Predikaatlogica: semantiek

Hoofdstuk 7

Predikaatlogica: semantiek

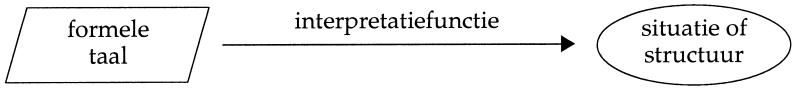
7.1 INLEIDING

De betekenis van een formule in de propositielogica hebben we geformuleerd als het verloop van de waarheidswaarde van die formule door alle waarderingen heen. De waarde van zo'n formule wordt daarbij bepaald door de waarheidswaarden van de atomen die in die formule voorkomen. Dit idee dat de betekenis van een geheel wordt bepaald door de betekenis van zijn delen, gebruiken we ook om de betekenis van een predikaatlogische formule te bepalen. Het staat bekend als *compositionele* semantiek.
De predikaatlogische taal is echter veel uitgebreider en we zullen dus ook een betekenis moeten geven aan predikaatletters, functieletters en formules waarin kwantoren voorkomen. Bovendien kan een formule vrije variabelen en constanten bevatten en ook zo'n formule moet een betekenis krijgen.
Een eenvoudig voorbeeld laat zien wat we onder de (informele) betekenis van een formule kunnen verstaan.

Voorbeeld 7.1

In de formule $\forall x\, \exists y\, (Rxy \wedge Ay)$ komen de predikaatletters R en A voor. Stel dat R staat voor de relatie 'kleiner dan' en A voor 'is een even getal' op het domein der natuurlijke getallen. Rxy betekent dan 'x is kleiner dan y' en Ay 'y is een even getal'. $\forall x\, \exists y\, (Rxy \wedge Ay)$ betekent dan 'bij elk getal is er een groter even getal'.
Maar laat Rxy nu betekenen 'x is de moeder van y' en Ay 'y is een vrouw', waarbij we nu denken aan het domein der mensen. Dan drukt $\forall x\, \exists y\, (Rxy \wedge Ay)$ iets heel anders uit, en wel: 'ieder mens is moeder van een dochter'.

Syntactische objecten uit de formele taal (zoals termen en formules) op zichzelf betekenen nog niets: ze moeten van geval tot geval geïnterpreteerd worden. Bij het bestuderen van semantiek voor de predikaatlogica zijn welbeschouwd drie aspecten in het geding:



De formele taal is in het vorige hoofdstuk beschreven. Met deze taal willen we spreken over doorgaans niet–talige situaties of 'structuren'. Daartoe is echter een koppeling tussen beide nodig, in de vorm van een 'interpretatiefunctie' die betekenis geeft aan de onderdelen van de taal (predikaatletters, functieletters, eigennamen, enzovoort). Hiermee kan dan de betekenis van (termen en) formules bepaald worden, via hun *waarheidswaarde* in een structuur onder een interpretatie.
Dat alle drie de aspecten essentieel zijn, kan men als volgt begrijpen. Denk aan een bericht gesteld in een zekere taal over een bepaalde situatie.
Geval 1: We kennen de syntactische tekst, alsmede de interpretatiefunctie van de taal. Zonder de situatie te kennen, weten we echter niet of het gezegde waar is.
Geval 2: We kennen de syntactische tekst en de situatie, maar niet de interpretatiefunctie. (Denk bijvoorbeeld aan een Turkse krant met een bericht vergezeld van een foto over de beschreven situatie.) Zonder de betekenis van de taal te leren, kunnen we niet bepalen of het gezegde waar is.
Geval 3: We kennen de interpretatiefunctie en de beschreven situatie, maar niet de tekst. Hieruit valt het bericht doorgaans niet te reconstrueren, zodat we over de waarheid ervan geen oordeel kunnen vellen.

7.2 STRUCTUREN

Bij de door een taal beschreven 'situaties' kan men denken aan 'stukjes werkelijkheid', wiskundige ruimtes ($\mathbb{N}$, $\mathbb{R}^2$, de verzameling van alle continue functies, enzovoort), maar ook bijvoorbeeld aan gestructureerde objecten uit de informatica zoals gegevensbestanden. In principe zou men daarbij nog onderscheid kunnen maken tussen concrete situaties, die echt in de fysische werkelijkheid voorkomen (het zonnestelsel, de Tweede Wereldoorlog, het feitelijk rekenproces in een computer) en abstracte situaties die deze eerste 'representeren' (getallenrechten, wiskundig atoommodel, computationele modellen). Wij zullen in dit boek vooral denken aan het laatste genre. In deze paragraaf zullen we een aantal voorbeelden geven van situaties zoals die in de wiskunde kunnen voorkomen. In plaats van situatie zullen we voortaan spreken van *structuur*.

Structuur

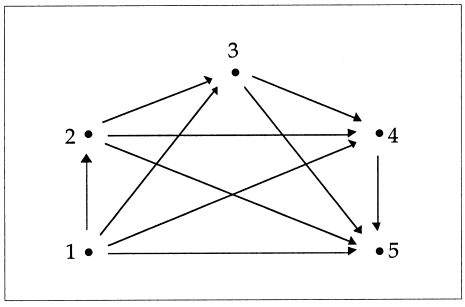
Een structuur bestaat uit een *domein* waarop mogelijk *relaties* of *operaties* gedefinieerd zijn. Informeel kunnen we relaties tussen objecten uit het domein zien als beweringen die wel of niet het geval zijn, terwijl operaties op objecten andere objecten opleveren en dus geen beweringen uitdrukken.

Relationele structuur

Een *relationele* structuur bestaat uit een domein D van objecten met daarop één of meer *relaties*. Voorbeelden van relationele structuren zijn lineaire ordeningen, bomen, grafen, enzovoort. Soms kan een structuur verduidelijkt worden met een plaatje. Zo'n plaatje bevat ten eerste punten, die de objecten van het domein voorstellen. Een éénplaatsige relatie ('eigenschap') P kunnen we weergeven door bij elk punt d een P te zetten als Pd in de structuur geldt. Een tweeplaatsige relatie R kan worden weergeven door een pijl van d_i naar d_j te tekenen als Rd_id_j in de structuur geldt. Het weergeven van drie– of meerplaatsige relaties in een plaatje vereist andere visuele middelen, bijvoorbeeld kleuren, verschillende soorten pijlen, enzovoort. Voor het gemak beperken we ons hier doorgaans tot de unaire en binaire gevallen. We geven enkele concrete structuren.

Voorbeeld 7.2

- Neem $D = \{1, 2, 3, 4, 5\}$ met de binaire (tweeplaatsige) relatie 'kleiner dan', weergegeven door R: $R = \{(1, 2), (1, 3), (1, 4), (1, 5), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 5)\}$. In een plaatje:



- $D = \mathbb{R}^3$, opgevat als de Euclidische ruimte, met de drieplaatsige relatie $Txyz$ ('y ligt tussen x en z') en de vierplaatsige relatie $Exyzu$ ('x ligt even ver van y als z van u').

Structuren kunnen dus zowel *eindig* als *oneindig* zijn. Daarmee is er nu een oneindige klasse van mogelijke structuren, zelfs voor één formule. Ter vergelijking: een formule in de propositielogica heeft slechts een eindig aantal relevante waarderingen.

Operationele structuren

Een andere grote familie binnen de wiskunde wordt gevormd door structuren die uit een domein D bestaan met daarop één of meer *operaties* (functies). Deze worden *operationele* structuren genoemd. In de algebra vindt men veel voorbeelden van operationele structuren (groepen, ringen, Boole–algebra's). Een nulplaatsige operatie heeft geen argumenten. Deze staat daarmee voor een zogenaamd 'uitverkoren object'. Voorbeelden hiervan zijn 0, e, π.

Voorbeeld 7.3

Neem $D = \{1, 2, 3, 4\}$ met daarop gedefinieerd twee verschillende mogelijke interpretaties van een binaire operatie '.', die we weergeven in vermenigvuldigingstabellen (dergelijke verzamelingen heten *groepen*).

'Cyclische groep' C_4

.	1	2	3	4
1	1	2	3	4
2	2	3	4	1
3	3	4	1	2
4	4	1	2	3

'Viergroep van Klein' V_4

.	1	2	3	4
1	1	2	3	4
2	2	1	4	3
3	3	4	1	2
4	4	3	2	1

Ten slotte zijn er natuurlijk situaties waarin zowel relaties als operaties voorkomen.

Voorbeeld 7.4

$\mathbb{N}$ met relaties $<$ en $=$, en operaties $+$ en $\cdot$.

Een structuur wordt nu als volgt gedefinieerd.

DEFINITIE 7.1

Structuur
Een *structuur* $\boldsymbol{D}$ is een drietal $\langle D, \boldsymbol{R}, \boldsymbol{O}\rangle$ bestaande uit een niet–lege verzameling D (het domein), een verzameling $\boldsymbol{R}$ van relaties op D en een verzameling $\boldsymbol{O}$ van operaties op D.

Naamgeving

Hoewel we predikaten ook vaak relaties noemen en operaties ook wel functies, reserveren we de termen predikaat en functie binnen de logica bij voorkeur voor syntactische objecten en niet voor – wat in de volgende paragraaf zal blijken – hun semantische interpretaties.
Om dezelfde reden hebben we de term 'constante' gereserveerd voor syntactische constanten en niet voor hun interpretaties als semantische 'uitverkoren objecten'. Om het talige aspect te benadrukken, spreken we in de predikaatlogische taal bij voorkeur over predikaat*letters* en niet over predikaten en evenzo over functie*letters* in plaats van over functies.

We zullen structuren gebruiken om formules te interpreteren, en wel door een koppeling te maken tussen de predikaatlogische taal en objecten in $\boldsymbol{D}$. In de latere hoofdstukken zullen we overigens zien dat voor natuurlijke talen en programmeertalen het nu ingevoerde structuurbegrip te eenvoudig is.

7.3 DE WAARHEIDSDEFINITIE

Om een predikaatlogische formule te kunnen interpreteren, is er een interpretatie van variabelen, constanten, predikaatletters en functieletters nodig.

DEFINITIE 7.2

Interpretatiefunctie
Laat $\boldsymbol{D} = \langle D, \boldsymbol{R}, \boldsymbol{O}\rangle$ een structuur zijn. Een *interpretatiefunctie* I kent aan elke individuele constante c uit een predikaatlogische taal een speciaal object $I(c) \in \boldsymbol{O}$ toe (speciale objecten vinden we als nulplaatsige operaties terug in de verzameling $\boldsymbol{O}$), aan elke predikaatletter P een relatie $I(P) \in \boldsymbol{R}$ (van dezelfde plaatsigheid) en aan elke functieletter f een operatie $I(f) \in \boldsymbol{O}$ (van dezelfde plaatsigheid).

Model

Een paar $(\boldsymbol{D}, I)$ heet een *model*.
Een (on)eindig model is een model met een (on)eindig domein.

Bij de interpretatiefunctie I kan men denken aan de betrekkelijk stabiele betekenisconventies die wij als kind bijvoorbeeld al leren voor onze eigen moedertaal, of later als student voor wetenschappelijke talen. Bijvoorbeeld, onze ouders leren ons met voorbeelden wat I('blond') of I('vervelend') is in de echte wereld. Daarnaast zijn er woorden die geen vaste interpretatie krijgen, maar meer een 'contextafhankelijke', zoals voornaamwoorden ('zij', 'het') of variabelen in de wiskunde of in programmatekst. Deze worden apart behandeld:

DEFINITIE 7.3

Bedeling
Een *bedeling* b is een functie die aan elke variabele x een object $b(x) \in D$ toekent.

Een bedeling is een soort 'hulpinterpretatie'.

Voorbeeld 7.5

Laat $M = (\langle D, \boldsymbol{R}, \boldsymbol{O}\rangle, I)$ een model zijn, met $D = \mathbb{N}$, $\boldsymbol{R} = \{<\}$, $\boldsymbol{O} = \{0, +, \cdot\}$, en interpretatiefunctie I als volgt: $I(P) = '<'$, $I(f) = '+'$, $I(g) = '\cdot'$, $I(a) = 0$. Laat b een bedeling zijn zodat $b(x_i) = i$ $(i = 1, 2, 3, ...)$. De formule $Px_1x_2 \wedge Pf(a, x_9)g(x_5, x_9)$ wordt dan geïnterpreteerd als: '$1 < 2$ en $9 < 45$'.

Notatie

Een (concrete) structuur als bijvoorbeeld $\langle \mathbb{N}, \{<\}, \{+, \cdot\}\rangle$ schrijven we ook wel korter als $\langle \mathbb{N}, <, +, \cdot\rangle$, wanneer duidelijk is wat de relaties en wat de functies zijn. De bedeling $b[x \mapsto d]$ is de bedeling b waarbij d aan de variabele x wordt toebedeeld, ongeacht wat $b(x)$ voorheen ook was.

Voorbeeld 7.6

Laat b een bedeling zijn met $b(x) = 1$ en $b(y) = 2$. Dan geldt $b[x \mapsto 7](x) = 7$ en $b[x \mapsto 7](y) = 2$.

Met behulp van I en b kunnen we nu definiëren wat in modellen de semantische waarden zijn van variabelen, constanten en termen.

DEFINITIE 7.4

Waardering van termen
Laat $M = (\boldsymbol{D}, I)$ een model zijn en b een bedeling. De semantische waarden van termen zijn als volgt inductief gedefinieerd:

a $V_{M,b}(x) = b(x)$, voor variabelen x
(lees: de interpretatie – waarde, value – van variabele x in model M onder bedeling b is $b(x)$);
b $V_{M,b}(a) = I(a)$, voor individuele constanten a;
c $V_{M,b}(f(t_1, ..., t_k)) = I(f)(V_{M,b}(t_1), ..., V_{M,b}(t_k))$.

Voorbeeld 7.7

Laat M een model zijn met $\boldsymbol{D} = \langle \mathbb{N}, 0, +\rangle$ en $I(f) = '+'$, $I(a) = 0$. Laat verder b een bedeling zijn waarbij $b(x) = 1$. Dan geldt:

$$V_{M,b}(f(a,x)) = I(f)(V_{M,b}(a), V_{M,b}(x)) = I(f)(I(a), b(x)) = +(0, 1) = 1$$

We kunnen nu ook de interpretatie van formules in modellen geven. Net als in de propositielogica is dit een waarheidswaarde 0 of 1. De volgende inductie is de zogenaamde *waarheidsdefinitie* voor de predikaatlogica, die in eindig veel stappen een semantische waarde toekent aan alle formules. Als de waarde 1 is, is het gebruikelijker om $M, b \vDash \varphi$ te schrijven dan om $V_{M,b}(\varphi) = 1$ te schrijven. We volgen dit gebruik.

DEFINITIE 7.5

Waardering van formules
Laat $M = (\boldsymbol{D}, I)$ een model zijn en b een bedeling.
De waarheidswaarden van formules ontstaan als volgt:

a $M, b \vDash P(t_1, ..., t_m) \Leftrightarrow I(P)(V_{M,b}(t_1), ..., V_{M,b}(t_m))$
b $M, b \vDash \neg\varphi \quad \Leftrightarrow M, b \nvDash \varphi$

De overige connectieven gaan op dezelfde manier, met behulp van de eerdere waarheidstabellen. Voor de kwantoren is de uitleg als volgt:

c $M, b \vDash \exists x\, \varphi \quad \Leftrightarrow$ *er is een* $d \in D$ zodat $M, b[x \mapsto d] \vDash \varphi$
d $M, b \vDash \forall x\, \varphi \quad \Leftrightarrow$ *voor alle* $d \in D$ geldt $M, b[x \mapsto d] \vDash \varphi$.

Notatie

$M, b \vDash \varphi$ wordt uitgesproken als 'φ *is waar in* M onder b'. In plaats van $M, b \vDash \varphi$ schrijven we dus soms $V_{M,b}(\varphi) = 1$, en in plaats van $M, b \nvDash \varphi$ soms $V_{M,b}(\varphi) = 0$. In uitdrukkingen $V_{M,b}$ zullen we voor de overzichtelijkheid vaak de index 'M, b' weglaten.
Behalve voor 'waar zijn in' wordt het symbool $\vDash$ ook gebruikt voor 'volgen uit', zie definitie 3.1. In definitie 7.6, hierna, blijkt het verband.

Model in propositielogica en predikaatlogica

In propositielogica gebruikten we het woord model alleen in relatieve zin: 'een waardering V *is model van* een formule φ als $V(\varphi) = 1$'. Zoals we inmiddels gezien hebben, gebruiken we in de predikaatlogica het woord model in absolute zin: 'het paar $(\boldsymbol{D}, I)$ *is een model*'. Het modelbegrip in de predikaatlogica is toch een passende generalisering van dat in de propositielogica, omdat we eveneens blijven zeggen: 'een paar $(\boldsymbol{D}, I)$ is model van een formule φ als voor iedere bedeling b: $V_{(\boldsymbol{D},I),b}(\varphi) = 1$'.

Intuïtie achter de waarheidsdefinitie

De waarheidsdefinitie dient om onze intuïtieve begrippen over wat formules betekenen exact vorm te geven. Het loont de moeite om bij een viertal aspecten van die vormgeving stil te staan.

a Een atomaire formule is waar in een structuur, als het feit dat door die atomaire formule wordt uitgedrukt inderdaad het geval is in die structuur. Wanneer $I(P) = '<'$, $b(x) = 2$ en $b(y) = 7$, dan is de atomaire formule Pxy waar in $\langle \mathbb{N}, < \rangle$ als in die structuur inderdaad geldt dat $2 < 7$.

b De speciale rol van bedelingen op variabelen wordt ten onrechte wel eens louter als een hulptruc beschouwd. In feite betreft het hier een essentieel proces in taalverwerking, getuige analogieën met voornaamwoorden in natuurlijke taal en identifiers in computerprogramma's:

Voornaamwoorden

In de zin 'Voor elke vrouw geldt dat *zij* van wijn houdt' kan 'zij' als variabele beschouwd worden. Om de waarde van deze zin te bepalen, moet de waarde bepaald worden van de zin 'zij houdt van wijn' onder elke bedeling die aan 'zij' een vrouw toekent. Bijvoorbeeld, $V_{b[\text{zij} \mapsto \text{Eva}]}(\text{zij houdt van wijn}) = V(\text{Eva houdt van wijn})$.

Identifiers

In het volgende programma verandert de bedeling voor identifier n telkens:

```
0  BEGIN;
1  n = 0;
2  n = n + 1;
3  IF n > 0 THEN n = n + 1;
4  IF n < 10 THEN GOTO 3 ELSE GOTO 5;
5  END.
```

Zoals we in hoofdstuk 15 nog zullen zien, is imperatief programmeren in wezen 'sturen' van steeds veranderende bedelingen, opgevat als momentane geheugentoestanden van de computer.

c Eén formule φ kan in verschillende structuren heel verschillende beweringen uitdrukken (zie voorbeeld 7.1). Zelfs bij gegeven φ en structuur $\boldsymbol{D}$ kunnen nog verschillende interpretatiefuncties φ op $\boldsymbol{D}$ waar maken:
Bijvoorbeeld $\forall x\, \forall y\, f(x, y) = f(y, x)$ is waar op $\mathbb{Z}$ zowel met $I(f) = '+'$ als met $I(f) = '\cdot'$ en daarentegen onwaar met $I(f) = '-'$. Op dit samenspel tussen φ, $\boldsymbol{D}$ en I komen we later in dit hoofdstuk nog terug.

d De relatie van gelijkheid (of *identiteit*) speelt een aparte rol in de predikaatlogica. Hoewel het gelijkteken al een paar keer gebruikt is in voorbeelden, hebben we het niet als aparte relatie gedefinieerd in de taal van de predikaatlogica. In principe moet echter expliciet gezegd worden of het gelijkteken wel of niet gebruikt mag worden in formules. Zoals we later nog zullen zien, kan deze afspraak bij sommige bewijzen van belang zijn.

Gelijkheid tussen termen

Met betrekking tot gelijkheid tussen termen spreken we een vaste interpretatie af:

$$M, b \vDash t_1 = t_2 \Leftrightarrow V_{M,b}(t_1) = V_{M,b}(t_2)$$

Merk op dat het gelijkteken rechts niet functioneert binnen onze formele taal, maar staat voor de echte identiteit tussen objecten in het domein. Het is uiteraard daarentegen weer mogelijk dat de gelijkheid van twee termen, links, afhangt van gekozen model en bedeling.
Bijvoorbeeld $f(x, g(y, z)) = g(f(x, y), f(x, z))$ is onder elke bedeling waar in $\mathbb{N}$ met $I(f) = '\cdot'$ en $I(g) = '+'$. De gelijkheid luidt dan $x \cdot (y + z) = x \cdot y + x \cdot z$. Dit is niet het geval onder elke bedeling in $\mathbb{N}$ met $I(f) = '+'$ en $I(g) = '\cdot'$. De gelijkheid wordt dan namelijk $x + (y \cdot z) = (x + y) \cdot (x + z)$. Deze laatste gelijkheid is weer wel waar onder elke bedeling die aan x het getal 0 toekent.

Compositionaliteit in de waarheidsdefinitie

We illustreren hoe de compositionaliteit in de waarheidsdefinitie werkt, door in het volgende voorbeeld de stapsgewijze berekening op componenten tot aan het basisniveau uit te voeren.

