (2 delen).

[Gordon 1988] M.J.C. Gordon, *Programming Language Theory and its Implementation*, Prentice Hall, New York etc.

[Gries 1983] D. Gries, *The Science of Programming*, Springer Verlag, Berlin etc. (second edition).

[Groenendijk & Stokhof 1987] J. Groenendijk & M. Stokhof, 'Dynamic Predicate Logic: towards a Compositional and Non-Representational Discourse Theory', ongepubliceerd manuscript, ITLI, Universiteit van Amsterdam.

[Harel 1979] D. Harel, *First-Order Dynamic Logic*, Lecture Notes in Computer Science, Volume 68, Springer-Verlag, Berlijn.

[Harel 1984] D. Harel, 'Dynamic Logic', in: D. Gabbay & F. Günthner (eds.), *Handbook of Philosophical Logic, Volume 2*, Reidel, Dordrecht.

[Heim 1982] I. Heim, *The Semantics of Definite and Indefinite Noun Phrases*, dissertatie, University of Massachusetts, Amherst.

[Hofstadter 1979] D.R. Hofstadter, *Gödel, Escher, Bach: an Eternal Golden Braid*, Basic Books, New York.

[Hughes & Becht 1978] P. Hughes & G. Becht, *Vicious Circles and Infinity; an Anthology of Paradoxes*, Penguin Books, Harmondsworth.

[Hughes & Cresswell 1968] G.E. Hughes & M.J. Cresswell, *An Introduction to Modal Logic*, Methuen, London.

[Humphreys 1951] C. Humphreys, *Buddhism*, Penguin, Harmondsworth.

[Jensen & Wirth 1974] K. Jensen & N. Wirth, *Pascal User Manual and Report*, Springer Verlag, New York etc. (Third edition: Revised for the ISO Pascal Standard, 1985).

[Kamp 1981] H. Kamp, 'A Theory of Truth and Semantic Representation'. in: Groenendijk, Janssen & Stokhof (eds.), *Formal Methods in the Study of Language*, volume 1, Mathematical Centre Tracts 135, Amsterdam; herdrukt in Groenendijk, Janssen & Stokhof (eds.), *Truth, Interpretation and Information*, GRASS 2, Foris, Dordrecht 1984.

[Kant 1787] I. Kant, *Kritik der reinen Vernunft*, tweede verbeterde druk, Hartknoch, Riga.

[Lewis 1972] D. Lewis, 'General Semantics', in [Davidson & Harman 1972].

[Montague 1974] *Formal Philosophy, Selected Papers of Richard Montague*, edited by R.H. Thomason, Yale University Press, New Haven & London.

[Nagel & Newman 1975] E. Nagel & J. Newman, *De stelling van Gödel*, Aula, het Spectrum, Utrecht.

[Pereira & Shieber 1987] F.C.N. Pereira & S.M. Shieber, *Prolog and Natural Language Analysis*, CSLI Lecture Notes Number 10, Stanford.

[Rucker 1984] R. Rucker, *Infinity and the Mind*, Paladin Books, London.

[Russell 1905] B. Russell, 'On Denoting', in: *Mind* 14, 479–493.

[Smullyan 1978] R. Smullyan, *What is the Name of this Book?*, Prentice Hall, Englewood Cliffs, New Jersey.

[Smullyan 1982] R. Smullyan, *The Lady or the Tiger?*, Knopf, New York (Nederlandse vertaling: *De prinses of de tijger*, Ambo boeken, Baarn 1983).

[Steels 1983] L. Steels, *Programmeren in LISP*, Academic Service, Den Haag.

[Strawson 1950] P.F. Strawson, 'On Referring', in: *Mind* 59.

[Thijsse 1983] E. Thijsse, 'On some proposed universals of natural language', in: A. ter Meulen (ed.), *Studies in Modeltheoretic Semantics*, Foris, Dordrecht.

[De Vries 1984] G. de Vries, *De ontwikkeling van wetenschap*, Wolters-Noordhoff.

[Winfield 1983] Alan Winfield, *The Complete Forth*, Sigma Technical Press, Wilmslow, Cheshire, UK (in het Nederlands verschenen bij Academic Service, Den Haag, onder de titel *Flitsend Forth*).

[Wirth 1976] N. Wirth, *Algorithms + Data Structures = Programs*, Prentice Hall, Englewood Cliffs, New Jersey.

[Zwarts 1981] F. Zwarts, 'Negatief polaire uitdrukkingen', in: *Glot* 4.

[Zwarts 1986] F. Zwarts, *Categoriale Grammatica en Algebraïsche Semantiek*, dissertatie, Groningen.