Voorbeeld 7.8

Laat M een model zijn met $\boldsymbol{D} = \langle \mathbb{Q}, < \rangle$ en $I(R) = '<'$. Laat b een bedeling zijn waarbij $b(x_1) = 4$. We hebben dan:

$M, b \vDash \forall y\,(Rx_1y \rightarrow \exists z\,(Rx_1z \wedge Rzy))$
$\Leftrightarrow$ *voor alle* $q \in \mathbb{Q}$: $M, b[y \mapsto q] \vDash (Rx_1y \rightarrow \exists z\,(Rx_1z \wedge Rzy))$
$\Leftrightarrow$ voor alle $q \in \mathbb{Q}$: *als* $M, b[y \mapsto q] \vDash Rx_1y$,
dan $M, b[y \mapsto q] \vDash \exists z\,(Rx_1z \wedge Rzy)$
$\Leftrightarrow$ voor alle $q \in \mathbb{Q}$: als $4 < q$, dan *is er een* $q' \in \mathbb{Q}$ met
$M, b[y \mapsto q][z \mapsto q'] \vDash (Rx_1z \wedge Rzy)$
$\Leftrightarrow$ voor alle $q \in \mathbb{Q}$: als $4 < q$, dan is er een $q' \in \mathbb{Q}$ met
$M, b[y \mapsto q][z \mapsto q'] \vDash Rx_1z$ *en* $M, b[y \mapsto q][z \mapsto q'] \vDash Rzy$
$\Leftrightarrow$ voor alle $q \in \mathbb{Q}$: als $4 < q$, dan is er een $q' \in \mathbb{Q}$ met $4 < q'$ en $q' < q$

M, b maakt dus deze formule waar.

Eigenschappen van de waarheidsdefinitie

We formuleren nu enkele eenvoudige eigenschappen van de waarheidsdefinitie.

Eindigheid

Als we met behulp van de waarheidsdefinitie de interpretatie van een formule φ willen bepalen, dan hoeven we (naast I) alleen te weten wat een bedeling b doet met de *vrije* variabelen in φ. Hoe de bedeling werkt op gebonden variabelen in φ en op vrije variabelen niet in φ is niet van belang. Dit is als volgt te verwoorden (zonder bewijs):

BEWERING 7.1

Eindigheid
Laten $x_1, \ldots, x_k$ de vrije variabelen van φ zijn, b_1 en b_2 twee bedelingen met $b_1(x_i) = b_2(x_i)$, voor $i = 1, \ldots, k$. Dan geldt: $M, b_1 \vDash \varphi \Leftrightarrow M, b_2 \vDash \varphi$.

Gevolg

Voor *zinnen*, dat wil zeggen 'gesloten formules', formules zonder vrije variabelen, doet de bedeling er niet toe. 'Een zin is waar' is equivalent met 'Hij is waar onder een bedeling', en met 'Hij is waar onder alle bedelingen'. Voor zinnen spreken we dus over waarheid en onwaarheid in een model zonder meer.

Substitutie

De notie 'substitutie' uit hoofdstuk 6 kan nu zowel meer syntactisch als meer semantisch begrepen worden:

BEWERING 7.2

Substitutie
Voor alle termen t, t' geldt:

$V_{M,b}([t/x]t') = V_{M,b[x \mapsto V_{M,b}(t)]}(t')$

Links substitueren we eerst de term t voor x in t' en vervolgens bepalen we de waarde hiervan ('call by name'); rechts rekenen we eerst t' uit en bedelen dan de waarde van t aan x ('call by value'). Het bewijs van deze bewering wordt in hoofdstuk 8 gegeven. Voor een concreet geval laten we alvast zien dat de bewering juist is:

Voorbeeld 7.9

Als $t' = f(x, y)$ en $t = a$ (a is een constante), dan hebben we:

$$
\begin{aligned}
&V_{M,b}([a/x]f(x, y)) \\
&= V_{M,b}(f(a, y)) \\
&= I(f)(V_{M,b}(a), V_{M,b}(y)) \\
&= I(f)(I(a), b(y))
\end{aligned}
$$

Anderzijds:

$$
\begin{aligned}
&V_{M,b[x \mapsto V_{M,b}(a)]}(f(x, y)) \\
&= I(f)(V_{M,b[x \mapsto V_{M,b}(a)]}(x), V_{M,b[x \mapsto V_{M,b}(a)]}(y)) \\
&= I(f)(V_{M,b}(a), b(y)) \\
&= I(f)(I(a), b(y))
\end{aligned}
$$

Dus $V_{M,b}([a/x]f(x, y)) = V_{M,b[x \mapsto V_{M,b}(a)]}(f(x, y))$.

7.4 GELDIG GEVOLG

Een van de centrale begrippen in de propositielogica was 'geldig gevolg'. De volgende definitie legt vast wanneer een gevolgtrekking in de predikaatlogica geldig is.

DEFINITIE 7.6

Geldig gevolg
Laat Σ een verzameling formules zijn en ψ een formule.

$\Sigma \models \psi$ (ψ *volgt uit* Σ) $\Leftrightarrow$ voor elk model M en elke bedeling b geldt:
als $M, b \models \varphi$ voor elke $\varphi \in \Sigma$, dan $M, b \models \psi$.

Omdat er oneindig veel mogelijkheden voor M en b zijn, is geldig gevolg in de predikaatlogica veel complexer dan in de propositielogica. Daarin hoefde immers slechts een eindig aantal waarderingen bekeken te worden om geldigheid van een gevolgtrekking vast te stellen.

Notatie

De notatie $M, b \vDash \Sigma$ betekent '$M, b \vDash \varphi$ *voor elke* $\varphi \in \Sigma$'. Verwar deze notatie voor waarheid in een model niet met de daarop lijkende notatie voor geldig gevolg, zoals bij semantische tableaus. Want de notatie $\Phi \vDash \Sigma$ betekent 'uit Φ volgt ten minste een van de formules uit Σ' (dus niet noodzakelijk alle).

Universeel geldig

Een grensgeval van de definitie van geldig gevolg ontstaat als er geen aannames zijn. Dit geeft $\vDash \psi$. De formule ψ heet dan *universeel geldig*. Universeel geldige formules zijn dus waar in alle modellen M onder iedere bedeling b.
Het vorige in aanmerking genomen betekent de notatie $\vDash \Sigma$ dus: *een van de* formules in Σ is universeel geldig (niet noodzakelijk alle).

Logisch equivalent

Als voor twee formules φ en ψ geldt dat $\vDash \varphi \leftrightarrow \psi$, dan heten φ en ψ *logisch equivalent*.

We onderzoeken beide noties nader in hoofdstuk 9. Nu geven we slechts enkele eenvoudige voorbeelden.

Voorbeeld 7.10

- $\forall x\,(Rx \rightarrow Px), \exists x\,Rx \vDash \exists x\,Px$
- $\vDash Ta \rightarrow \exists x\,Tx$
- $\vDash \forall x\,(x = x)$
- $\vDash \forall x\,Rx \leftrightarrow \neg\exists x\,\neg Rx$
- $\vDash \forall x\,(Rx \wedge Ax) \leftrightarrow (\forall x\,Rx \wedge \forall x\,Ax)$
- $\forall x\,\exists y\,Rxy, \forall x\,\forall y\,(Rxy \rightarrow Ryx), \forall x\,\forall y\,\forall z\,((Rxy \wedge Ryz) \rightarrow Rxz) \vDash \forall x\,Rxx$

7.5 MODELLEN EN AXIOMATIEKEN

Als φ een zin is, dan geldt volgens bewering 7.1 voor *elk* tweetal bedelingen b_1 en b_2: $M, b_1 \vDash \varphi \Leftrightarrow M, b_2 \vDash \varphi$. Dus kunnen we ondubbelzinnig spreken van 'de waarheidswaarde' van een zin in een model. Geldig gevolg voor zinnen kan dan net zo geformuleerd blijven als in hoofdstuk 3: $\Sigma \vDash \varphi$ als elk model van Σ ook een model van φ is. In toepassingen van de predikaatlogica spelen zinnen een belangrijke rol en daarmee ook modellen van zinnen. Een aantal vragen komt daarbij op, afhankelijk van de richting waarin men kijkt.

Van modellen naar theorieën

Gegeven een model M zijn we bijvoorbeeld geïnteresseerd in een effectieve beschrijving van de zinnen die waar zijn in M. De verzameling van die zinnen heet de *theorie* van het model M:

DEFINITIE 7.7

Theorie
$Th(M) = \{\varphi \mid \varphi \text{ is een zin en } M \models \varphi\}$

We kunnen theorieën weergeven door middel van axioma's.

DEFINITIE 7.8

Axiomaverzameling
Een formuleverzameling Σ *axiomatiseert* $Th(M)$ als voor alle zinnen φ geldt: $\varphi \in Th(M) \Leftrightarrow \Sigma \models \varphi$.

We zeggen ook wel dat Σ een *axiomatiek* of *axiomaverzameling* is voor $Th(M)$. Merk op dat $Th(M)$ zichzelf axiomatiseert. Uiteraard zijn we alleen geïnteresseerd in axiomatiseringen die theorieën op een zinvolle en nuttige manier weergeven. Een geslaagde axiomatisering geeft, om zo te zeggen, de essentiële kenmerken van een model kort weer. Axiomatiseringen kunnen overigens zowel eindig als oneindig veel formules bevatten. Oneindig veel formules worden bijvoorbeeld geproduceerd door zogenaamde 'axiomaschema's', zoals het 'Inductieschema' in de taal van de rekenkunde, dat alle instanties omvat van $([0/x]\varphi \wedge \forall x\,(\varphi \rightarrow [Sx/x]\varphi)) \rightarrow \forall x\,\varphi$.

Voorbeeld 7.11

De theorie van het model $(\langle \mathbb{Z}, < \rangle, I)$ met $I(R) = '<'$, wordt geaxiomatiseerd door de volgende formules:
- $\forall x \neg Rxx$ (irreflexiviteit)
- $\forall x \forall y\,(x = y \vee Rxy \vee Ryx)$ (lineariteit)
- $\forall x \forall y \forall z\,((Rxy \wedge Ryz) \rightarrow Rxz)$ (transitiviteit)
- $\forall x \exists y\,(Rxy \wedge \neg\exists z\,(Rxz \wedge Rzy))$ ('onmiddellijke opvolger')
- $\forall x \exists y\,(Ryx \wedge \neg\exists z\,(Ryz \wedge Rzx))$ ('onmiddellijke voorganger').

Een bewijs hiervan valt buiten het bestek van dit boek.

Van zinnen naar modellen

Andersom is het ook mogelijk te beginnen bij een zin of een verzameling zinnen, en de vraag te stellen welke modellen hier nu eigenlijk aan voldoen. We definiëren:

DEFINITIE 7.9

Modelverzameling
a $MOD(\varphi) = \{M \mid M \models \varphi\}$

Evenzo voor een formuleverzameling Σ:

b $MOD(\Sigma) = \{M \mid M \models \varphi \text{ voor alle } \varphi \in \Sigma\}$

Nu is de intuïtie dat we proberen ons een 'beeld' te vormen van de eventuele modellen, door na te gaan wat de diverse eisen in Σ aan structurele condities uitdrukken.

Voorbeeld 7.12

$\Sigma = \{\forall x\, \forall y\, (Rxy \rightarrow \neg Ryx),\ \forall x\, \exists y\, Ryx,\ \forall x\, \forall y\, \forall z\, ((Rxy \wedge Rxz) \rightarrow y = z)\}$.
Om ons een beeld te vormen, beschouwen we eerst kleine eindige modellen. Wanneer we de relatie R eenvoudig interpreteren als een pijl, dan kunnen we de betekenis van de drie eisen achtereenvolgens als volgt visualiseren:

- geen pijl heeft een omgekeerde pijl;
- in elk punt komt een pijl aan;
- vanuit elk punt vertrekt hoogstens één pijl.

Eén object

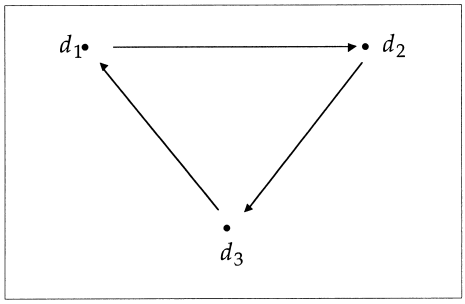
Een model $M = (\boldsymbol{D}, I)$ met één object in het domein, zeg d, kan geen model zijn voor Σ. Volgens $\forall x\, \exists y\, Ryx$ zou op $\boldsymbol{D}$ moeten gelden $I(R)dd$ (dit noemen we een lus in het punt d). Volgens $\forall x\, \forall y\, (Rxy \rightarrow \neg Ryx)$ zou dan op $\boldsymbol{D}$ $I(R)dd$ niet moeten gelden, een tegenspraak. Dus er is geen model met slechts één object.

Twee objecten

Ook een domein met twee objecten, zeg d_1 en d_2, is niet mogelijk. Er moet dan altijd een pijl zijn die een omgekeerde heeft: ófwel een lus in een van de punten óf de pijl van d_1 naar d_2.

Drie objecten

Met drie objecten lukt modellering wél. Een mogelijk model is:



Algemeen

In het algemeen bestaat een model voor Σ uit een verzameling van een of meer structuren van de volgende vormen: een k–cykel ($k \geq 3$), een kopie van de gehele getallen, of een kopie van de negatieve gehele getallen, waarbij in elk punt nog kopieën van deze laatste vorm mogen aankomen.

Voorbeeld 7.13

Het komt ook voor dat plausibel ogende zinnen helemaal geen modellen hebben. Bijvoorbeeld de zin $\exists x\ \forall y\ (Rxy \leftrightarrow \neg Ryy)$ heeft geen modellen. Stel dat $M = (\boldsymbol{D}, I)$ een model is voor deze zin. Dan is er een $d_1 \in D$ zodat op $\boldsymbol{D}$ geldt $I(R)d_1d \Leftrightarrow$ niet $I(R)dd$, voor alle $d \in D$. In het bijzonder geldt dit voor d_1 zelf: $I(R)d_1d_1 \Leftrightarrow$ niet $I(R)d_1d_1$. Dit is echter een contradictie, dus $\exists x\ \forall y\ (Rxy \leftrightarrow \neg Ryy)$ heeft geen model. Deze formule wordt ook wel de Russell–formule genoemd.

Voorbeeld 7.14

Sommige (verzamelingen van) zinnen hebben alleen oneindige modellen. Een bekend voorbeeld is $\Sigma = \{\forall x\ \forall y\ \forall z\ ((Rxy \wedge Ryz) \rightarrow Rxz),$ $\forall x\ \exists y\ Rxy,\ \forall x\ \neg Rxx\}$.
Stel dat $M = (\boldsymbol{D}, I)$ een model hiervoor is. Stel $d_1 \in D$ (D is per definitie niet leeg). Volgens de tweede formule is er dus een $d_2 \in D$ zodat $I(R)d_1d_2$ geldt op $\boldsymbol{D}$. Bovendien is volgens de derde formule $d_1 \neq d_2$. Voor deze d_2 is er, weer volgens de tweede formule, een $d_3 \in D$ zodat $I(R)d_2d_3$, en weer volgens de derde geldt $d_2 \neq d_3$. Volgens de eerste formule geldt dan ook $I(R)d_1d_3$ op $\boldsymbol{D}$, en dus wegens de derde formule, $d_1 \neq d_3$. Zo doorgaand moet er een oneindige stijgende $I(R)$–keten in het domein liggen. Dus ieder model voor Σ is oneindig.

Over het proces van het bepalen van $MOD(\Sigma)$ kan men weer 'dynamisch' denken, net als in hoofdstuk 2. Elke volgende zin in Σ perkt de reeds bereikte modelklasse verder in, tot we het gewenste eindstadium bereiken.

Het algemene semantische schema

We besluiten met een algemeen perspectief. De semantiek van dit hoofdstuk analyseert het begrip 'waarheid' in wezen als een *relatie* tussen vier argumenten:

$WAAR(\varphi, \boldsymbol{D}, I, b)$

Vele vraagstellingen in de logica en de wetenschap, maar ook uit onze dagelijkse redeneerpraktijk, zijn te begrijpen als gevallen waarin we slechts enkele van de argumenten in dit schema kennen en andere juist willen weten. Bij het hiervoor besproken axiomatiseren zoeken we φ als $\boldsymbol{D}$ en I gegeven zijn, bij het zoeken naar modelverzamelingen zoeken we $\boldsymbol{D}$ als φ gegeven is. Ook bijvoorbeeld de typische situatie van het leren van een taal valt in dit kader te beschrijven: gegeven formule φ en structuur $\boldsymbol{D}$, vind de interpretatiefuncties I en bedelingen b die tot waarheid leiden.

Verder zullen we in de semantiek van programmeertalen (zie hoofdstuk 14) ook nog een ander thema tegenkomen, namelijk dat van 'systematische variatie' in de argumenten in het voorgaande schema. Bijvoorbeeld, welke veranderingen in de bedeling ontstaan tijdens programmaverwerking? Binnen de logica is een andere vraag van variatie bekend: wat gebeurt er met de waarheid van formules als we het model veranderen? Voor informatici is ook deze tweede vraag bijvoorbeeld interessant wanneer we willen weten wat er gebeurt met correctheidsbeweringen over programmagedrag bij overgang van de ene datastructuur op een andere.

Stelselmatig onderkennen en benutten van alle mogelijkheden in dit schema '$WAAR(\varphi, \boldsymbol{D}, I, b)$' is een proces dat ook binnen de moderne logica nog lang niet is voltooid.

7.6 OPGAVEN

7.1 Beschouw het model $(\boldsymbol{D}, I)$ met $\boldsymbol{D} = \langle \mathbb{N}, >, priem \rangle$ en $I(R) = >$, $I(P) = priem$, en b is willekeurig. Schrijf de volledige definitie uit van $(\boldsymbol{D}, I), b \vDash \forall x \exists y\ (Ryx \wedge Py)$, en ga na dat uiteindelijk een juiste bewering resulteert.

7.2 Zij $\boldsymbol{D} = \langle \mathbb{R}, \{0, 1, +, \cdot\} \rangle$, met voor de hand liggende I.
a Laat $b(x_i) = \sqrt{i}$, voor alle i. Wat is $V_{(\boldsymbol{D},I),b}(+ \cdot x_3\ x_3 \cdot x_2\ x_8)$?
b Geef een bedeling b zodat $\forall z\ (\cdot\ x_1\ z = z)$ waar wordt.
c Evenzo voor $\exists z\ (\cdot \cdot x_1\ x_2\ z = +\ x_1\ x_2)$.
d Voor welke bedelingen is de volgende formule waar:

$$\exists x\ (+ + \cdot x\ x + x\ y\ 1 = 0)$$

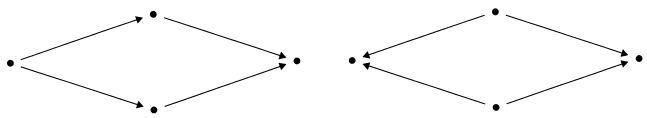
7.3 Zij $\boldsymbol{D} = \langle P(\{1, 2, 3\}), \subseteq \rangle$, $I(R) = '\subseteq'$.
a Geef een bedeling die de volgende formule waar maakt:

$$\forall y\ ((Ryx_1 \wedge Ryx_2) \rightarrow \forall z\, Ryz)$$

b Geef een bedeling die de formule uit a onwaar maakt.
c Geef een andere interpretatie voor R, zodat de volgende formule waar is op $\boldsymbol{D}$:

$$\forall x \neg Rxx \wedge \forall x \forall y\ (Rxy \rightarrow Ryx)$$

7.4 a Geef een zin die waar is op $\langle \mathbb{Q}, < \rangle$, maar niet op $\langle \mathbb{Z}, < \rangle$.
b Geef een zin die waar is op de linkerstructuur, maar niet op de rechter-:



7.5 Zij $\boldsymbol{D} = \langle \mathbb{N}, 0, S \rangle$, met $Sn = n + 1$.
Bewijs dat er slechts één interpretatiefunctie I is voor een drieplaatsig predikaat P dat aan de volgende drie eisen voldoet:
i $\forall x\, \forall y\, \exists! z\, Pxyz$ ($\exists!$ betekent 'precies één'. Zie ook opgave 6.8.)
ii $\forall x\, P0xx$
iii $\forall x\, \forall y\, \forall z\, (Pxyz \rightarrow PSxySz)$

7.6 Beschrijf de modellen van de volgende formule:

$$\forall x\, \forall y\, (Sx = Sy \rightarrow x = y) \wedge \forall x \exists y\, x = Sy$$

7.7 a De weergave van 'Er is een P zodat Q' moet zijn: $\exists x\, (Px \wedge Qx)$, en niet: $\exists x\, (Px \rightarrow Qx)$. Waarom is het tweede fout?
b De weergave van 'Alle P zijn Q' moet zijn: $\forall x\, (Px \rightarrow Qx)$, en niet: $\forall x\, (Px \wedge Qx)$. Waarom is het tweede fout?

7.8 Toon aan dat de volgende Quine–variant van de Russell–formule geen modellen heeft:

$$\exists x\, \forall y\, (Rxy \leftrightarrow \neg R^2yy)$$

met $R^2uv \Leftrightarrow \exists s\, (Rus \wedge Rsv)$.

7.9 Geef voor elke volgende zin een model waarin de betreffende zin niet waar is:
i $\forall x \exists y\, Pxy \rightarrow \exists y\, \forall x\, Pxy$
ii $\forall x\, (Qx \vee Rx) \rightarrow (\forall x\, Qx \vee \forall x\, Rx)$
iii $(\forall x\, Sx \rightarrow \forall x\, Tx) \rightarrow \forall x\, (Sx \rightarrow Tx)$

7.10 Bepaal, indien mogelijk, voor elk van de volgende paren modellen M en N een zin φ, zodat $M \vDash \varphi$ en $N \nvDash \varphi$:

i $M = \langle \mathbb{Z}, <, 0 \rangle$ $N = \langle \mathbb{Z}, \leq, 0 \rangle$

ii $M = \langle \mathbb{Z}, x = y = 2, 0 \rangle$ $N = \langle \mathbb{Z}, x = y = 3, 0 \rangle$

iii $M = \langle \mathbb{R}, <, 0 \rangle$ $N = \langle \mathbb{R}, x < y + 1, 0 \rangle$

iv $M = \langle \mathbb{Q}, <, 0 \rangle$ $N = \langle \mathbb{Q}, x^2 < y^2, 0 \rangle$

v $M = \langle \mathbb{Q}, x + y = 1, 0 \rangle$ $N = \langle \mathbb{Q}, x - y = 1, 0 \rangle$

vi $M = \langle \mathbb{R}, x^2 + y^2 = 1, 0 \rangle$ $N = \langle \mathbb{R}, x^2 - y^2 = 1, 0 \rangle$

Opmerking: in elk van deze modellen komt één tweeplaatsige relatie voor. Gebruik dus voor bijvoorbeeld de relatie '$x = y = 2$' een predikaatletter R, zodat $I(R) = \{(2, 2)\}$, enzovoorts.

Predikaatlogica: eenvoudige theorie

Hoofdstuk 8

Predikaatlogica: eenvoudige theorie

In dit hoofdstuk zullen we enkele meer theoretische aspecten van de predikaatlogica bekijken. De syntaxis van de predikaatlogica is moeilijker dan die van de propositielogica en hetzelfde geldt voor de bijbehorende semantiek.

8.1 SUBSTITUTIE

De belangrijke syntactische notie van substitutie, die in hoofdstuk 6 ingevoerd is, heeft een semantische tegenhanger waarvan het gedrag een zekere 'logische hygiëne' vereist. Om te beginnen geven we nu een precies bewijs van de in het vorige hoofdstuk genoemde methode om voor willekeurige termen t en t' de semantische waarde van de term $[t/x]t'$ te bepalen. Net zoals in hoofdstuk 5 blijkt inductie hier een nuttige methode om de gedachtegang te organiseren.
Merk op dat behalve in gevallen waar van variabelen of kwantoren sprake is, we substitutie recursief door predikaat– en functieletters en logische connectieven heen kunnen halen, dus bijvoorbeeld $V([t/x]\neg\psi) = V(\neg[t/x]\psi)$.

BEWERING 8.1

Substitutie in termen
Voor termen t en t' en variabele x geldt $V_{M,b}([t/x]t') = V_{M,b[x \mapsto V_{M,b}(t)]}(t')$.

Bewijs

Aangezien substitutie in termen inductief definieerbaar is, kunnen we een bijpassend inductief bewijs leveren. Voor de overzichtelijkheid schrijven we in het volgende V in plaats van $V_{M,b}$.

Basisstap

a Als $t' = x$, dan geldt $[t/x]t' = t$, en hebben we $V([t/x]t') = V(t)$. Anderzijds geldt ook $V_{b[x \mapsto V(t)]}(t') = V_{b[x \mapsto V(t)]}(x) = V(t)$. Dus de gewenste gelijkheid gaat op.
b Als $t' = y$, waarbij y een andere variabele is dan x of een constante, dan $[t/x]t' = y$, en hebben we $V([t/x]t') = V(y)$. Maar anderzijds geldt ook $V_{b[x \mapsto V(t)]}(t') = V_{b[x \mapsto V(t)]}(y) = V(y)$.

Inductiestap

De inductiehypothese stelt dat het lemma reeds geldt voor een reeks termen t_i, $i = 1, ..., n$. Dan geldt voor $t' = f^n(t_1, ..., t_n)$ dat $V([t/x]t') = V(f^n([t/x]t_1, ..., [t/x]t_n)) = I(f^n)(V([t/x]t_1), ..., V([t/x]t_n))$ (waarheidsdefinitie), hetgeen volgens de inductiehypothese gelijk is aan $I(f^n)(V_{b[x\mapsto V(t)]}(t_1), ..., V_{b[x\mapsto V(t)]}(t_n))$. Anderzijds geldt ook $V_{b[x\mapsto V(t)]}(t') = V_{b[x\mapsto V(t)]}(f^n(t_1, ..., t_n)) = I(f^n)(V_{b[x\mapsto V(t)]}(t_1), ..., V_{b[x\mapsto V(t)]}(t_n))$. □

Natuurlijk kan substitutie ook plaatsvinden in formules. Hier kwamen we echter eerder het probleem tegen, dat na substitutie op een voorheen vrije positie een variabele gebonden kan raken, waardoor de oorspronkelijke formule wezenlijk verandert. Om dit te voorkomen, definieerden we in hoofdstuk 6 wanneer een term 'vrij' is voor de variabele waarvoor hij gesubstitueerd wordt. Dit leidt nu semantisch tot het gewenste gedrag:

BEWERING 8.2

Substitutie in formules
Als φ een formule is, t een term, x een variabele en t is vrij voor x in φ, dan geldt in elk model M voor elke bedeling b:

$$V_{M,b}([t/x]\varphi) = V_{M,b[x\ \mapsto\ V_{M,b}(t)]}(\varphi)$$

Bewijs

Dit leveren we weer met formule–inductie naar φ. Voor de overzichtelijkheid schrijven we weer V in plaats van $V_{M,b}$.

Basisstap

a $\varphi = P(t_1, ..., t_n)$:

$V([t/x]P(t_1, ..., t_n)) = 1$
$\Leftrightarrow V(P([t/x]t_1, ..., [t/x]t_n)) = 1$
$\Leftrightarrow I(P)(V([t/x]t_1), ..., V([t/x]t_n))$ (waarheidsdefinitie)
$\Leftrightarrow I(P)(V_{b[x\mapsto V(t)]}(t_1), ..., V_{b[x\mapsto V(t)]}(t_n))$ (bewering 8.1)
$\Leftrightarrow V_{b[x\mapsto V(t)]}(P(t_1, ..., t_n)) = 1$ (waarheidsdefinitie)

Inductiestap

b $\varphi = \neg\psi$:

$V([t/x]\neg\psi) = 1$
$\Leftrightarrow V(\neg[t/x]\psi) = 1$
$\Leftrightarrow V([t/x]\psi) = 0$ (waarheidsdefinitie)
$\Leftrightarrow V_{M,b[x\mapsto V(t)]}(\psi) = 0$ (inductiehypothese)
$\Leftrightarrow V_{M,b[x\mapsto V(t)]}(\neg\psi) = 1$ (waarheidsdefinitie)

De overige connectieven zijn op eenzelfde manier te behandelen.

c $\varphi = \forall z\ \psi$:
We moeten twee gevallen nagaan.
1 x is niet vrij in φ.
Dan geldt $[t/x]\varphi = \varphi$ en dus $V([t/x]\varphi) = V(\varphi)$. Omdat x niet vrij voorkomt in φ, geldt volgens bewering 7.1 (eindigheid) $V_{b[x \mapsto V(t)]}(\varphi) = V(\varphi)$. Deze twee gelijkheden leveren inderdaad $V([t/x]\varphi) = V_{b[x \mapsto V(t)]}(\varphi)$.
2 x is vrij in φ.
Dan zal gelden $[t/x]\varphi = \forall z\ ([t/x]\psi)$ en dus $V([t/x]\varphi) = V(\forall z\ ([t/x]\psi))$. Volgens de waarheidsdefinitie geldt: $V(\forall z\ ([t/x]\psi)) = 1 \Leftrightarrow$ voor alle $d \in D\ V_{b[z \mapsto d]}([t/x]\psi) = 1$.
De inductiehypothese vertelt ons hier dat voor ψ de bewering van het lemma reeds opgaat, voor willekeurige substituties van geschikte termen en willekeurige bedelingen. Derhalve geldt:

- $V_{b[z \mapsto d]}([t/x]\psi) = V_{b[z \mapsto d][x \mapsto V'(t)]}(\psi)$, waar $V' = V_{b[z \mapsto d]}$.

Omdat t vrij is voor x in φ, komt z niet in t voor en dus geldt $V'(t) = V_{b[z \mapsto d]}(t) = V(t)$. Bovendien, omdat x vrij is in φ, geldt zeker $x \neq z$. Daardoor geldt $b[z \mapsto d][x \mapsto V(t)] = b[x \mapsto V(t)][z \mapsto d]$. (Dit lijkt triviaal, maar voor $x = z$ hoeft dit niet waar te zijn!) De gelijkheid wordt nu

- $V_{b[z \mapsto d]}([t/x]\psi) = V_{b[x \mapsto V(t)][z \mapsto d]}(\psi)$.

Dit levert dan de volgende keten van equivalenties:

$V([t/x]\ \forall z\ \psi) = 1$	
$\Leftrightarrow V(\forall z\ [t/x]\psi) = 1$	(definitie substitutie)
$\Leftrightarrow$ voor alle $d \in D\ V_{b[z \mapsto d]}([t/x]\psi) = 1$	(waarheidsdefinitie)
$\Leftrightarrow$ voor alle $d \in D\ V_{b[x \mapsto V(t)][z \mapsto d]}(\psi) = 1$	(volgens de zojuist gegeven analyse)
$\Leftrightarrow V_{b[x \mapsto V(t)]}(\forall z\ \psi) = 1$	(waarheidsdefinitie).

Het geval $\varphi = \exists z\ \psi$ gaat op eenzelfde manier. □

8.2 PRENEXVORMEN

Onder de geldige principes van de predikaatlogica bevinden zich enkele principes die ons in staat stellen formules op bepaalde 'normaalvormen' te brengen. Zo is het bijvoorbeeld nuttig het kwantorpatroon van een formule zo expliciet mogelijk zichtbaar te maken.

Verband tussen $\forall$ en $\exists$

Er bestaat een logisch verband tussen de kwantoren $\forall$ en $\exists$. We kunnen met de waarheidsdefinitie laten zien dat een formule van de vorm $\forall x\ \neg\varphi$ dezelfde waarde heeft als de formule $\neg\exists x\ \varphi$ (voor elke M, b):

$V(\forall x\ \neg\varphi) = 1$
$\Leftrightarrow$ voor alle $d \in D$ geldt dat $V_{b[x \mapsto d]}(\neg\varphi) = 1$
$\Leftrightarrow$ voor alle $d \in D$ geldt dat $V_{b[x \mapsto d]}(\varphi) = 0$
$\Leftrightarrow$ er is geen $d \in D$ zodat $V_{b[x \mapsto d]}(\varphi) = 1$
$\Leftrightarrow V(\exists x\ \varphi) = 0$
$\Leftrightarrow V(\neg\exists x\ \varphi) = 1$

Anders gezegd: $\forall x\ \neg\varphi$ en $\neg\exists x\ \varphi$ zijn logisch equivalent. Kennelijk mogen we bij een formule $\neg\exists x\ \varphi$ de kwantor en het negatieteken verwisselen, waarbij de existentiële kwantor verandert in een universele kwantor. Dit geldt ook voor formules van de vorm $\neg\forall x\ \varphi$: deze kunnen we herschrijven tot $\exists x\ \neg\varphi$. Door herhaald toepassen van deze regels kunnen we negatietekens in een formule door de kwantoren heen naar binnen brengen. Bijvoorbeeld, $\neg\neg\neg\exists x\ \forall y\ \varphi$ is logisch equivalent met $\neg\exists x\ \forall y\ \varphi$ (onderdruk een dubbele negatie om redenen van propositielogica) en vervolgens met achtereenvolgens $\forall x\ \neg\forall y\ \varphi$ en $\forall x \exists y\ \neg\varphi$. Onder bepaalde voorwaarden mag een kwantor ook over andere connectieven dan de negatie heen naar voren gehaald worden.

LEMMA 8.1

Als φ en ψ formules zijn en x een variabele is die niet vrij in ψ voorkomt, dan geldt (voor elke M en b) voor $Q = \exists$, $\forall$ en $\Delta = \wedge$, $\vee$: $V((Qx\ \varphi)\ \Delta\ \psi) = V(Qx\ (\varphi\ \Delta\ \psi))$.

Bewijs

Een kenmerkend geval is $\Delta = \wedge$ en $Q = \exists$:
$V((\exists x\ \varphi) \wedge \psi) = 1$
$\Leftrightarrow V(\exists x\ \varphi) = 1$ en $V(\psi) = 1$
$\Leftrightarrow$ (er is een $d \in D$ zodat $V_{b[x \mapsto d]}(\varphi) = 1$) en $V(\psi) = 1$
$\Leftrightarrow$ er is een $d \in D$ zodat $V_{b[x \mapsto d]}(\varphi) = 1$ en $V_{b[x \mapsto d]}(\psi) = 1$
(want x komt niet vrij in ψ voor!)
$\Leftrightarrow$ er is een $d \in D$ zodat $V_{b[x \mapsto d]}(\varphi \wedge \psi) = 1$
$\Leftrightarrow V(\exists x\ (\varphi \wedge \psi)) = 1$
De andere combinaties gaan evenzo. □

Dankzij de symmetrie van $\wedge$ en $\vee$ geldt het omwisselprincipe ook voor kwantoren in de rechterpositie. En dankzij de logische equivalenties $(\varphi \rightarrow \psi) \leftrightarrow (\neg\varphi \vee \psi)$ en $(\varphi \leftrightarrow \psi) \leftrightarrow ((\varphi \wedge \psi) \vee (\neg\varphi \wedge \neg\psi))$ kunnen we ook in formules van de vorm $\varphi \rightarrow \psi$ en $\varphi \leftrightarrow \psi$ de kwantoren naar buiten brengen, door combinatie met de eerdere behandeling van negatie. Let wel, het netto-effect daarvan is dat bijvoorbeeld in een combinatie $(\exists x\ \varphi \rightarrow \psi)$ de existentiële kwantor als een universele naar buiten komt en eenzelfde omklappen vertoont de universele. Vanuit de rechterpositie komen beide echter ongewijzigd naar voren. De conditie in

lemma 8.1 is niet erg beperkend. Indien x toch vrij in ψ voorkomt, dan kunnen we in $Qx\ \varphi$ overgaan op een alfabetische variant met een geschikte nieuwe gebonden variabele.
Deze principes tezamen leiden tot de volgende definitie en stelling.

DEFINITIE 8.1

Prenexvorm
Een formule van de vorm $Q_1x_1 \ldots Q_nx_n\ \psi$, waarbij Q_i $(i = 1, \ldots, n)$ kwantoren zijn en ψ een formule is waarin geen kwantoren meer voorkomen, heet een *prenex*vorm met $Q_1x_1 \ldots Q_nx_n$ als *prefix* en ψ als *matrix*.

STELLING 8.1

Prenexstelling
Voor elke formule φ bestaat er een prenexformule ψ zodat φ en ψ logisch equivalent zijn (dat wil zeggen $V(\varphi) = V(\psi)$).

Bewijs
Basisstap
Inductiestappen

Met inductie naar φ.
a $\varphi = P(t_1, \ldots, t_n)$: φ is reeds in prenexvorm.
b $\varphi = \neg\xi$: volgens de inductiehypothese is ξ te schrijven in prenexvorm, zeg $\xi = Q_1x_1 \ldots Q_nx_n\ \chi$, met χ kwantorvrij. Dan is φ equivalent met $\psi = Q_1'x_1 \ldots Q_n'x_n\ \neg\chi$, waarbij Q_j' de 'omgeklapte' Q_j is (dat wil zeggen $\forall$ wordt $\exists$ en andersom).
c $\varphi = \xi \wedge \chi$: volgens de inductiehypothese zijn ξ en χ te schrijven in prenexvorm, zeg $\xi = Q_1x_1 \ldots Q_nx_n\ \xi'$ en $\chi = K_1y_1 \ldots K_my_m\ \chi'$. Door geschikte alfabetische varianten te kiezen, kunnen we ervoor zorgen dat alle gekwantificeerde variabelen verschillend zijn. Dan is φ op grond van lemma 8.1 equivalent met $\psi = Q_1x_1 \ldots Q_nx_n\ K_1y_1 \ldots K_my_m\ (\xi' \wedge \chi')$.
Het geval $\varphi = \xi \vee \chi$ gaat analoog, en de gevallen voor $\xi \rightarrow \chi$ en $\xi \leftrightarrow \chi$ gaan als in de eerdere bespreking.
d $\varphi = \forall x\ \xi$: volgens de inductiehypothese is ξ in prenexvorm te schrijven, zeg $\xi = Q_1x_1 \ldots Q_nx_n\ \chi$, met χ kwantorvrij. Maar dan is φ equivalent met $\psi = \forall x\ Q_1x_1 \ldots Q_nx_n\ \chi$.
Het geval $\varphi = \exists x\ \xi$ gaat op eenzelfde manier. □

Voorbeeld 8.1

Het inductieve bewijs van de stelling levert ons in feite een effectieve methode om stapsgewijs prenexvormen te maken. Hier is een voorbeeld. Begin met de formule

- $\forall x\ (\forall y\ (Ryx \rightarrow Ay) \rightarrow Ax) \rightarrow \forall x\ Ax$

en werk van binnen uit. De implicatie $\forall y\ (Ryx \rightarrow Ay) \rightarrow Ax$ is equivalent met $\exists y\ ((Ryx \rightarrow Ay) \rightarrow Ax)$. We krijgen zo een vorm waaruit alle drie de kwantoren naar voren kunnen worden gehaald (met de informele notatie $\varphi \Leftrightarrow \psi$ bedoelen we: voor alle modellen M en bedelingen b geldt $M, b \vDash \varphi \Leftrightarrow M, b \vDash \psi$):

$\forall x\, \exists y\, ((Ryx \to Ay) \to Ax) \to \forall x\, Ax$
$\Leftrightarrow \forall x\, (\forall x\, \exists y\, ((Ryx \to Ay) \to Ax) \to Ax)$
$\Leftrightarrow \forall x\, (\forall x'\, \exists y\, ((Ryx' \to Ay) \to Ax') \to Ax)$ (een alfabetische variant)
$\Leftrightarrow \forall x\, \exists x'\, (\exists y\, ((Ryx' \to Ay) \to Ax') \to Ax)$
$\Leftrightarrow \forall x\, \exists x'\, \forall y\, (((Ryx' \to Ay) \to Ax') \to Ax)$.

Merk op dat er keuzen in dit proces zijn. De vorm van het prefix is dus niet eenduidig bepaald. Logici streven vaak naar een 'eenvoudigste' vorm met zo weinig mogelijk afwisselingen van verschillende kwantoren voorop. We hadden bijvoorbeeld de laatste reeks, die nu op twee wisselingen uitkomt, kunnen herschrijven tot een reeks met slechts één wisseling:

- $\exists x'\, \forall y\, \forall x\, (((Ryx' \to Ay) \to Ax') \to Ax)$.

8.3 FRAGMENTEN VAN DE PREDIKAATLOGICA

De volledige taal van de predikaatlogica laat zich opdelen in diverse natuurlijke 'fragmenten' of deeltalen. Een fragment kan vanuit verschillende uitgangspunten gedefinieerd worden. Zo kunnen we in een fragment bijvoorbeeld alleen formules van een bepaalde vorm toelaten, maar we kunnen ook eisen opleggen aan het alfabet. In diverse toepassingsgebieden blijkt men zich vaak te kunnen beperken tot fragmenten, die zich niet zelden ook blijken te onderscheiden door plezierige semantische eigenschappen die niet opgaan voor de taal als geheel.

Monadische logica

Een voorbeeld van zo'n fragment is de *monadische* taal. Hierin zijn alleen eenplaatsige predikaatletters toegestaan. De monadische predikaatlogica is voldoende om de zogenaamde 'syllogismen' uit de traditionele logica te behandelen.

Syllogisme

Informeel kunnen we een *syllogisme* beschrijven als een gevolgtrekking bestaande uit twee aannames en één conclusie, alle drie van de vorm 'alle/geen/sommige A is/zijn B' (alle/geen/sommige objecten met eigenschap A heeft/hebben eigenschap B). In hoofdstuk 1 staat een voorbeeld van zo'n syllogisme, waarvan de predikaatlogische formalisering is:

$\forall x\, (Kx \to Rx)$	('alle kaaimannen zijn reptielen')
$\neg\exists x\, (Rx \wedge Fx)$	('geen reptiel kan fluiten')
dus	
$\neg\exists x\, (Kx \wedge Fx)$	('geen kaaiman kan fluiten')

Meer in het algemeen kunnen we in dit fragment elementaire redeneringen weergeven over verzamelingen of eigenschappen. Eigenschappen van objecten zijn immers eenplaatsige predikaten. Semantisch typerend voor de monadische predikaatlogica met een vast eindig aantal predikaten k is bijvoorbeeld dat het volstaat om *eindige modellen* te beschouwen. Het enige dat bij de evaluatie van zinnen namelijk van belang blijkt, is de al dan niet aanwezigheid van individuen die de 2^k mogelijke combinaties van deze k eigenschappen vertonen.

Universele formules

Een ander nuttig fragment wordt gesuggereerd door de prenexstelling. Ongeveer de eenvoudigste beweringsvorm in de predikaatlogica is die waarbij alle prenexkwantoren van dezelfde soort zijn. Er zijn dan geen 'afhankelijkheden' tussen de diverse kwantoren. Bijvoorbeeld, *universele* formules hebben alleen universele kwantoren in hun prefix. Vele belangrijke wiskundige en computationele eigenschappen hebben zo'n universele vorm (denk bijvoorbeeld aan algebraïsche axioma's of relationele eigenschappen als transitiviteit en lineariteit). Dit fragment zal terugkomen in hoofdstuk 16, omdat het een fundamentele rol speelt in de praktijk van automatisch stellingbewijzen. Een prettige semantische eigenschap van zulke universele formules is de volgende: indien ze waar zijn in een model, dan blijven ze waar wanneer we uit dat model overgaan naar een *deelmodel* met minder individuen (en met daarop de voor de hand liggende beperkte versies van de oorspronkelijke relaties en operaties).

Horn–zinnen

Binnen de universele formules ligt een nog kleiner fragment van de predikaatlogica dat groot belang heeft verworven in 'logic programming' (zie weer hoofdstuk 16) en ook in de theorie van gegevensbestanden:

DEFINITIE 8.2

Horn–zin
Een *Horn–zin* is een universele zin van de vorm

$$\forall x_1 \ldots \forall x_n ((A_1 \wedge \ldots \wedge A_k) \rightarrow B)$$

waarbij $A_1, \ldots, A_k, B$ staan voor atomaire beweringen. Het is ook toegestaan dat het linker– of rechterlid van de implicatie leeg is.

Voorbeeld 8.2

Horn–zinnen zijn bijvoorbeeld

- $\forall x\,(Rx \to Tx)$
- $\forall x\,\forall y\,((Ax \wedge By) \to Cxy)$
- $\forall x\,\forall y\,\forall z\,((Rxy \wedge Ryz) \to Rxz)$
- $\forall x\,Rx$

Geen Horn–zinnen zijn daarentegen

- $\forall x\,\forall y\,(Rxy \vee Ryx \vee x = y)$
- $\forall x\,\exists y\,Sxy$

Dit nog kleinere fragment heeft nog specialere semantische eigenschappen dan de universele formules, zoals het bestaan van zogenaamde *minimale* ('kleinste') modellen. Ook die zullen in hoofdstuk 16 worden uitgelegd.

Juist in toepassingen van de logica binnen de informatica blijkt vaak dat verrassende vondsten niet altijd hoeven te liggen 'in het grote', door logische formalismen steeds verder uit te breiden. Ze kunnen ook heel goed schuilen 'in het kleine' en berusten op de mooie eigenschappen van een goed gekozen fragment.

8.4 OPGAVEN

8.1 Zet de formule $\forall x\,\exists y\,(Rxy \to \exists w\,Rwy) \to \exists y\,\forall x\,(Sxy \to \exists w\,Syw)$ om in prenexvorm.

8.2 Geef een prenexvorm voor de formule $\neg\exists x\,(\exists y\,\forall z\,Rxyz \to \exists z\,\forall y\,Szyx)$, met zo weinig mogelijk afwisselingen van de kwantoren $\forall$ en $\exists$ in de prefix vooraan.

8.3 a Stel dat $\varphi = Q_1x\,Q_2y\,Q_3z\,\psi$ een formule is met elke Q_i een kwantor en ψ kwantorvrij. Als σ een verwisseling (*permutatie*) is van (Q_1, Q_2, Q_3), dan geven we met φ^σ de formule aan die ontstaat uit φ door verwisseling van de kwantoren volgens σ. Bijvoorbeeld, als $\varphi = \exists x\,\forall y\,\exists z\,\psi$, en $\sigma(\exists, \forall, \exists) = (\forall, \exists, \exists)$, dan $\varphi^\sigma = \forall y\,\exists x\,\exists z\,\psi$.
Bepaal alle drietallen (Q_1, Q_2, Q_3) en permutaties σ van (Q_1, Q_2, Q_3) waarvoor geldt: $\varphi \models \varphi^\sigma$.

* b Hoe zit dit in het algemeen voor n–tallen $(Q_1, \ldots, Q_n)$ van kwantoren?

* 8.4 Zet de universele formule $\forall y\,\forall z\,(((Ryc \to Ay) \to Ac) \to Az)$ om in een conjunctie van Horn–zinnen.

* 8.5 Een model $M_1 = (\boldsymbol{D}_1, I_1)$ is een *deelmodel* van een model $M_2 = (\boldsymbol{D}_2, I_2)$, notatie $M_1 \subseteq M_2$, als het volgende geldt:

i $D_1 \subseteq D_2$ (D_1 en D_2 zijn de domeinen van $\boldsymbol{D}_1$ respectievelijk $\boldsymbol{D}_2$)

ii voor elke individuele constante a geldt: $I_1(a) = I_2(a) \in D_1$

iii voor elke predikaatletter P^n geldt: $I_1(P^n)(d_1, ..., d_n) \Leftrightarrow I_2(P^n)(d_1, ..., d_n)$, voor alle $(d_1, ..., d_n) \in (D_1)^n$

iv voor elke functieletter f^n geldt: $I_1(f^n)(d_1, ..., d_n) = I_2(f^n)(d_1, ..., d_n)$, voor alle $(d_1, ..., d_n) \in (D_1)^n$

Voorbeelden:

- $(\mathbb{N}, <, 0, S) \subseteq (\mathbb{R}, <, 0, S)$, met de bekende interpretatie.
- ('even natuurlijke getallen', $< 0) \subseteq (\mathbb{R}, <, 0)$
- niet geldt: $(\mathbb{N}, \leq, 0, S) \subseteq (\mathbb{R}, <, 0, S)$

a Laat M_1 en M_2 modellen zijn zodat $M_1 \subseteq M_2$. Bewijs dat voor elke kwantorvrije φ geldt: $M_2, b \vDash \varphi \Rightarrow M_1, b \vDash \varphi$.

b Bewijs a voor universele zinnen.

c Weerleg de omkering van b.

Predikaatlogica: semantische tableaus

Hoofdstuk 9

Predikaatlogica: semantische tableaus

9.1 INLEIDING

Evenals in de propositielogica kunnen we ook in de predikaatlogica semantische tableaus gebruiken om de geldigheid van een gevolgtrekking te testen. Het hoofdidee van tableaus was het systematisch zoeken naar een tegenvoorbeeld voor $\varphi_1, ..., \varphi_n / \psi$.
Als na een eindig aantal reductiestappen blijkt dat zo'n tegenvoorbeeld niet bestaat, dan geldt:

$\varphi_1, ..., \varphi_n \models \psi$

In de propositielogica bestaat een tegenvoorbeeld voor een gevolgtrekking uit een waardering die $\varphi_1, ..., \varphi_n$ waar maakt en ψ onwaar. Als de gevolgtrekking echter in de predikaatlogica is geformuleerd, dan bestaat een tegenvoorbeeld uit een predikaatlogisch model plus een bedeling die $\varphi_1, ..., \varphi_n$ waar maakt en ψ onwaar. Om zo'n model te vinden, moeten de mogelijkheden om een tableau te maken uitgebreid worden. In vergelijking met de propositielogica zijn nu extra nodig:

- geschikte reductieregels voor de kwantoren $\forall$ en $\exists$;
- het gaandeweg construeren van een domein D;
- het bijhouden van de interpretatiefunctie I en de bedeling b.

We zullen de methode aldus uitbreiden. Hierbij beperken we ons tot een taal zonder functieletters en tot gevolgtrekkingen zonder individuele constanten en zonder vrije variabelen. Voor een taal met functies en constanten is de methode veel ingewikkelder.
De eerdere regels voor connectieven blijven in deze nieuwe situatie natuurlijk onverkort van kracht. Verder hanteren we de verkorte schrijfwijze van hoofdstuk 3 waarin slechts wordt aangegeven wat er *verandert* in een sequent als deze gereduceerd wordt.

9.2 SEMANTISCHE TABLEAUS

Voorbeeld 9.1

We beginnen met een simpel voorbeeld van een geldige gevolgtrekking:

$\forall x\,(Ax \rightarrow Bx), \forall x\,(Bx \rightarrow Cx) \;/\; \forall x\,(Ax \rightarrow Cx)$

Hoe zouden we mogelijke tegenvoorbeelden hiervan systematisch kunnen onderzoeken? Om te beginnen plaatsen we een topsequent als in het propositielogische geval:

$\forall x\,(Ax \rightarrow Bx), \forall x\,(Bx \rightarrow Cx) \circ \forall x\,(Ax \rightarrow Cx)$

De formules links moeten waar worden, die rechts onwaar. Nu levert de linker eis niet direct informatie over wat in het domein moet zitten. Voor de waarheid van universele condities geldt immers 'hoe minder objecten, hoe minder kans dat de regel faalt'. Maar aan de rechterkant liggen de zaken anders: om een universele bewering onwaar te maken, zal het domein minstens één object moeten bevatten dat tegen die bewering zondigt. Laten we zo'n object dus invoeren, zeg d_1, met de passende eis

$\circ\, Ad_1 \rightarrow Cd_1 \qquad\qquad D = \{d_1\}$

Notatie

Merk op dat we hier voor het gemak de individuele constante d_1 uit de predikaatlogische taal identificeren met het object d_1 uit het domein D. Ook hierna zullen we dit vaak blijven doen.

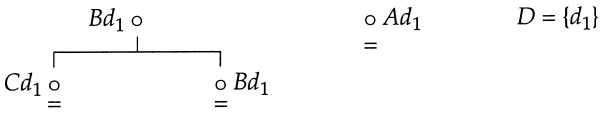
Het domein van het te construeren tegenvoorbeeld bevat dus in ieder geval al een object. De gestelde eis aan dat object kunnen we nog nader analyseren met een propositionele reductie op het implicatieconnectief:

$Ad_1 \circ Cd_1 \qquad\qquad D = \{d_1\}$

Hiermee is de universele bewering aan de rechterkant geheel afgehandeld. Maar, nu er eenmaal een object in het domein is verschenen, ontwaken de universele formules aan de linkerkant. Zij zeggen immers dat in ieder geval dit object aan hun beweringen moet voldoen

$Ad_1 \rightarrow Bd_1, Bd_1 \rightarrow Cd_1 \circ \qquad D = \{d_1\}$

Deze eisen zijn nu verder weer zuiver propositioneel te onderzoeken, hetgeen leidt tot de volgende splitsingen voor implicaties (waar we eigenlijk langs elke tak het domein zouden moeten bijhouden):



Alle drie de resulterende mogelijkheden blijken te sluiten. Er is dus geen tegenvoorbeeld en de gevolgtrekking is geldig, zoals verwacht.

Invoering van reductieregels

Na deze informele uitleg gaan we nu over tot de officiële formulering van de reductieregels in predikaatlogische tableaus. Zoals reeds opgemerkt, zijn om te beginnen alle eerdere regels voor propositionele connectieven toegestaan. Die voor de kwantoren zijn als volgt:

Reductieregels voor $\forall$

Stel dat een sequent de volgende vorm heeft:

Φ o $\forall x\, \varphi$, Ψ

waarbij we voor een tegenvoorbeeld een model $M = (\boldsymbol{D}, I)$ zoeken zodat

$M \models \Phi$, $M \not\models \forall x\, \varphi$ en $M \not\models \Psi$ voor alle $\psi \in \Psi$

In het algemeen kunnen we reeds een deel van het domein van M tijdens de tableauconstructie hebben gevonden, zeg tot en met d_k: $D = \{d_1, d_2, ..., d_k\}$.
Om $\forall x\, \varphi$ onwaar te maken in M, moet er volgens de waarheidsdefinitie minstens één object $d \in D$ zijn zodat $[d/x]\varphi$ onwaar is.

Reductieregel $\forall_R$

We voeren nu een *nieuw* object d_{k+1} in, breiden D daarmee uit, en eisen dat φ voor dat nieuwe object onwaar wordt. In het speciale geval dat er nog niets in D zat, wordt nu een eerste object gekozen:

$\forall_R$: Φ o $\forall x\, \varphi$, Ψ

Φ o $[d_{k+1}/x]\varphi$, Ψ voor een nieuwe d_{k+1}

We kiezen een nieuw object, omdat er geen reden is om aan te nemen dat voor weerlegging van $\forall x\ \varphi$ een der reeds aanwezige objecten zou moeten dienen.

Achtereenvolgende toepassingen van de reductieregel $\forall_R$ zorgen ervoor dat het domein D langzamerhand wordt opgebouwd.

Reductieregel $\forall_L$

De reductieregel voor $\forall_L$ is 'passiever' in dit opzicht. Stel dat de volgende sequent gereduceerd moet worden:

$\Phi,\ \forall x\ \varphi \circ \Psi$

Om $\forall x\ \varphi$ waar te maken in een model $M = (\boldsymbol{D}, I)$, moet $[d/x]\varphi$ waar worden op $\boldsymbol{D}$, voor elke $d \in D$. Dit vereist 'invullen' voor alle objecten die tijdens de constructie in het domein terechtkomen:

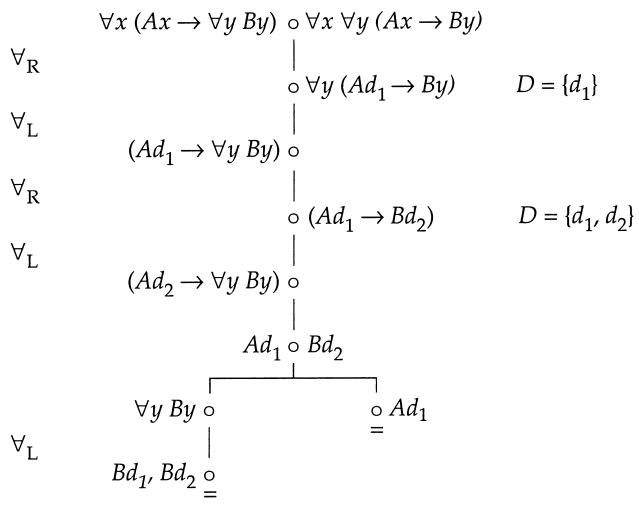
$$\forall_L: \qquad \begin{array}{l} \Phi,\ \forall x\ \varphi \circ \Psi \\ \quad\Big| \\ \Phi,\ [d/x]\varphi \circ \Psi \end{array} \qquad \text{voor alle } d \in D \text{ hier aanwezig}$$

In een ander opzicht is een ware universele kwantor echter juist 'actiever' dan een onware. Het is namelijk goed mogelijk dat verder in het reductieproces het domein D wordt uitgebreid met een nieuw object d_{k+1}. Dan eisen we dat φ ook voor dit nieuwe object opgaat, enzovoorts. Een universele kwantor links is dus in principe nooit 'uitgewerkt'. Dit is wezenlijk verschillend van de situatie met de propositionele logische operatoren. Het weerspiegelt een van de aspecten van kwantificatie die de notie 'semantisch geldig' voor predikaatlogica wezenlijk complexer maakt dan voor propositielogica.
(Omdat de universele formule actief blijft, zou het correcter zijn deze ook in de gereduceerde sequent links van de sequentstip te noteren. Eigenlijk hebben we in de definitie van de regel $\forall_L$ een vereenvoudigde notatie toegepast op de universele formule, zie de opmerking over notatie na voorbeeld 3.5. Iets dergelijks geldt voor de regel $\exists_R$, die hierna nog aan bod komt.)

We geven nog een illustratie van deze twee regels, aan de hand van een van de prenexequivalenties van hoofdstuk 8.

Voorbeeld 9.2

Een tableau voor $\forall x\ (Ax \rightarrow \forall y\ By)\ /\ \forall x\ \forall y\ (Ax \rightarrow By)$:



Dit tableau sluit, dus de gevolgtrekking is geldig.

Notatie

Voor de overzichtelijkheid geven we toepassingen van reductieregels voor kwantoren wél, maar van propositionele reductieregels níet aan in tableaus (evenals in hoofdstuk 3).

Reductieregels $\exists_L$ en $\exists_R$

De reductieregels voor $\exists$ vertonen veel overeenkomst met de regels voor $\forall$, met een omkering van rollen links en rechts. Dit valt te begrijpen omdat $\forall$ zich tot $\exists$ verhoudt op een analoge wijze als $\wedge$ zich tot $\vee$ verhoudt, samen met het feit dat in propositielogische tableaus conjunctie en disjunctie reeds een duaal gedrag vertoonden voor wat betreft splitsing.

$\exists_L$:

$$\frac{\Phi, \exists x\, \varphi \circ \Psi}{\Phi, [d_{k+1}/x]\varphi \circ \Psi}$$

voor een nieuwe d_{k+1} in het domein

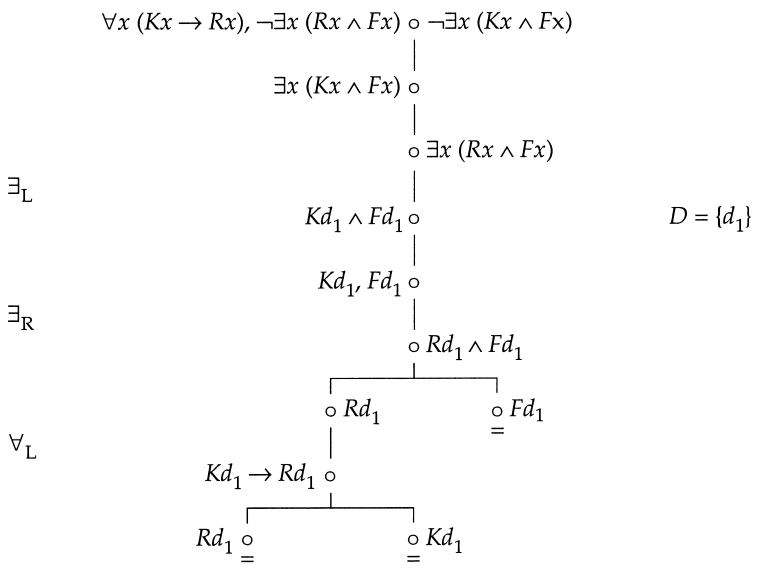
$\exists_R$:

$$\frac{\Phi \circ \exists x\, \varphi, \Psi}{\Phi \circ [d/x]\varphi, \Psi}$$

voor alle $d \in D$ hier aanwezig

Voorbeeld 9.3

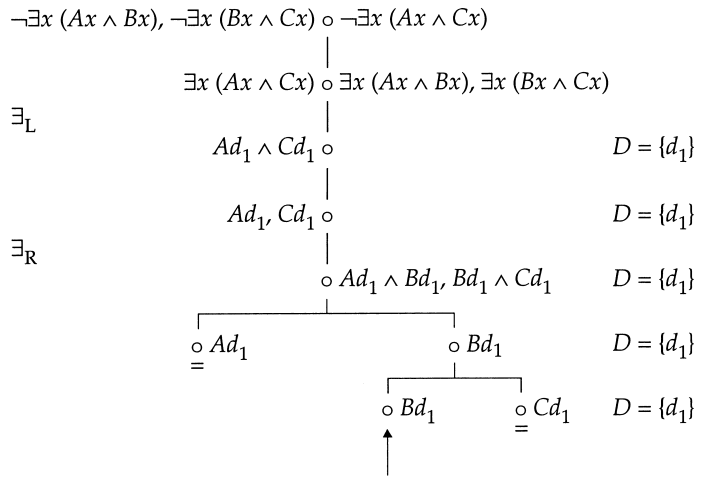
Het eerdere syllogisme over kaaimannen, reptielen en fluiters uit de hoofdstukken 1 en 8 is als volgt te testen:



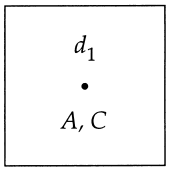
Ook hier sluit iedere tak en daarmee het gehele tableau. Er zijn dus geen tegenvoorbeelden, zodat van geldigheid sprake is.

Voorbeeld 9.4

Vervolgens beschouwen we nu een geval van ongeldigheid, teneinde de constructie van tegenvoorbeelden uit tableaus te demonstreren. Daartoe kiezen we het syllogisme: 'geen A is B, geen B is C; dus geen A is C'. Het tableau ziet er nu zo uit:



Dit tableau houdt één open tak (aangegeven met de pijl). Deze tak levert een model met één element $d_1 \in D$ waarin Ad_1 en Cd_1 waar zijn en Bd_1 onwaar (onder dezelfde afleesconventie op de tak als vroeger in de propositielogica). In een plaatje:



Preciezer: $D = \{d_1\}$, $I(A) = I(C) = \{d_1\}$, $I(B)$ is leeg.
In dit model zijn inderdaad $\neg\exists x\,(Ax \wedge Bx)$ en $\neg\exists x\,(Bx \wedge Cx)$ waar en $\neg\exists x\,(Ax \wedge Cx)$ onwaar. Door dit tegenvoorbeeld is de gegeven gevolgtrekking niet geldig.

We beschouwen nu gevallen waarin meerplaatsige predikaten optreden.

Voorbeeld 9.5

We testen de volgende wisseling van kwantoren op geldigheid:
$\exists x\ \forall y\ Rxy \;/\; \forall y\ \exists x\ Rxy$.

	$\exists x\ \forall y\ Rxy$ o $\forall y\ \exists x\ Rxy$	
$\exists_L$	|	
	$\forall y\ Rd_1y$ o	$D = \{d_1\}$
$\forall_L$	|	
	Rd_1d_1 o	$D = \{d_1\}$
$\forall_R$	|	
	o $\exists x\ Rxd_2$	$D = \{d_1, d_2\}$
$\forall_L$	|	
	Rd_1d_2 o	$D = \{d_1, d_2\}$
$\exists_R$	|	
	o Rd_1d_2, Rd_2d_2	$D = \{d_1, d_2\}$
	=	

Het tableau heeft dus maar één tak en die sluit. De gevolgtrekking is dus geldig.

Oneindige tak

Met binaire predikaten kan zich ook voor het eerst een verschijnsel gaan voordoen dat we in de propositielogica nooit ontmoeten, te weten een *oneindig* voortlopende tak in een tableau.

Voorbeeld 9.6

Een andere wisseling van kwantoren wordt gegeven in de sequent: $\forall y\, \exists x\, Rxy \circ \exists x\, \forall y\, Rxy$.
Hier stuiten we om te beginnen op een klein probleem: noch de linker-, noch de rechterformule is te reduceren, omdat er niets in het domein is. Omdat een leeg domein in onze modellen echter niet is toegestaan, geven we een object als 'voorgift'. Uitgaande van $D = \{d_1\}$ kunnen we dan als volgt beginnen:

$$\begin{array}{lll}
 & \forall y\, \exists x\, Rxy \circ \exists x\, \forall y\, Rxy & D = \{d_1\} \\
\forall_L, \exists_R & \Big| & \\
 & \exists x\, Rxd_1 \circ \forall y\, Rd_1y & D = \{d_1\} \\
\exists_L & \Big| & \\
 & Rd_2d_1 \circ \forall y\, Rd_1y & D = \{d_1, d_2\} \\
\forall_R & \Big| & \\
 & Rd_2d_1 \circ Rd_1d_3 & D = \{d_1, d_2, d_3\} \\
\forall_L, \exists_R & \Big| & \\
 & Rd_2d_1, \exists x\, Rxd_2, \exists x\, Rxd_3 \circ Rd_1d_3, \forall y\, Rd_2y, \forall y\, Rd_3y & D = \{d_1, d_2, d_3\}
\end{array}$$

Het zal duidelijk zijn wat hier gebeurt. Na het creëren van nieuwe objecten moeten we nieuwe formules toevoegen op grond van de zich herhalende regels $\forall_L$ en $\exists_R$. Dat leidt daarop weer tot creëren van nieuwe objecten, enzovoorts. Aldus ontstaat een oneindig tableau met een oneindige tak. In hoofdstuk 11 zullen we bewijzen dat zo'n oneindige tak een tegenvoorbeeld oplevert. Zo'n tegenvoorbeeld is een model met een oneindig domein. Bijvoorbeeld, in het voorgaande wordt een structuur van kopieën van de natuurlijke getallen gecreëerd met een interpretatie die onze kwantorwissel weerlegt. Een mogelijk tegenvoorbeeld zou zijn $I(R) = $ '$>$' op $\mathbb{N}$.

Naast eindig sluiten en eindig open blijven, hebben we nu dus een derde mogelijkheid gevonden waarin semantische tableaus voor de predikaatlogica terecht kunnen komen: non–terminatie in minstens één tak.

9.3 EEN VERFIJNING VAN DE METHODE

Voorbeeld 9.6 is nog niet helemaal doorslaggevend, omdat er ook een *eindig* tegenvoorbeeld bestaat voor zijn kwantorwissel. Om dat te vinden, dienen de regels $\exists_L$ en $\forall_R$ echter enigszins te worden aangepast.

Stel dat er, bij het maken van een tableau, op een bepaald moment in een sequent rechts van $\circ$ een formule van de vorm $\forall x\, \varphi$ staat. Om een tegenvoorbeeld voor deze sequent te vinden, voeren we niet meteen een nieuw object in, zoals we tot nu toe deden. We kijken nu eerst of er al een d_i in het domein aanwezig is waarvoor $[d_i/x]\varphi$ onwaar gemaakt zou kunnen worden. Dit kunnen we in de volgorde van eerdere invoering van de objecten doen: we proberen eerst of $[d_1/x]\varphi$ een tegenvoorbeeld levert en stoppen als dit inderdaad zo is. Zo niet, dan verlaten we deze poging en beproeven $[d_2/x]\varphi$. Zo gaan we door tot het reeds aanwezige domein is uitgeput. Alleen als al deze pogingen gefaald hebben, springen we terug naar de oude regel, en voeren een nieuw object in D in.
Voor de existentiële kwantor kunnen we evenzo te werk gaan.
De nieuwe 'probeerregels' noemen we $\exists_L{}^*$ en $\forall_R{}^*$.

Voorbeeld 9.7

We maken nu nog eens opnieuw een semantisch tableau voor de sequent uit voorbeeld 9.6, maar nu met gebruikmaking van $\exists_L{}^*$ en $\forall_R{}^*$:

	$\forall y\, \exists x\, Rxy \circ \exists x\, \forall y\, Rxy$	$D = \{d_1\}$
$\forall_L, \exists_R$	$\vert$	
	$\exists x\, Rxd_1 \circ \forall y\, Rd_1y$	$D = \{d_1\}$
$\exists_L{}^*$	$\vert$	
	$Rd_1d_1 \circ$	$D = \{d_1\}$
$\forall_R{}^*$	$\vert$	
	$\circ\, Rd_1d_1$	$D = \{d_1\}$
	=	

Deze laatste sequent sluit en heeft dus geen tegenvoorbeeld. Maar hier hebben we nog slechts geprobeerd of d_1 al een tegenvoorbeeld levert. Dit lukt niet, dus *herzien* we de laatste stap ($\forall_R{}^*$) en voeren daar alsnog een nieuw object d_2 in. Dit geeft:

	$\vert$	
$\forall_R$	$\circ\, Rd_1d_2$	$D = \{d_1, d_2\}$
	$\vert$	
$\forall_L, \exists_R$ (!)	$\exists x\, Rxd_2 \circ \forall y\, Rd_2y$	$D = \{d_1, d_2\}$
	$\vert$	
$\exists_L{}^*$ (!)	$Rd_1d_2 \circ$	$D = \{d_1, d_2\}$
	=	

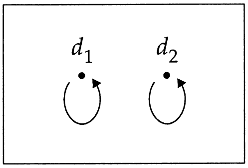
Dit levert evenmin een tegenvoorbeeld, Rd_1d_2 komt immers aan weerskanten van o voor. We proberen dan in het laatste geval d_2:

$\exists_L{}^*$ | Rd_2d_2 o | $D = \{d_1, d_2\}$

Reduceren we nu $\forall y\ Rd_2y$ rechts door voor y d_1 te kiezen, dan krijgen we:

$\forall_R{}^*$ | o Rd_2d_1 | $D = \{d_1, d_2\}$

Nu is geen regel meer toepasbaar en er is geen sluiting opgetreden. Aan de gevonden eindige open tak kunnen we een tegenvoorbeeld voor de topsequent aflezen. Dit is het *eindige* model met domein $D = \{d_1, d_2\}$ en relatie $I(R) = \{\langle d_1, d_1\rangle, \langle d_2, d_2\rangle\}$. In het plaatje van dit model zien we nog eens dat $\forall y\ \exists x\ Rxy$ waar is en $\exists x\ \forall y\ Rxy$ niet:



De probeermethode is nuttig indien er eindige tegenvoorbeelden bestaan. Bestaan die echter niet, dan zal de hele probeerfase verspilde extra moeite blijken, want uit het sluiten van een tableau waarin sterregels zijn toegepast, kan men geen algemene conclusies trekken over de geldigheid van de corresponderende gevolgtrekking. In de praktijk is het dus niet zonder meer aan te raden met de stervarianten van de tableauregels te werken.

Belangrijk is bovendien het volgende. Hoewel in het voorgaande geval inderdaad een oneindig tegenvoorbeeld vervangen kon worden door een eindig, zal dit *in het algemeen* zeker niet mogelijk zijn. Zo zagen we in hoofdstuk 7 een voorbeeld van een formule φ die wel modellen had, maar alleen oneindige. Willen we met een tableau testen of zekere conclusie ψ al dan niet uit φ volgt, dan zal deze formule aan de linkerkant geplaatst, onherroepelijk tot een oneindige tak leiden en dan nog wel één die niet door 'proberen' eindig is te omzeilen.

9.4 SAMENVATTING EN OPMERKINGEN

In een predikaatlogisch tableau kunnen zich twee situaties voordoen:
1 Het tableau sluit.
Het tableau is dan eindig en de gevolgtrekking die de sequent boven in het tableau uitdrukt is *geldig*.
2 Er is een niet–sluitende tak. Deze kan:
2.1 eindig afbreken, maar ook
2.2 oneindig doorlopen.
In beide gevallen beschrijft zo'n tak een tegenvoorbeeld en de gevolgtrekking die de sequent boven in het tableau uitdrukt, is *niet geldig*.

Onbeslisbaarheid van de predikaatlogica

Hier weerspiegelt zich een essentieel verschil tussen de predikaatlogica en de eerdere propositielogica. De propositielogica is *beslisbaar*, de predikaatlogica niet. Dit negatieve resultaat staat bekend als de 'Stelling van Church' (1936). Het bewijs van deze stelling is gecompliceerd en valt buiten het bestek van dit boek. Bij semantische tableaus uit zich deze onbeslisbaarheid onder meer in het feit dat niet steeds effectief valt te beslissen of we op een oneindige tak in wording zitten.
Voor sommige *fragmenten* van de predikaatlogica bestaan overigens wel beslissingsmethoden voor geldigheid. Voorbeelden zijn de monadische predikaatlogica en de deeltaal der universele formules zonder functiesymbolen (zie hoofdstuk 8).

Wat de tableaumethode wel gemeen heeft met haar propositielogische verwant, is *adequaatheid*. Mits we nu werken met zogenaamde 'fair scheduling', om ervoor te zorgen dat altijd iedere op een sequent toepasbare reductieregel ooit aan de beurt komt (in elke nog niet gesloten tak), hebben we:

Adequaatheid

Een sequent heeft een gesloten tableau dan en slechts dan als de corresponderende gevolgtrekking geldig is.

Dit resultaat zal in hoofdstuk 11 bewezen worden. Anders gezegd, de onbeslisbaarheid die we hier zijn tegengekomen, is niet een toevallig effect van de tableaumethode, maar weerspiegelt een essentiële eigenaardigheid van geldigheid in de predikaatlogica.

9.5 OPGAVEN

9.1 Bewijs met semantische tableaus:

i $\neg\exists x\,(Ax \wedge Bx), \exists x\,(Bx \wedge Cx) \models \neg\forall x\,(Cx \rightarrow Ax)$

ii $\forall x\,(Ax \rightarrow Bx) \vee \forall y\,(By \rightarrow Ay) \models \forall x\,\forall y\,((Ax \wedge By) \rightarrow (Bx \vee Ay))$

iii $\forall x\,\forall y\,((Ax \wedge By) \rightarrow (Bx \vee Ay)) \models \forall x\,(Ax \rightarrow Bx) \vee \forall y\,(By \rightarrow Ay)$

iv $\models \neg\exists x\,\forall y\,(Rxy \leftrightarrow \neg Ryy)$

v $\models \neg\exists x\,\forall y\,(Rxy \leftrightarrow \neg\exists z\,(Ryz \wedge Rzy))$

* vi $\forall x\,\forall y\,(Rxy \rightarrow \neg Ryx), \forall x\,\forall y\,(Rxy \rightarrow \forall z\,(Rxz \vee Rzy)) \models \forall x\,\forall y\,(Rxy \rightarrow \forall z\,(Ryz \rightarrow Rxz))$

9.2 Geef met behulp van een semantisch tableau een tegenvoorbeeld voor de volgende gevolgtrekking:

$\forall x\,\forall y\,((Ax \wedge By) \rightarrow Rxy), \forall x\,\neg Rxx, \forall x\,(Ax \vee Bx) \;/\; \forall x\,\forall y\,(Rxy \rightarrow Ax)$

9.3 Test de volgende syllogismen op geldigheid en geef in geval van niet–geldigheid een tegenvoorbeeld:

i $\forall x\,(Ax \rightarrow Bx), \exists x\,(Ax \wedge Cx) \;/\; \exists x\,(Cx \wedge Bx)$

ii $\forall x\,(Ax \rightarrow Bx), \exists x\,(Ax \wedge \neg Cx) \;/\; \exists x\,(Cx \wedge \neg Bx)$

iii $\neg\exists x\,(Ax \wedge Bx), \forall x\,(Bx \rightarrow Cx) \;/\; \neg\exists x\,(Cx \wedge Ax)$

* 9.4 Hoe moet de tableaumethode worden aangepast om formules met vrije variabelen toe te staan in de topsequent?

* 9.5 Vind met een semantisch tableau een kleinste tegenvoorbeeld voor:
$\exists!x\,\exists!y\,Rxy \;/\; \exists!y\,\exists!x\,Rxy$

($\exists!$ betekent 'precies één'. Zie ook opgave 6.8.)
Welke reductieregels gebruikt u voor de benodigde identiteit ' = '?

Predikaatlogica: afleidingen

Hoofdstuk 10

Predikaatlogica: afleidingen

Nadat in het vorige hoofdstuk een uitgebreide versie van semantische tableaus is geïntroduceerd om *geldigheid* van een gevolgtrekking in de predikaatlogica te testen, zal in dit hoofdstuk het systeem van natuurlijke deductie uit de propositielogica uitgebreid worden om *afleidingen* te maken in de predikaatlogica.

10.1 AFLEIDBAARHEID IN NATUURLIJKE DEDUCTIE

De afleidingsregels die we al kennen uit de propositielogica, kunnen weer gebruikt worden. Verder zijn afleidingsregels nodig voor de kwantoren $\forall$ en $\exists$. Voor elk van deze kwantoren zullen we een introductie– en eliminatieregel geven. Alle begrippen die in hoofdstuk 4 zijn ingevoerd met betrekking tot het afleidingssysteem voor de propositielogica, vinden we woordelijk terug in de predikaatlogica, zoals 'afleidbaarheid', 'stelling', 'consistentie' en de afleidbaarheidsnotatie '$\vdash$'.

Afleidingsregel $\forall$E

De eliminatieregel voor de universele kwantor weerspiegelt zijn typische gebruik: het nemen van een of ander bijzonder geval ('instantie') van de algemene uitspraak. De $\forall$E–regel wordt daarom ook wel de regel van *instantiatie* genoemd.

$$\dfrac{\begin{array}{c}\Sigma \\ \vdots \\ \forall x\ \varphi\end{array}}{[t/x]\varphi}\ \forall\text{E} \qquad \text{mits } t \text{ vrij is voor } x \text{ in } \varphi$$

Voorbeeld 10.1

Een correcte instantiatie waarin z voor x is gesubstitueerd:

$$\dfrac{\forall x\ \exists y\ Rxy}{\exists y\ Rzy}\ \forall\text{E}$$

Voorbeeld 10.2

De volgende afleiding is niet juist, omdat y niet vrij is voor x in $\exists y\, Rxy$:

$$\dfrac{\forall x\, \exists y\, Rxy}{\exists y\, Ryy}$$

Afleidingsregel $\forall$I

De introductieregel voor de universele kwantor $\forall x$ weerspiegelt de kenmerkende manier om zo'n bewering aan te tonen: bewijs φ voor een 'willekeurige' y, dat wil zeggen, voor een object waarover niets bijzonders is aangenomen. De $\forall$I–regel heet ook wel de regel van *generalisatie*:

$$\dfrac{\begin{array}{c}\Sigma \\ \vdots \\ \varphi\end{array}}{\forall x\, [x/y]\varphi}\ \forall\text{I}$$

mits y niet vrij in Σ,
x vrij voor y in φ,
en x niet vrij in $\forall y\ \varphi$

De restrictie is redelijk. Stel namelijk dat er een aanname in Σ zit die y als vrije variabele bevat. De afleiding van φ uit Σ geldt dan alleen voor die objecten waarvoor die aanname geldt. De conclusie $\forall x\, [x/y]\varphi$ is dan niet juist. In het meest flagrante geval zouden we zo immers $\forall x\, Px$ kunnen afleiden uit Py, hetgeen semantisch geheel ongewenst is. (In de meeste toepassingen van $\forall$I volstaat het om te denken: 'als φ afleidbaar is uit Σ en y komt niet voor in Σ, dan is $\forall y\ \varphi$ afleidbaar uit Σ'. De substitutie en de ingewikkelde restrictie zijn nodig om in deze stap ook nog van gebonden variabele te veranderen. Iets dergelijks geldt voor de regel $\exists$E, die verderop wordt geïntroduceerd.)

We illusteren het functioneren der $\forall$–regels in twee langere afleidingen.

Voorbeeld 10.3

$\forall x\, (Ax \to Bx) \vdash \forall x\, Ax \to \forall x\, Bx$:

$$\dfrac{\dfrac{\dfrac{\dfrac{1}{\forall x\, Ax}\ \forall\text{E} \qquad \dfrac{\forall x\, (Ax \to Bx)}{Ay \to By}\ \forall\text{E}}{Ay}}{By}}{\dfrac{\forall x\, Bx}{\forall x\, Ax \to \forall x\, Bx}\ [-1]}\ \forall\text{I}$$

Voorbeeld 10.4

$\forall x\, Ax \vee \forall x\, Bx \vdash \forall x\, (Ax \vee Bx)$:

$$\dfrac{\forall x\, Ax \vee \forall x\, Bx \qquad \dfrac{\dfrac{\dfrac{\dfrac{1}{\forall x\, Ax}\,\forall\mathrm{E}}{Ay}}{Ay \vee By}\,\forall\mathrm{I}}{\forall x\, (Ax \vee Bx)} \qquad \dfrac{\dfrac{\dfrac{\dfrac{2}{\forall x\, Bx}\,\forall\mathrm{E}}{By}}{Ay \vee By}\,\forall\mathrm{I}}{\forall x\, (Ax \vee Bx)}}{\forall x\, (Ax \vee Bx)}\,[-1, -2]$$

Bewijsstrategie

Een vaak gevolgde strategie bij dit soort bewijzen is om te beginnen met een geschikte instantiatie en te eindigen met een geschikte generalisatie.

Afleidingsregel $\exists I$

Vervolgens bespreken we de regels voor de existentiële kwantor. Een existentiële bewering kan typisch worden ingevoerd in een bewijs zodra we een speciaal geval bewezen hebben:

$$\dfrac{\begin{array}{c}\Sigma \\ \vdots \\ [t/x]\,\varphi\end{array}}{\exists x\, \varphi}\,\exists\mathrm{I} \qquad \text{mits } t \text{ vrij is voor } x \text{ in } \varphi$$

De voorwaarde zorgt ervoor dat bijvoorbeeld de volgende ongewenste afleiding inderdaad niet is toegestaan:

$$\dfrac{\forall y\, Ryy}{\exists x\, \forall y\, Ryx}$$

Hier is y namelijk niet vrij voor x in $\forall y\, Ryx$, omdat y daarin gebonden is. De volgende afleidingsstap is wel toegestaan, want y is vrij voor x in Ryx aangezien y nu ongebonden voorkomt:

$$\dfrac{Ryy}{\exists x\, Ryx}\,\exists\mathrm{I}$$

Afleidingsregel ∃E

De eliminatieregel voor ∃ is gecompliceerder. Dit valt te begrijpen naar analogie met de ∨E–regel. Net zoals een disjunctie geen van beide disjuncten impliceert, lijkt een existentiële bewering weinig informatie te bevatten, omdat deze geen enkel speciaal voorbeeld impliceert. Een conclusie uit een existentiële bewering moet dus losstaan van een specifiek voorbeeld van die bewering. Deze restrictie vinden we inderdaad terug in de nu volgende formulering van de ∃E–regel:

$$\dfrac{\exists x\,\varphi \qquad \begin{matrix}\Phi & \Sigma,\,[y/x]\varphi \\ \vdots & \vdots \\ \psi & \psi\end{matrix}}{\psi}\ \exists\text{E},\,[-[y/x]\varphi]$$

mits y niet vrij in Σ of ψ, y vrij voor x in φ, en y niet vrij in $\exists x\,\varphi$

Strategie

Wanneer we uit een existentiële bewering iets willen afleiden, is de toepassing van de ∃E–regel meestal een van de *laatste* bewijsstappen.

Voorbeeld 10.5

$\exists x\,(Ax \wedge Bx) \vdash \exists x\,Ax \wedge \exists x\,Bx$:

$$\dfrac{\exists x\,(Ax \wedge Bx) \qquad \dfrac{\dfrac{\dfrac{1}{Ay \wedge By}}{Ay}}{\exists x\,Ax} \qquad \dfrac{\dfrac{\dfrac{1}{Ay \wedge By}}{By}}{\exists x\,Bx}}{\exists x\,Ax \wedge \exists x\,Bx}\ \exists\text{E},\,[-1]$$

Voorbeeld 10.6

In de volgende afleiding wordt de ∃E–regel verkeerd gebruikt, omdat y als vrije variabele voorkomt in $\exists z\,Ryz$:

$$\dfrac{\dfrac{\exists x\,Rxx \qquad \dfrac{\dfrac{1}{Ryy}}{\exists z\,Ryz}}{\exists z\,Ryz}\ [-1]}{\forall x\,\exists z\,Rxz}\ \forall\text{I}$$

Voorbeeld 10.7

Het syllogisme 'sommige A zijn B, geen B is C, dus niet alle A zijn C' kan in termen van afleidbaarheid als volgt worden weergegeven:
$\exists x\,(Ax \wedge Bx), \neg\exists x\,(Bx \wedge Cx) \vdash \neg\forall x\,(Ax \to Cx)$.
Een afleiding hiervoor ziet er als volgt uit:

$$\dfrac{\dfrac{\exists x\,(Ax \wedge Bx) \qquad \dfrac{\dfrac{\dfrac{\dfrac{1}{Ay \wedge By}}{By} \qquad \dfrac{\dfrac{1}{Ay \wedge By}}{Ay} \quad \dfrac{2}{\forall x\,(Ax \to Cx)}}{Ay \to Cy}}{Cy}}{By \wedge Cy}}{\exists x\,(Bx \wedge Cx)} \qquad \dfrac{}{\neg\exists x\,(Bx \wedge Cx)}\,[-2]}{\neg\forall x\,(Ax \to Cx)}\,\exists E, [-1]$$

Voorbeeld 10.8

Kwantorwisseling zijn we eerder tegengekomen bij de behandeling van semantische tableaus. Hier volgt een afleiding van de semantisch geldige gevolgtrekking $\exists x\,\forall y\,Rxy \;/\; \forall y\,\exists x\,Rxy$:

$$\dfrac{\exists x\,\forall y\,Rxy \qquad \dfrac{\dfrac{\dfrac{1}{\forall y\,Ruy}}{Ruv}}{\exists x\,Rxv}}{\forall y\,\exists x\,Rxy}\,\exists E, [-1]$$

10.2 AXIOMATISCHE THEORIEËN

In de propositielogica was er een alternatief bewijssysteem voor natuurlijke deductie, namelijk een aantal *axioma's* plus de regel Modus Ponens. Ook voor de predikaatlogica is een axiomatisch bewijssysteem een alternatief. We formuleren weer voor een taal met alleen $\neg$ en $\to$, en verder slechts de kwantor $\forall$, gebruikmakend van de equivalentie $\exists x\,\varphi \leftrightarrow \neg\forall x\,\neg\varphi$. De eerdere axiomatiek moet uitgebreid worden met de volgende axiomaschema's voor kwantoren:

1	$(\forall x\,(\varphi \to \psi)) \to (\forall x\,\varphi \to \forall x\,\psi)$	
2	$\varphi \to \forall x\,\varphi$	mits x niet vrij voorkomt in φ
3	$\forall x\,\varphi \to [t/x]\varphi$	mits de term t vrij is voor x in φ

Generalisatie

Er komt echter nog een verruiming bij. Onder een *generalisatie* van een formule φ verstaan we elke formule van de vorm $\forall x_1 \ldots \forall x_n\, \varphi$, voor willekeurige variabelen $x_1, \ldots, x_n$. Generalisaties van de formule $Rxy \wedge \exists z\, Sz$ zijn bijvoorbeeld $\forall x\, (Rxy \wedge \exists z\, Sz)$, $\forall x\, \forall y\, \forall u\, (Rxy \wedge \exists z\, Sz)$, enzovoort. We voegen nu nog een extra afleidingsregel toe, die zegt dat elke generalisatie van een axiomatisch afleidbare formule zelf ook weer afleidbaar is:

4 Uit φ volgt $\forall x\, \varphi$.

Weer kan bewezen worden dat dit systeem equivalent is in bewijskracht met natuurlijke deductie.

Predikaatlogische theorieën

Wat zijn nu in feite belangrijke aannames Σ waarmee we werken in de wiskunde en informatica om stellingen uit af te leiden? Er zijn veel interessante *voorbeelden* van theorieën die op het bewijssysteem voor de predikaatlogica zijn gebaseerd. We geven een aantal voorbeelden, overigens meer voor de algemene ontwikkeling van de lezer dan om er hier in dit boek mee te werken. We willen slechts aantonen dat de predikaatlogica buitengewoon veelzijdig en rijk aan uitdrukkingskracht is.

Peano–rekenkunde

Een eerste grote groep zijn axiomatieken voor belangrijke wiskundige deelgebieden, zoals rekenkunde, meetkunde of verzamelingenleer. Een bekend voorbeeld is de zogenaamde *Peano–rekenkunde* PA. Dit is een theorie om het optellen en vermenigvuldigen van natuurlijke getallen formeel te beschrijven. De taal van PA heeft één individuele constante 0 en drie functieletters: +, · (beide tweeplaatsig) en de opvolgerfunctie S (eenplaatsig). Termen van de taal zijn dan bijvoorbeeld 0, $S0$, $SS0$,..., $0 + x$, $x \cdot y$, $S(x + Sy)$, enzovoort. Het (oneindige) rijtje termen 0, $S0$, $SS0$, ... kunnen we daarbij interpreteren als namen voor de natuurlijke getallen 0, 1, 2, ...
De axioma's voor PA zijn:

PA1 $\forall x \neg\, 0 = Sx$
PA2 $\forall x\, \forall y\, (Sx = Sy \rightarrow x = y)$
PA3 $\forall x\, x + 0 = x$
$\forall x\, \forall y\, x + Sy = S(x + y)$
PA4 $\forall x\, x \cdot 0 = 0$
$\forall x\, \forall y\, x \cdot Sy = x \cdot y + x$
PA5 $([0/x]\varphi \wedge \forall x(\varphi \rightarrow [Sx/x]\varphi)) \rightarrow \forall x\, \varphi$ (voor elke formule φ)

PA1 zegt dat 0 van geen getal de opvolger is.
PA2 zegt dat de opvolgerfunctie injectief is.
PA3 en PA4 geven de recursievergelijkingen voor optelling respectievelijk vermenigvuldiging.
PA5 ten slotte geeft alle instanties van het principe van natuurlijke inductie, voorzover uit te drukken in de taal van PA. In feite is natuurlijke inductie dus inductie op de opvolgerfunctie S uit deze taal van de rekenkunde, en daarmee een speciaal geval van de algemene inductie naar de opbouw van formules en termen.
Verder formaliseren van de rekenkunde binnen deze theorie komt nu neer op geschikt afleiden van stellingen uit deze axioma's met natuurlijke deductie of met de axiomatische bewijscalculus. Hiervoor zou strikt genomen een versie van natuurlijke deductie vereist zijn, waaraan we ook speciale 'identiteitsregels' hebben toegevoegd voor het opereren met gelijkheid. Hierover meer in hoofdstuk 12! In hoofdstuk 14 over berekenbaarheid gaan we overigens verder in op de rekenkunde.

Verzamelingenleer

Een belangrijke axiomatiek vormen de zogenaamde 'Zermelo–Fraenkel-axioma's' voor de verzamelingenleer. Een van die axioma's zegt bijvoorbeeld dat twee verzamelingen gelijk zijn als ze dezelfde elementen bevatten:

ZF1 $\quad \forall a\, \forall b\, (\forall x\, (x \in a \leftrightarrow x \in b) \rightarrow a = b)$

De machtsverzameling van een verzameling a is de verzameling van alle deelverzamelingen van a $\{x \mid x \subseteq a\}$. Een ander axioma zegt dat dit inderdaad een verzameling is:

ZF5 $\quad \forall a\, \exists b\, \forall x\, (x \in b \leftrightarrow \forall y (y \in x \rightarrow y \in a))$

De overige ZF–axioma's laten we onvermeld.

Partiële ordeningen

Een andere grote groep van wiskundige theorieën is op te vatten, niet als de beschrijving van een structuur (zoals in het geval van PA de structuur der natuurlijke getallen $\mathbb{N}$), maar eerder als een stel kenmerken van een grote klasse van modellen. Een voorbeeld hiervan is de theorie van geordende verzamelingen, die op veel plaatsen in de wiskunde een rol speelt. De ordeningsrelatie wordt meestal met een apart symbool aangegeven, vaak $\preccurlyeq$. De theorie van de *partiële* ordeningen wordt geaxiomatiseerd door de volgende axioma's:

O1 $\forall x\, x \preccurlyeq x$
O2 $\forall x\, \forall y\, ((x \preccurlyeq y \wedge y \preccurlyeq x) \rightarrow y = x)$
O3 $\forall x\, \forall y\, \forall z\, ((x \preccurlyeq y \wedge y \preccurlyeq z) \rightarrow x \preccurlyeq z)$

Deze axioma's drukken respectievelijk reflexiviteit, antisymmetrie en transitiviteit uit. De theorie der *lineaire* ordeningen ontstaat door het volgende axioma nog toe te voegen:

O4 $\forall x\, \forall y\, (x \preccurlyeq y \vee y \preccurlyeq x)$

De ordening is nu 'totaal': elk tweetal objecten is ten opzichte van elkaar geordend.

Axiomatieken in de informatica

Binnen de informatica komen we ook heel vaak verzamelingen formules Σ tegen waarmee wordt omgegaan als axiomastelsels. Voorbeelden zijn deductieve gegevensbestanden, abstracte datatypen en logische programma's. In volgende blokken van het boek komen we uitvoerig terug op axiomatieken in de informatica.

10.3 OPGAVEN

10.1 Bewijs met behulp van natuurlijke deductie:
i $\forall x\, (\varphi \wedge \psi) \vdash \forall x\, \varphi \wedge \forall x\, \psi$, en omgekeerd
ii $\exists x\, (\varphi \vee \psi) \vdash \exists x\, \varphi \vee \exists x\, \psi$, en omgekeerd

10.2 Bewijs de volgende, bij prenexvormconstructies gebruikte, gevolgtrekkingen (x niet vrij in ψ):
i $\exists x\, \varphi \rightarrow \psi \vdash \forall x\, (\varphi \rightarrow \psi)$, en omgekeerd
ii $\forall x\, \varphi \rightarrow \psi \vdash \exists x\, (\varphi \rightarrow \psi)$, en omgekeerd
Waar is $\neg$E* nodig?

10.3 Bewijs het volgende syllogisme: $\exists x\, (Ax \wedge Bx), \neg\exists x\, (Bx \wedge Cx) \vdash \neg\forall x\, (Ax \rightarrow Cx)$.

10.4 Bewijs dat de Russell–formule een stelling van het systeem van natuurlijke deductie is: $\neg\exists x\, \forall y\, ((Rxy \rightarrow \neg Ryy) \wedge (\neg Ryy \rightarrow Rxy))$.

10.5 We spreken van afleidingen zonder *annotatie* als niet wordt aangegeven welke afleidingsregels gebruikt worden. Is uit een afleiding zonder annotatie uniek te achterhalen welke stap in het systeem werd gemaakt?

10.6 Bewijs axiomatisch $\forall x\, (\neg(Ax \rightarrow Bx) \rightarrow (\neg Ax \rightarrow \neg Bx))$.

10.7 We voegen aan de axioma's van de Peano–rekenkunde de volgende axioma's toe met betrekking tot identiteit:
$\forall x\, x = x$
$\forall x\, \forall y\, (x = y \rightarrow [x/z]t = [y/z]t)$, voor elke term t
$\forall x\, \forall y\, (x = y \rightarrow ([x/z]\varphi \rightarrow [y/z]\varphi)$, voor elke formule φ

Bewijs nu in PA:
i $S0 + S0 = SS0$
ii $\forall x\, \forall y\, \forall z\, ((x + y) + z = x + (y + z))$

Predikaatlogica: metatheorie

Hoofdstuk 11

Predikaatlogica: metatheorie

In dit laatste hoofdstuk over de predikaatlogica beschouwen we weer enkele metaeigenschappen van dit systeem. De meest voor de hand liggende theoretische vragen zijn weer adequaatheid van tableaus en correctheid en volledigheid van de predikaatlogica. Deze eigenschappen zijn reeds voor de propositielogica bewezen en in principe gaan we nu langs dezelfde lijnen te werk.

11.1 ADEQUAATHEID VAN TABLEAUS

De semantische tableaus uit hoofdstuk 9 vormen een juiste methode om de geldigheid van een gevolgtrekking in de predikaatlogica te bewijzen of te weerleggen. Ter herinnering: we hebben ons beperkt tot een taal zonder functiesymbolen, individuele constanten en identiteit om niet–essentiële maar lastige complicaties te vermijden. We herinneren bovendien aan enkele andere afspraken. Een tableau heet gesloten als op elke tak, gegeven 'fair scheduling', sluiting optreedt. Een open tableau is een tableau met minstens één niet–sluitende tak.

STELLING 11.1

Adequaatheid
Als Σ een formuleverzameling is en φ een formule, dan geldt:

$\Sigma \models \varphi \Leftrightarrow$ er bestaat een gesloten tableau voor $\Sigma \circ \varphi$

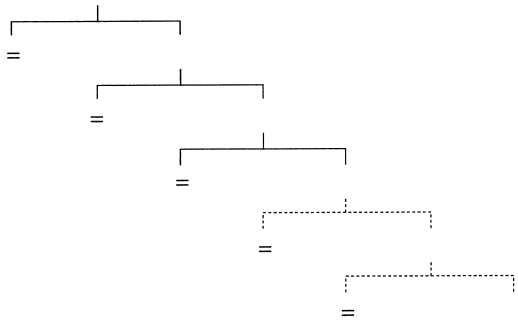
Bewijs

Net als in hoofdstuk 5 gaan we weer het volgende bewijzen: als een tableau gesloten is, dan is de corresponderende gevolgtrekking geldig ($\Leftarrow$), en als een tableau open is, dan is de corresponderende gevolgtrekking ongeldig ($\Rightarrow$). Als gevolg hiervan kan, ondanks de vrijheid in volgorde van regeltoepassing, een sequent nooit zowel open als gesloten tableaus hebben.

Oneindige takken

Alvorens tot het bewijs over te gaan eerst nog het volgende. Tot nu toe was niet duidelijk of een tableau met oneindig veel knopen open of gesloten is. Zo'n oneindig tableau blijkt altijd open te zijn. Met andere

woorden, oneindige vormen als de volgende, met steeds langere paden die alle tot sluiting leiden, kunnen zich niet voordoen voor een gesloten tableau:



Het onderliggend principe staat als volgt bekend (zonder bewijs):

LEMMA 11.1

'Königs Lemma'
Elke eindig vertakkende boom met oneindig veel knopen bevat een oneindige tak.

Dit principe kunnen we op tableaus toepassen. Een tableauboom is eindig vertakkend, aangezien bij elke sequent hoogstens een splitsing in tweeën optreedt. Dus bevat een oneindig tableau een oneindig voortlopende tak die dus open is, dus het tableau is open.

Na deze opmerking beginnen we nu aan het bewijs van de adequaatheidsstelling. Het adequaatheidsbewijs van hoofdstuk 5 valt in hoofdlijnen over te nemen, met geschikte aanpassingen aan de predikaatlogische situatie.

$\Leftarrow$

We willen bewijzen dat een gevolgtrekking geldig is, als een tableau ervoor gesloten is. Het is voldoende om het volgende te bewijzen:

BEWERING 11.1

Stel $\Phi \circ \Psi$ is de topsequent van een gesloten tableau. Dan geldt $\Phi \vDash \Psi$.

Bewijs

Dankzij de hiervoor bewezen eindigheid is een bewijs met inductie naar het aantal knopen in een gesloten tableau zinvol.

Basisstap

Voor eindsequenten $\Sigma \circ \Delta$ met $\Sigma \cap \Delta \neq \varnothing$ geldt kennelijk: als $M, b \vDash \varphi$ voor elke formule $\varphi \in \Sigma$, dan $M, b \vDash \varphi$ voor minstens een $\varphi \in \Delta$. Dus eindsequenten $\Sigma \circ \Delta$ corresponderen met een geldige gevolgtrekking $\Sigma \vDash \Delta$.

Inductiestap

Vervolgens moeten we alle reductieregels langslopen. Voor de propositionele gevallen is de verificatie identiek aan de eerdere. Voor de kwantorregels bekijken we alleen het geval van $\forall$:

Geval $\forall_L$

Gegeven een knoop $\Sigma, \forall x\, \varphi \circ \Delta$. Stel dat in de volgende stap een instantie van $\forall x\, \varphi$ links wordt toegevoegd. Er is dan een gesloten tableau met topknoop $\Sigma, [d/x]\varphi \circ \Delta$, dat een knoop minder bevat. Op grond van inductie geldt dus: $\Sigma, [d/x]\varphi \models \Delta$. Vanwege de geldige implicatie $\forall x\, \varphi \models [d/x]\varphi$ geldt dan ook $\Sigma, \forall x\, \varphi \models \Delta$.

Geval $\forall_R$

Gegeven een knoop $\Sigma \circ \Delta, \forall x\, \varphi$. Stel dat in de volgende stap $\forall x\, \varphi$ rechts is weerlegd door een nieuwe instantie $[d/x]\varphi$ rechts toe te voegen. Er is dan een gesloten tableau met topknoop $\Sigma \circ \Delta, [d/x]\varphi$ waar d niet voorkomt in Σ of Δ, dat één knoop minder bevat. Met inductie geldt dus: $\Sigma \models \Delta, [d/x]\varphi$. We willen aantonen dat ook $\Sigma \models \Delta, \forall x\, \varphi$. Dit gaat als volgt:
Zij M, b een willekeurig model met een bedeling die Σ waar maken. Er geldt $M, b \models \forall x\, \varphi$ of $M, b \not\models \forall x\, \varphi$.
Als geldt $M, b \models \forall x\, \varphi$, zijn we klaar. Minstens één formule in Δ is waar, dus $\Sigma \models \Delta, \forall x\, \varphi$.
Als $M, b \not\models \forall x\, \varphi$, dan is er volgens de waarheidsdefinitie een object a in het domein zodat $M, b[x \mapsto a] \not\models \varphi$. Ken nu dit object a aan d toe. Het overgaan op bedeling $b[x \mapsto a]$ verandert niets aan de evaluatie van de formules in Σ en Δ, omdat d daar immers niet in voorkomt. Dus tevens $M, b[x \mapsto a] \models \Sigma$. Wegens de inductiehypothese $\Sigma \models \Delta, [d/x]\varphi$ moet dan minstens één formule uit $\Delta \cup \{[d/x]\varphi\}$ waar zijn. Omdat $[d/x]\varphi$ dat niet kan zijn, moet minstens één formule uit Δ waar zijn. Dus $\Sigma \models \Delta, \forall x\, \varphi$. □

$\Rightarrow$

We dienen nu nog het volgende te bewijzen: een gevolgtrekking is ongeldig, als geen enkel tableau ervoor gesloten is. Dit gaat als volgt: Stel dat er geen gesloten tableau bestaat voor de sequent $\Sigma \circ \varphi$. Omdat een tableau altijd gemaakt kan worden, is er dan een open tableau. Dan is er in het tableau een tak τ, eindig of oneindig, die open is. We zoeken nu een tegenvoorbeeld van de gevolgtrekking Σ / φ. Dit blijkt het volgende model M te zijn.

Domein

Voor het domein D kiezen we bij elke op de tak τ genoemde d_i een object, zeg voor de concreetheid het natuurlijke getal i zelf.
D kan dus zowel eindig als oneindig uitvallen.

Interpretatiefunctie

De interpretatiefunctie I wordt dan als volgt gedefinieerd:

- $I(d_i) = i$, als d_i op τ voorkomt
- $I(P)(i_1, ..., i_n) \Leftrightarrow P(d_{i_1}, ..., d_{i_n})$ staat *links* op de tak τ.

Tegenvoorbeeld

Dit model levert een tegenvoorbeeld op voor Σ / φ. Om dit in te zien moeten we eerst de volgende bewering bewijzen.

BEWERING 11.2

Voor model M, bedeling b en tak τ als hiervoor geldt voor elke formule ξ: als ξ links op τ voorkomt, dan $M, b \vDash \xi$; als ξ rechts op τ voorkomt, dan $M, b \nvDash \xi$.

Bewijs

Basisstap

Stel $\xi = P(d_{i_1}, ..., d_{i_n})$. Dan volgt het gevraagde uit de definitie van de interpretatiefunctie I.

Inductiestap

De gevallen voor de connectieven gaan als in hoofdstuk 5. Resteren de kwantoren.
Stel $\xi = \forall x\, \varphi$. Dan geldt:

$\forall x\, \varphi$ komt links op τ voor
$\Rightarrow$ voor alle $i \in D$: $[d_i/x]\varphi$ staat links op τ ($\forall_L$–regel)
$\Rightarrow$ voor alle $i \in D$: $M, b \vDash [d_i/x]\varphi$ (inductiehypothese)
$\Rightarrow$ voor alle $i \in D$: $M, b[x \mapsto i] \vDash \varphi$ (bewering 8.2)
$\Rightarrow M, b \vDash \forall x\, \varphi$ (waarheidsdefinitie)

$\forall x\, \varphi$ komt rechts op τ voor
$\Rightarrow$ er is een $i \in D$: $[d_i/x]\varphi$ staat rechts op τ ($\forall_R$–regel)
$\Rightarrow$ er is een $i \in D$: $M, b \nvDash [d_i/x]\varphi$ (inductiehypothese)
$\Rightarrow$ er is een $i \in D$: $M, b[x \mapsto i] \nvDash \varphi$ (bewering 8.2)
$\Rightarrow M, b \nvDash \forall x\, \varphi$ (waarheidsdefinitie)

Het geval $\exists x\, \varphi$ gaat analoog. □

In het bijzonder geldt deze bewering voor de topsequent $\Sigma \circ \varphi$, dat wil zeggen: $M, b \vDash \psi$, voor elke $\psi \in \Sigma$ en $M, b \nvDash \varphi$. Dus M, b is een tegenvoorbeeld van de gevolgtrekking $\Sigma \,/\, \varphi$.

Hiermee is het bewijs van de adequaatheid van semantische tableaus voor de predikaatlogica voltooid. □

11.2 DE VOLLEDIGHEIDSSTELLING VOOR DE PREDIKAATLOGICA

Correctheid

Het tweede onderwerp in dit hoofdstuk is de semantische correctheid en volledigheid van de natuurlijke deductie voor de predikaatlogica. We beginnen met correctheid.

STELLING 11.2

Correctheid
Voor elke formuleverzameling Σ en formule φ geldt: $\Sigma \vdash \varphi \Rightarrow \Sigma \vDash \varphi$.

Bewijs

Correctheid is te bewijzen met inductie naar het aantal knopen in een afleidingsboom. In wezen komt dit neer op een controle van de semantische correctheid van de diverse afleidingsregels, waarbij we ons natuurlijk kunnen beperken tot de kwantorregels. We bespreken die voor de existentiële kwantor.

Geval $\exists$I

Stel dat $\exists$I de laatst gebruikte regel is. Er is dus een boom met minder knopen, met aannames Σ en conclusie $[t/x]\varphi$, met t vrij voor x in φ. Laat nu M, b zo zijn dat $M, b \vDash \Sigma$. Volgens de inductiehypothese op de kleinere boom geldt dan: $M, b \vDash [t/x]\varphi$. Volgens bewering 8.2 geldt dan: $M, b[x \mapsto V(t)] \vDash \varphi$. Er is dus een $d \in D$, namelijk $V(t)$, waarvoor geldt $M, b[x \mapsto d] \vDash \varphi$, zodat volgens de waarheidsdefinitie geldt: $M, b \vDash \exists x\, \varphi$.

Geval $\exists$E

Stel dat $\exists$E het laatst gebruikt is. Deze regel werd dus toegepast op twee bomen met minder knopen. De ene boom bestaat uit een afleiding van $\exists x\, \varphi$ met aannames uit Σ, de andere boom is een afleiding van ψ uit $\Delta \cup \{[y/x]\varphi\}$, onder zekere condities voor de gebruikte variabelen (die we nog zullen gebruiken). De inductiehypothese levert dan

- $\Sigma \vDash \exists x\, \varphi$ en $\Delta \cup \{[y/x]\varphi\} \vDash \psi$.

Stel nu dat $M, b \vDash \Sigma \cup \Delta$. Dan geldt dus ook $M, b \vDash \Sigma$ en dus $M, b \vDash \exists x\, \varphi$. Volgens de waarheidsdefinitie is er dan een $d \in D$ zodat $M, b[x \mapsto d] \vDash \varphi$. Met bewering 8.2 volgt dan ook dat $M, b' \vDash [y/x]\varphi$ voor de bedeling b' die gelijk is aan b met eventueel het verschil dat $b'(y) = d$. Voorts geldt $M, b \vDash \Delta$, en omdat y niet vrij voorkomt in Δ, ook $M, b' \vDash \Delta$. Dus $M, b' \vDash \Delta \cup \{[y/x]\varphi\}$, zodat volgens de inductiehypothese ook $M, b' \vDash \psi$. Omdat y niet als vrije variabele in ψ voorkomt, geldt ook $M, b \vDash \psi$. Met andere woorden, zoals gewenst:

- $\Sigma \cup \Delta \vDash \psi$. □

Volledigheid

Resteert de omgekeerde richting van de volledigheidsstelling. In tegenstelling tot het voorgaande bewijs van correctheid en het bewijs van de volledigheid van de propositielogica geven we slechts een ruwe schets van het bewijs van de volledigheid van de predikaatlogica.

STELLING 11.3

Volledigheid
Voor elke formuleverzameling Σ en formule φ geldt: $\Sigma \vDash \varphi \Rightarrow \Sigma \vdash \varphi$.

Bewijs

We bewijzen de volledigheid net als in hoofdstuk 5 met contrapositie. Stel $\Sigma \nvdash \varphi$. Als φ niet afleidbaar is uit Σ, dan is $\Sigma \cup \{\neg\varphi\}$ consistent. Met behulp van een Lindenbaum–constructie die lijkt op die in hoofdstuk 5, breiden we $\Sigma \cup \{\neg\varphi\}$ uit tot een maximaal consistente verzameling Σ^*. De constructie is in zoverre anders dat we de taal blijken te moeten

uitbreiden met oneindig veel nieuwe constanten. Tijdens de Lindenbaum–constructie voegen we namelijk voor iedere existentiële formule $\exists x\ \varphi$ een formule $[c/x]\varphi$ toe voor een nieuwe constante c (zo'n constante wordt een *getuige* genoemd). Deze nieuwe constanten hebben we nodig om als volgt een model van $\Sigma \cup \{\neg\varphi\}$ te construeren.
Het model M voor Σ^* heeft als domein alle constanten die in de taal voorkomen, en heeft als benodigde interpretatiefunctie voor predikaatletters uit Σ^*: $I(P)(c_1, ..., c_n) \Leftrightarrow P(c_1, ..., c_n) \in \Sigma^*$.
Inductie naar het aantal logische operatoren in formules ξ toont dan aan dat niet alleen voor atomen maar ook voor formules in het algemeen geldt: $\xi \in \Sigma^* \Leftrightarrow M, b \models \xi$.
In het bijzonder geldt dus: $M, b \models \Sigma$ en $M, b \not\models \varphi$, ofwel: $\Sigma \not\models \varphi$.
Hiermee is de volledigheid van de predikaatlogica bewezen. □

11.3 MODELTHEORIE

Verdere theorievorming binnen de predikaatlogica gaat langs de beide 'fronten' van de volledigheidsstelling: de semantiek en de bewijsbaarheid. Deze gebieden hebben zelfs geleid tot logische subspecialisaties, te weten de zogenaamde *modeltheorie* respectievelijk *bewijstheorie*. Als een voorproefje van moderne logische theorievorming die we bijvoorbeeld ook in de informatica terugvinden, behandelen we nu achtereenvolgens een modeltheoretisch onderwerp (semantische monotonie), een bewijstheoretisch onderwerp (tableaubewijzen) en een onderwerp dat vanuit beide specialisaties benaderd kan worden (expliciet en impliciet definiëren).

Modeltheorie

Modeltheoretische vragen rijzen zodra we ons systematisch buigen over het verband tussen de syntactische vorm van beweringen en hun gewenst semantisch gedrag in structuren. Voorbeelden hiervan kwamen we reeds tegen in hoofdstuk 8, in de bespreking van fragmenten van de predikaatlogica.

Behoud van beweringen onder deelmodellen

Zo kunnen we universele zinnen in verband brengen met deelmodellen: overgaan van een model naar een deelmodel laat de waarheid van universele beweringen onaangetast (zie hoofdstuk 8 en opgave 8.5).
Dit is handig: zodra we bijvoorbeeld weten dat een ordening transitief is – $\forall x\ \forall y\ \forall z\ ((x < y \wedge y < z) \rightarrow x < z)$ – op de reële getallen, dan hoeven we deze eigenschap niet te gaan controleren indien we met deze ordening binnen een deelsituatie zoals de natuurlijke getallen gaan werken. Let wel, voor niet–universele eigenschappen bestaat deze garantie niet: dichtheid bijvoorbeeld – $\forall x\ \forall y\ (x < y \rightarrow \exists z\ (x < z \wedge z < y))$ – gaat juist verloren bij de overgang van reële naar natuurlijke getallen.

Een natuurlijke theoretische vraag is of we hiermee dit verschijnsel van 'behoud van beweringen onder submodellen' uitputtend hebben beschreven. Zijn er nog andere syntactische vormen van beweringen die zo'n overgang zouden overleven? Het volgende fundamentele modeltheoretische resultaat zegt dat die er niet zijn:

STELLING 11.4

'Preservatiestelling van Los'
Predikaatlogische zinnen blijven waar bij het overgaan naar deelmodellen dan en slechts dan als ze logisch equivalent zijn met een universele zin.

Querying en monotonie

Een andere modeltheoretische vraag komt voort uit de theorie van relationele gegevensbanken.

Querying

Het opvragen van predikaten (*querying*) in gegevensbanken verloopt vaak via complexe omschrijvingen. De primaire informatie ligt opgeslagen in predikaten. Die predikaten beschrijven een relationele structuur. We willen nu informatie over nieuwe complexe combinaties van predikaten.
Als we bijvoorbeeld een bestand hebben dat gegevens bevat over de familierelatie ouderschap, dan kunnen we de grootouderrelatie expliciet als volgt uitdrukken:

$$Grootouder\_van(x, y) \leftrightarrow \exists z\,(Ouder\_van(x, z) \wedge Ouder\_van(z, y))$$

Heel gebruikelijk is een situatie waarin we geen expliciete definitie voor een predikaat kunnen geven, maar wel een minder doorzichtige definitie in de vorm van een recursieve omschrijving.
Een standaardvoorbeeld van zo'n recursieve omschrijving is de voorouderrelatie. De voorouderrelatie kunnen we als volgt omschrijven:

$$Voorouder\_van(x, y)$$
$$\leftrightarrow (Ouder\_van(x, y) \vee \exists z\,(Ouder\_van(x, z) \wedge Voorouder\_van(z, y)))$$

Hoe kan het systeem deze omschrijving verwerken en de verzameling antwoordparen voor de voorouderrelatie berekenen? Een standaard procédé is om de interpretatie van de omschrijving stapsgewijs te *benaderen.* Voor de voorouderrelatie gaat dit als volgt.

Stapsgewijs interpreteren

Om te beginnen weten we niets, en stellen daarom in eerste benadering dat de interpretatie leeg is: $I_0(Voorouder\_van) = \varnothing$.
Met deze eerste benadering gaan we nu verder. We passen de gegeven omschrijving toe, met de reeds gevonden voorouderparen in het rechterlid ingevuld, om een nieuwe benadering $I_1(Voorouder\_van)$ te

krijgen. Een lege interpretatie correspondeert met nul voorouderparen rechts in de omschrijving, dus de tweede benadering wordt de ouderrelatie zelf. Merk verder op dat $I_0(Voorouder\_van) \subseteq I_1(Voorouder\_van)$. Iteratie van dit proces levert een serie benaderingen $I_0 \subseteq I_1 \subseteq I_2 \subseteq ...$ van predikaten op het domein van de gegevensbank. Zodra een zekere I_{k+1} gelijkblijft aan I_k stoppen we. Het proces is dan gestabiliseerd. We geven de laatst berekende interpretatie als antwoord. Merk op dat dit in eindige structuren betekent dat het benaderingsproces termineert: er zijn slechts eindig veel paren die kunnen worden toegevoegd.

Wat zijn de logische achtergronden van dit procédé? Een voor de hand liggende vraag is waarom deze iteratieprocedure termineert en met inclusie verloopt. Dat dit niet vanzelf spreekt, kunnen we constateren bij de gefingeerde familierelatie 'weifelouder':

$$Weifelouder\_van(x, y) \leftrightarrow (Ouder\_van(x, y) \vee \neg Weifelouder\_van(x, y))$$

We krijgen nu de volgende weinig informatieve reeks 'benaderingen': lege relatie, universele relatie, ouderrelatie, universele relatie, ouderrelatie en daarna herhaling van dit inmiddels zichtbare patroon en dus geen terminatie en geen benadering in eigenlijke zin. Wat gaat hier verkeerd en bij de vooruderrelatie juist goed?
Het antwoord ligt in de modeltheorie. Het blijkt van cruciaal belang dat elke volgende benadering tenminste een uitbreiding is: paren van individuen die reeds aan $I_k(P)$ voldeden, moeten nog steeds voldoen aan $I_{k+1}(P)$, ofwel $I_k(P) \subseteq I_{k+1}(P)$. Dit kunnen we garanderen als de definiërende omschrijving rechts van $\leftrightarrow$ *monotoon* is:

DEFINITIE 11.1

Monotoon
Laat M_1 een model zijn en P een predikaatletter. Een formule φ heet *semantisch monotoon* met betrekking tot P als geldt: indien M_1, $b \vDash \varphi$ en een model M_2 verschilt alleen van M_1 in zoverre dat $I_{M_1}(P) \subseteq I_{M_2}(P)$, dan geldt M_2, $b \vDash \varphi$.

Positief

Welke vormen van definitie garanderen monotonie? De logica levert het volgende antwoord (te bewijzen met formule–inductie):

STELLING 11.5

Een formule φ die een predikaat P omschrijft is semantisch monotoon, indien ieder voorkomen van de predikaatletter P in φ syntactisch *positief* is, dat wil zeggen: met een schrijfwijze voor φ in termen van alleen $\neg$, $\wedge$, $\vee$, $\forall$, $\exists$ ligt elk voorkomen van P binnen het bereik van een *even* aantal negatietekens.

Merk op dat inderdaad *Voorouder_van* positief voorkomt in zijn definiërende beschrijving, maar *Weifelouder_van* juist niet.
Wezenlijk complexer dan het volgen van monotonie uit positiviteit is de omkering ervan, wat bekendstaat als de 'Stelling van Lyndon'.

STELLING 11.6

'Stelling van Lyndon'
Indien een predikaatlogische formule φ semantisch monotoon is met betrekking tot P, dan is φ logisch equivalent met een formule waarin P alleen syntactisch positief voorkomt.

Alleen positieve queries garanderen dus stabilisatie.

Er valt overigens over de semantiek van dit voorbeeld met familierelaties nog veel meer te zeggen. Informatici hebben de laatste tijd belangrijke bijdragen geleverd aan de semantiek van eindige gegevensbestanden, de zogenaamde *eindige modeltheorie*. Anderzijds kunnen we bijvoorbeeld ook de stapsgewijze benadering uitvoeren op *oneindige* modellen. Terminatie is dan alleen gegarandeerd in een meer algemene zin, waarbij van oneindige approximatie sprake is.

11.4 SEQUENTENBEWIJZEN

We geven nu een voorbeeld van een thema uit de logische bewijstheorie, namelijk *tableau–* of *sequentenbewijzen*. We hebben inmiddels drie methoden om de geldigheid van gevolgtrekkingen na te gaan:
a gesloten semantische tableaus;
b natuurlijke deductiebomen;
c axiomatische bewijsreeksen.

De equivalentie tussen b en c is zuiver *combinatorisch* te bewijzen, dat wil zeggen: elke regel van het ene systeem kan uitgedrukt worden in regels van het andere en omgekeerd. De belangrijkste benodigde observatie voor zo'n combinatorisch bewijs is de deductiestelling uit hoofdstuk 4. Uit volledigheid en adequaatheid weten we dat ook a en b equivalente methoden zijn (en dus ook a en c). Ook hier kunnen we een puur combinatorische vergelijking maken, maar a en b zijn niet evident combinatorisch equivalent: de structuur van gesloten tableaus blijkt niet zo eenvoudig vergelijkbaar met de structuur van een afleiding. In het bijzonder komt de in axiomatisch bewijzen centrale regel Modus Ponens bij gesloten tableaus niet voor, maar drukt daarentegen een niet triviale eigenschap uit (zonder bewijs):

Indien er een gesloten tableau bestaat voor de sequent $\circ\, \varphi$ (φ is geldig) en ook een voor de sequent $\varphi \circ \psi$ (uit φ volgt ψ), dan bestaat er ook een gesloten tableau voor de sequent $\circ\, \psi$ (ψ is geldig).

Het gemis van Modus Ponens brengt een interessante kwestie aan het licht over de algemene structuur van bewijzen.

Sequentenbewijzen

Merk op dat men een gesloten tableau kan herinterpreteren als een bewijs door het van onder naar boven te lezen. Op die manier krijgt men een systeem voor het zogeheten 'sequentenbewijzen', dat sequenten in sequenten omzet via afleidingsregels als de volgende.

- Axiomatische sequenten (axioma's):

$\Sigma \circ \Delta$ mits $\Sigma \cap \Delta \neq \varnothing$ (sluiten)

- Opbouwregels, bijvoorbeeld de propositionele afleidingsregels voor de conjunctie $\wedge$:

$$\frac{\Sigma, \alpha, \beta \circ \Delta}{\Sigma, \alpha \wedge \beta \circ \Delta} \qquad \frac{\Sigma \circ \Delta, \alpha \qquad \Sigma \circ \Delta, \beta}{\Sigma \circ \Delta, \alpha \wedge \beta}$$

- Onder geschikte condities voor substitutie en vrij voorkomen krijgen we voor bijvoorbeeld de $\forall$–kwantor:

$$\frac{\Sigma, [t/x]\varphi \circ \Delta}{\Sigma, \forall x\, \varphi \circ \Delta} \qquad \frac{\Sigma \circ \Delta, \varphi}{\Sigma \circ \Delta, \forall y\, [y/x]\varphi}$$

Dit leidt tot een bewijssysteem voor de geldige sequenten zonder expliciete regel Modus Ponens. Hierdoor is het wat moeilijker te vergelijken met natuurlijke deductie en axiomatisch stellingbewijzen. We hebben echter gewonnen dat sequentenbewijzen meer expliciete informatie bevatten over geldige gevolgtrekkingen.
Een eenvoudig voorbeeld hiervan is het volgende. Bij een semantisch tableau worden formules in de topsequent geheel ontleed in hun subformules. Omgekeerd geldt blijkbaar dat de eindsequent $\Sigma \circ \Delta$ van een sequentenbewijs bewezen kan worden door het combineren van subformules van Σ en Δ.

STELLING 11.7

Subformule–eigenschap
Een bewijsbare gevolgtrekking heeft een sequentenbewijs waarin alleen regels worden gebruikt voor connectieven die ook werkelijk in die gevolgtrekking voorkomen.

In de andere bewijssystemen is dit vanwege de aanwezigheid van Modus Ponens of de →E–regel niet zo eenvoudig aan te tonen of zelfs onmogelijk.

Stelling van Gentzen

Combinatorische equivalentie tussen bewijzen met natuurlijke deductie en bewijzen via de sequentenversie van gesloten tableaus is dus een inhoudrijk resultaat. Toch blijkt die equivalentie aan te tonen, en dit staat bekend als de *stelling van Gentzen*. Het bewijs van die stelling vereist subtiele bewijstheoretische methoden die ook steeds vaker in de informaticaliteratuur opduiken.

Ten slotte kunnen we dus vaststellen dat de excursie via semantische volledigheid overbodig is om de verschillende bewijstheorieën voor de predikaatlogica met elkaar te vergelijken.

11.5 DEFINIEERBAARHEID

We besluiten dit hoofdstuk met een theoretisch onderwerp waarbij het verband tussen semantiek en bewijsbaarheid centraal staat: *definieerbaarheid*. Een belangrijke kwestie in predikaatlogische bewijzen is de definieerbaarheid van begrippen in theorieën.

Voorbeeld 11.1

In de rekenkunde kunnen we deling definiëren in termen van vermenigvuldiging: 'x is een deler van y': $\exists z\,(x \cdot z = y)$. Maar bijvoorbeeld vermenigvuldiging is niet definieerbaar in termen van optelling.

Formeel zeggen we:

DEFINITIE 11.2

Expliciet en impliciet definiëren
Laat Σ een theorie zijn in een taal met vocabulaire $P_1, ..., P_n, Q$ van predikaatletters. Zeg Q is k–plaatsig.

a Σ *definieert* Q *expliciet* als er een formule δ bestaat waarin hoogstens $P_1, ..., P_n$ voorkomen en Q niet, met vrije variabelen $x_1, ..., x_k$, zodat $\Sigma \vdash \forall x_1 ... \forall x_k\,(Q(x_1, ..., x_k) \leftrightarrow \delta)$.

b Σ *definieert* Q *impliciet* als er geen twee modellen van Σ zijn die alleen verschillen in hun interpretatie van Q.

Expliciete definieerbaarheid is de syntactische notie van definiëren. Als een theorie Σ een predikaat Q expliciet definieert, dan kunnen we die theorie vervangen door de volgende: vervang in alle axioma's alle voorkomens van Q door de definitie van Q en voeg aan de nieuwe axiomatiek ten slotte nog de expliciete definitie van Q toe.
Impliciete definieerbaarheid is de parallelle semantische notie van definiëren. Dat Σ impliciet Q definieert, betekent dat de interpretaties voor $P_1, ..., P_n$ de interpretatie voor Q reeds vastleggen.

Een voor de hand liggende vraag betreft de relatie tussen impliciete en expliciete definieerbaarheid. Eén richting blijkt eenvoudig te bewijzen:

STELLING 11.8

Σ definieert Q expliciet $\Rightarrow$ Σ definieert Q impliciet.

Bewijs

Stel dat $M_1 = (\boldsymbol{D}, I_1)$ en $M_2 = (\boldsymbol{D}, I_2)$ twee modellen van Σ zijn waarin I_1 hoogstens van I_2 verschilt wat betreft Q. Uit $\Sigma \vdash \forall x_1 \ldots \forall x_k\,(Q(x_1, ..., x_k) \leftrightarrow \delta)$ volgt met correctheid $\Sigma \models \forall x_1 \ldots \forall x_k\,(Q(x_1, ..., x_k) \leftrightarrow \delta)$. Derhalve is de equivalentie $Q(x_1, ..., x_k) \leftrightarrow \delta$ in beide modellen waar. Aangezien δ in M_1 en M_2 hetzelfde predikaat definieert, moeten $I_1(Q)$ en $I_2(Q)$ dan toch opgaan voor dezelfde rijtjes van individuen in het domein, dat wil zeggen I_1 verschilt niet van I_2 wat betreft Q. □

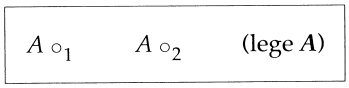
Met behulp van deze stelling is expliciete definieerbaarheid vaak te *weerleggen*:

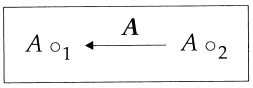
Voorbeeld 11.2

In de natuurlijke taal is er vaak een verband tussen bijvoeglijke naamwoorden die eigenschappen aangeven als 'groot' en de bijbehorende comparatieve relaties als 'groter (dan)'. Hier zijn enkele van die verbanden, weergegeven in een theorie Σ met predikaten 'A' (eenplaatsig) en '$\boldsymbol{A}$' (tweeplaatsig). Denk bij de tweede relatie aan 'A–er dan'.

1	$\forall x\,\forall y\,(\boldsymbol{A}xy \rightarrow \neg \boldsymbol{A}yx)$	(asymmetrie)
2	$\forall x\,\forall y\,\forall z\,((\boldsymbol{A}xy \wedge \boldsymbol{A}yz) \rightarrow \boldsymbol{A}xz)$	(transitiviteit)
3	$\forall x\,\forall y\,(\boldsymbol{A}xy \rightarrow \forall z\,(\boldsymbol{A}xz \vee \boldsymbol{A}zy))$	(zwakke lineariteit)
4	$\forall x\,(Ax \rightarrow \forall y\,(\boldsymbol{A}yx \rightarrow Ay))$	
5	$\forall x\,\forall y\,((Ax \wedge \neg Ay) \rightarrow \boldsymbol{A}xy)$	

In de klassieke logica dacht men wel dat zo'n (tweeplaatsig) comparatief expliciet te definieren valt uit de corresponderende (eenplaatsige) eigenschap, zodat men op grond daarvan voor alle logische doeleinden met monadische predikaatlogica (zie hoofdstuk 8) kon volstaan. Dat dit echter niet kan, tonen de volgende twee modellen voor Σ aan:





Deze modellen geven dezelfde interpretatie aan de eigenschap A en verschillen alleen in hun interpretatie van het comparatief $\boldsymbol{A}$. Deze theorie definieert $\boldsymbol{A}$ dus niet impliciet, en op grond van de zojuist bewezen stelling dus ook niet expliciet.

De omkering van stelling 11.8 geldt ook, maar is een veel dieper resultaat:

STELLING 11.9

Stelling van Beth
Σ definieert Q impliciet $\Rightarrow$ Σ definieert Q expliciet

11.6 OPGAVEN

11.1 Bewijs de $\exists$–stap in het bewijs van bewering 11.1 van het adequaatheidsbewijs.

11.2 Bewijs de $\forall$–stap in het bewijs van de correctheidsstelling.

11.3 Bepaal voor elk van de volgende formules of de voorkomens van P daarin positief zijn:
i $\forall x\,((Px \wedge Rx) \vee (Px \rightarrow Sx))$
ii $\neg\exists x\,((Rx \wedge Px) \rightarrow \forall y\,(Sy \rightarrow \neg Py))$
iii $\forall x\,(((Px \rightarrow Rx) \rightarrow Px) \rightarrow Px)$
iv $\neg\forall x\,((Px \wedge Qx) \rightarrow (Sx \rightarrow Px))$

11.4 Geef een sequentenbewijs voor $(\varphi \rightarrow (\psi \rightarrow \chi)) \vdash (\varphi \rightarrow \psi) \rightarrow (\varphi \rightarrow \chi)$.

11.5 Weerleg met een semantisch tableau dat $\boldsymbol{A}$ expliciet definieerbaar is in termen van A in de theorie over comparatieven.

* 11.6 Bewijs de volgende bewering uit het volledigheidsbewijs: $\xi \in \Gamma^* \Leftrightarrow M, b \vDash \xi$, met Γ^*, M en b als in het bewijs gegeven.

Blok IV

Voorbij de predikaatlogica

Voorbij de predikaatlogica

Hoofdstuk 12

Voorbij de predikaatlogica

12.1 INLEIDING

De predikaatlogica die is behandeld in het voorgaande blok van dit boek, wordt vaak beschouwd als de 'standaard'–logica. We hebben nu immers een formele taal met grote uitdrukkingskracht, voorzien van een exacte semantiek en van uiteenlopende manieren om geldig redeneren te karakteriseren.
Toch wil dit niet zeggen dat alles in dit standaardkader moet passen, maar veeleer dat, met dit voorbeeld van precisie voor ogen, verdere logische systemen ontwikkeld zijn, die aangepast zijn aan de speciale behoeften van een beoogde toepassing.
Er zijn ruwweg drie richtingen voor verdere ontwikkeling:
a variaties op de predikaatlogica;
b uitbreidingen van de predikaatlogica;
c alternatieven voor de predikaatlogica.
In dit hoofdstuk behandelen we een aantal onderwerpen ter illustratie van elk van deze richtingen, achtereenvolgens in paragrafen 12.2–12.4 (a), 12.5–12.9 (b) en 12.10 (c). Steeds wordt één onderwerp benadrukt: als variatie de equationele logica, als uitbreiding de lambda–calculus en als alternatief de intuïtionistische logica.

12.2 VARIATIES OP DE PREDIKAATLOGICA

Notatie

Om te beginnen is er niets dogmatisch aan de predikaatlogische *notatie* zoals die tot nu toe is gehanteerd. Bijvoorbeeld, diverse symbolen of plaatsingsconventies voor logische operatoren zijn aanvaardbaar. Zo zijn bijvoorbeeld zowel &, $\wedge$, als AND gebruikelijke symbolen voor de conjunctie, en kunnen in het algemeen logische connectieven zowel prefix als infix geschreven worden. Evenzo kan men de logische symbolen zelf op verschillende manieren kiezen, getuige de diverse functioneel volledige stellen connectieven in hoofdstuk 2. Het is natuurlijk wel zaak van een eenmaal gekozen notatie niet meer af te wijken.

Meersoortige predikaatlogica

Ingrijpender dan zulke notatiekwesties zijn veranderingen in het systeem om beweringen gemakkelijker te kunnen formaliseren. Vaak moeten we kwantificeren over een 'inhomogeen' domein van individuen, waarin zich zulke uiteenlopende objecten kunnen bevinden als personen, getallen, tijdstippen of machineregisters. Het is dan heel handig om over te gaan op een zogenaamde *meersoortige* predikaatlogica, waarin kwantoren gerelateerd worden aan zekere 'soorten' of 'subdomeinen' van het totale domein van individuen. De winst aan overzichtelijkheid zien we in een simpel voorbeeld als het volgende 'tweesoortige' geval:

$\forall x\, \exists t\, Zxt$ ('iedereen is wel eens ziek')

Zxt staat voor 'x is ziek op tijdstip t'. De variabele x loopt over personen en t over tijdstippen. We kiezen t in plaats van y om te benadrukken dat de variabele t van een andere soort is dan x.

Eensoortige notatie

In de ons vertrouwde 'eensoortige' notatie zouden we definiërende predikaten voor deze twee soorten moeten gebruiken en is de formalisering als volgt:

$\forall x\, (Px \rightarrow \exists y\, (Ty \wedge Zxy))$

Daarin staat Px voor 'object x is een persoon' en Ty voor 'object y is een tijdstip'.
Merk overigens op dat deze formalisering equivalent is met de notatie waarin we eenplaatsige predikaten niet als eigenschappen maar als verzamelingen opvatten. Bij die equivalente notatie spreken we dus niet van meersoortigheid:

$\forall x\, (x \in P \rightarrow \exists y\, (y \in T \wedge Zxy))$

Dit wordt nog wel korter geschreven als:

$(\forall x \in P)(\exists y \in T)\, Zxy$

Dat we meersoortige logica opvatten als een variatie op de predikaatlogica en niet als een uitbreiding ervan, heeft twee redenen. Ten eerste zijn alle eerdere begrippen en resultaten te generaliseren naar een meersoortige taal, en ten tweede is deze taal blijkbaar tot de standaardpredikaatlogica herleidbaar.

12.3 VARIATIES: PREDIKAATLOGICA MET IDENTITEIT

Identiteit

De *identiteit* of *gelijkheid* is een speciaal predikaat dat meestal infix wordt geschreven en aan bijzondere eigenschappen voldoet. Het wordt geïnterpreteerd als de *echte* identiteit in modellen, zie hoofdstuk 7. De predikaatlogische onderwerpen kunnen we tot identiteiten met termen specialiseren. Zo kan men semantische tableaus maken, voorzien van louter identiteitsregels die termen identificeren. Ook kan natuurlijke deductie worden aangevuld met identiteitsregels.

Gelijkheidsaxioma's

Het meest overzichtelijk is om de kenmerkende principes voor identiteit apart op te schrijven in een calculus:

DEFINITIE 12.1

Axioma's voor gelijkheid van termen

a	$\vdash t = t$	reflexiviteit
b	$t_1 = t_2 \vdash t_2 = t_1$	symmetrie
c	$t_1 = t_2, t_2 = t_3 \vdash t_1 = t_3$	transitiviteit
d	$t_1 = t_1', ..., t_n = t_n' \vdash g(t_1, ..., t_n) = g(t_1', ..., t_n')$	substitutie

Deze axiomatiek is volledig met betrekking tot het afleiden van semantisch geldige gevolgtrekkingen met identiteiten. We geven die stelling zonder bewijs:

STELLING 12.1

Een gevolgtrekking $t_1 = t_1', ..., t_n = t_n' / t = t'$ is semantisch geldig dan en slechts dan als zij afleidbaar is met de gelijkheidsaxioma's.

Omdat de identiteit feitelijk een predikaat is, spreken we ook hier niet van een uitbreiding, ondanks de extra gelijkheidsaxioma's.

12.4 VARIATIES: DEELTALEN EN EQUATIONELE LOGICA

Fragmenten van de predikaatlogica

Interessante variaties ontstaan door de *beperking* van de predikaatlogica tot deeltalen. Voorbeelden hiervan zijn het monadische en het universele fragment uit hoofdstuk 8. De winst van zo'n beperking schuilt onder meer in bijzondere logische eigenschappen: zo kan het geldigheidsprobleem voor een fragment veel eenvoudiger liggen dan voor de predikaatlogica in haar geheel.

Equationele logica

Een belangrijk deelsysteem van de predikaatlogica ontstaat wanneer we alleen werken met *identiteitsbeweringen* tussen termen, en geen andere predikaten toestaan. De identiteitsbeweringen worden steeds *universeel gekwantificeerd* gelezen. Dit deelsysteem wordt *equationele logica*

genoemd, en omdat behalve de identiteit geen predikaten in de taal voorkomen, spreken we van een *algebraïsche* deeltaal.

Algebra

De bijbehorende semantische structuur is de *algebra* (dit is hetzelfde als de operationele structuur, zie hoofdstuk 7): een equationele logica wordt geïnterpreteerd als een algebra. Merk op dat de identiteit wordt geïnterpreteerd als de echte identiteit. Verder identificeren we voor het gemak functiesymbolen in de formele *taal* met informele aanduidingen voor de operaties in algebraïsche *structuren*.

Similariteitstype

Behalve door specifieke axioma's, kunnen klassen algebra's in eerste instantie van elkaar onderscheiden worden door het aantal en de plaatsigheid van hun operatoren. Zoals eerder reeds is opgemerkt, kunnen in operationele structuren nulplaatsige functies gebruikt worden als 'uitverkoren objecten' om individuele constanten te interpreteren. Een rijtje $\langle a, b, c, ... \rangle$ dat de plaatsigheden opsomt van de uitverkoren objecten en operaties van een algebra, noemen we het *similariteitstype* van die algebra. Het is gebruikelijk om het rijtje te ordenen naar oplopende plaatsigheid.

Specifieke genres algebra's ontstaan door de keuze van een similariteitstype en een aantal axioma's. (Aan het begin van een axioma laten we universele kwantoren voor alle in dat axioma voorkomende variabelen steeds weg). We geven een aantal voorbeelden:

Voorbeeld 12.1

Groepen

Een *groep* is een algebra $\langle A, e, ^{-1}, \circ \rangle$, waarbij $\circ$ een tweeplaatsige functie (groepsbewerking) is, $^{-1}$ een eenplaatsige functie (inverse) en e een nulplaatsige functie (eenheidselement). Het similariteitstype is dus: $\langle 0, 1, 2 \rangle$. Bovendien moet aan de volgende groepsaxioma's voldaan zijn:

G1	$(x \circ y) \circ z = x \circ (y \circ z)$	associativiteit
G2	$x \circ e = x$	rechter eenheid
G3	$e \circ x = x$	linker eenheid
G4	$x \circ x^{-1} = e$	rechts inverse
G5	$x^{-1} \circ x = e$	links inverse

Een concrete groep ontstaat zodra we een specifiek domein A aangeven, met specifieke operaties daarop: bijvoorbeeld $A = \mathbb{R}$, met $x \circ y = x + y$, $x^{-1} = -x$ en $e = 0$.

Andere wiskundige voorbeelden zijn 'ringen' en 'lichamen', die overigens aan nog meer predikaatlogische axioma's voldoen dan groepen in het algemeen, en een uitgebreider similariteitstype hebben met onder meer twee binaire operaties.

Voorbeeld 12.2

Boolese algebra
Ook binnen de logica zelf bestaan bekende algebraïsche structuren. Een belangrijk voorbeeld hiervan zijn *Boolese algebra's*, die sterk verwant zijn aan de propositielogica als structuur. Het similariteitstype is $\langle 0, 0, 1, 2, 2\rangle$, weerspiegeld in een taal met twee individuele constanten 0 (onwaar) en 1 (waar), een eenplaatsig functiesymbool – (complement) en twee tweeplaatsige functiesymbolen + (disjunctie) en · (conjunctie). De volgende axioma's regelen deze noties:

B1	$x + y = y + x$	commutativiteit
	$x \cdot y = y \cdot x$	
B2	$(x + y) + z = x + (y + z)$	associativiteit
	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	
B3	$(x + y) \cdot y = y$	absorptie
	$(x \cdot y) + y = y$	
B4	$x + (-x) = 1$	complement
	$x \cdot (-x) = 0$	
B5	$(x + y) \cdot z = (x \cdot z) + (y \cdot z)$	distributiviteit
	$(x \cdot y) + z = (x + z) \cdot (y + z)$	

Hier zijn enkele specifieke voorbeelden van Boolese algebra's:
- Het domein is de machtsverzameling $P(U)$ van een verzameling U.

Interpreteer de Boolese axioma's als volgt:

+	vereniging
·	doorsnede
–	complement
0	lege verzameling
1	universum

- Laat het domein bestaan uit alle gehele delers van 30 (inclusief 30 zelf en 1), met de interpretaties:

+	kleinste gemene veelvoud
·	grootste gemene deler
–	de functie $30/x$
0	1
1	30

Merk op dat de 0 hier als 1 wordt geïnterpreteerd, wat natuurlijk puur een kwestie is van conventie.

- Stel de objecten zijn equivalentieklassen onder logische equivalentie van propositielogische formules. De functiesymbolen kunnen dan voor de hand liggend geïnterpreteerd worden door middel van de syntax van de propositielogica, bijvoorbeeld: het complement van de klasse die φ bevat, is de klasse die $\neg\varphi$ bevat.

Boolese algebra's in de informatica

In de informatica komen Boolese structuren vaak voor, zowel in de software als in de hardware. Programmeertalen hebben doorgaans Boolese operaties in de tests die het rekenproces sturen: OR, AND, NOT. Maar elektronische apparatuur zelf heeft ook logische componenten, zoals AND– en OR–poorten:

Boolese algebra's: elektrische netwerken

Een voorbeeld van een hardware–realisering van Boolese algebra zijn elektrische *netwerken*. De analogie met de Boolese operaties is nu als volgt:

·	serieschakeling
+	parallelschakeling
–	complementaire schakeling

Met behulp van de Boolese axioma's kunnen we netwerken *ontwerpen* of *vereenvoudigen*.

Voorbeeld 12.3

Relatiealgebra

Bij Boolese algebra kan men denken aan een calculus van operaties op verzamelingen. Verzamelingen zijn op te vatten als – eenplaatsige – *eigenschappen* van individuen. Daarmee ligt dit onderwerp in de sfeer van de monadische predikaatlogica. Het algebraïsche gezichtspunt binnen de logica valt ook uit te breiden naar een calculus op tweeplaatsige *relaties* en natuurlijke operaties daarover. Zo ontstaat de *relatiealgebra*. Die wordt ook binnen de informatica veel gebruikt, bijvoorbeeld in de semantiek van programmaconstructies in programmeertalen (zie hoofdstuk 15) en bij het combineren van predikaten tijdens het bevragen van gegevensbestanden. De eerdere Boolese operaties zijn natuurlijk ook op binaire relaties (opgevat als verzamelingen) van toepassing. Bovendien introduceren we drie nieuwe operaties:

$R \circ S = \{(u, v) \mid \exists w$: Ruw en $Swv\}$	compositie
$R^{\smile} = \{(u, v) \mid (v, u) \in R\}$	converse
$I = \{(u, v) \mid u = v\}$	identiteit

Het similariteitstype van de relatiealgebra is dus $\langle 0, 0, 0, 1, 1, 2, 2, 2\rangle$. Behalve de Boolese axioma's zijn de kenmerkende axioma's van de relatiealgebra:

a $X \circ I = X = I \circ X$
b $X^{\smile\smile} = X$
c $(-X)^{\smile} = -X^{\smile}$
d $(X \cdot Y)^{\smile} = X^{\smile} \cdot Y^{\smile}$
e $(X \circ Y) \circ Z = X \circ (Y \circ Z)$
f $X \circ (Y + Z) = (X \circ Y) + (X \circ Z)$
$(X + Y) \circ Z = (X \circ Z) + (Y \circ Z)$
g $(X \circ Y)^{\smile} = Y^{\smile} \circ X^{\smile}$
h $(X^{\smile} \circ -(X \circ Y)) \cdot -Y = (X^{\smile} \circ -(X \circ Y))$

Nog wat concreter kan men zich bijvoorbeeld de volgende relatiealgebra voorstellen:

Voorbeeld 12.4

Toepassing van een relatiealgebra
Het tramcircuit van een stad kan met behulp van relaties worden weergegeven:

$T_i xy \Leftrightarrow$ het is mogelijk om met tramlijn i van straat x naar straat y te gaan

Gegeven deze interpretatie kunt u nu de verzamelingsoperaties interpreteren en de axioma's nagaan. Een aanzet hiertoe:
- De relatie $T_i \circ T_j$ bestaat nu uit die stratenparen (x, y) waarvoor geldt dat straat y bereikbaar is vanaf straat x via overstappen van lijn i op lijn j. Compositie van meerdere tramlijnen geeft dus stratenparen die, via (eventueel) meerdere keren overstappen, met elkaar door een tramlijn verbonden zijn. Het is nu eenvoudig in te zien dat axioma e geldig is.
- Merk op dat niet noodzakelijk geldt $T_i = T_i^{\smile}$. Immers, de heen– en terugroutes tussen begin– en eindpunt van lijn i kunnen verschillen. Axioma b is natuurlijk wel geldig.
- De relatie $T_i \cdot T_j$ bestaat uit stratenparen (x, y) waarvoor geldt dat zowel lijn i als lijn j van straat x naar straat y gaat.

Voorbeeld 12.5

Procesalgebra
Zowel Boolese algebra als relatiealgebra werden reeds in de negentiende eeuw ontwikkeld. Maar ook meer recentelijk zijn binnen de informatica zelf algebraïsche structuren heel gebruikelijk geworden. Een voorbeeld is de zogenaamde *procesalgebra*, die het gedrag van parallelle processen beschrijft. We geven enkele kenmerkende principes van deze theorie, met als interpretatie:

$+$ keuze tussen twee processen
$\cdot$ achtereenvolgens uitvoeren
$||$ parallel uitvoeren
$|\!\llcorner$ 'left merge': als $||$, maar met verplicht de eerste stap uit het linker proces gekozen.

Enkele axioma's zijn:
- $(x + y) \cdot z = x \cdot z + y \cdot z$
- $x \,||\, y = x \,|\!\llcorner\, y + y \,|\!\llcorner\, x$
- $(x + y) \,|\!\llcorner\, z = x \,|\!\llcorner\, z + y \,|\!\llcorner\, z$

Universele algebra

Ten slotte nog een theoretische kwestie over het algemene verband tussen algebra's (modellen) en equationele logica (logische theorieën). Zoals we eerder hebben gezien, kenmerken fragmenten van de predikaatlogica zich vaak door speciaal semantisch gedrag. Nu zijn uit de algebra enkele belangrijke operaties bekend die uit structuren nieuwe structuren maken, zoals het nemen van 'subalgebra's', 'productalgebra's' en 'homomorfe beelden'. Deze keren overigens binnen de informatica terug als fundamentele operaties op gegevensstructuren. Wat deze operaties gemeen hebben, is dat algebraïsche gelijkheden ook geldig blijven in de beelden van een algebra onder zulke operaties. Voor de informatica betekent dit dat algebraïsche specificaties van gegevensstructuren behouden blijven onder deze operaties.
Een fundamenteel resultaat over het verband tussen die structuureigenschappen en de logica (dit vakgebied heet *Universele algebra*) is de volgende 'Stelling van Birkhoff':

Een predikaatlogische formule is definieerbaar met alleen identiteiten tussen termen dan en slechts dan als zij bewaard blijft onder de drie genoemde algebraïsche operaties.

12.5 UITBREIDINGEN VAN DE PREDIKAATLOGICA: HOGERE–ORDE LOGICA

Na het 'verkleinen' gaan we nu 'vergroten'. Echte versterkingen van de predikaatlogica blijken ook nodig, zowel voor wiskundige doeleinden als bij computationele en taalkundige toepassingen.

Een schoolvoorbeeld is de zogenaamde *hogere–orde logica*. In de predikaatlogica tot nu toe hebben we ons beperkt tot individuele objecten, met predikaten en functies daarover. De belangrijkste logische operatie, te weten kwantificatie, liep alleen over het domein der individuen. Maar in de praktijk willen we ook kwantificeren over predikaten van individuen of functies van individuen naar individuen. Dit kan men bereiken door te kwantificeren over deelverzamelingen van het domein, en andere 'hogere objecten'.

Voorbeeld 12.6

Natuurlijke inductie
Een voorbeeld is het eerdere principe van natuurlijke inductie, dat intuïtief iets zegt over alle eigenschappen E (eenplaatsige predikaten) van natuurlijke getallen:

$$\forall E\,((E(0) \wedge \forall n\,(E(n) \rightarrow E(n+1))) \rightarrow \forall n\,E(n))$$

Deze voor de hand liggende formulering heeft twee vormen van kwantificatie: één over individuele getallen ($\forall n$) en een tweede over eigenschappen van individuen ($\forall E$).

Voorbeeld 12.7

Principe van Leibniz
Een ander voorbeeld van hogere kwantificatie is de logische formulering van het 'Principe van Leibniz' dat twee individuen precies dan gelijk zijn als ze dezelfde eigenschappen hebben:

$$\forall x\,\forall y\,(x = y \leftrightarrow \forall P\,(Px \leftrightarrow Py))$$

DEFINITIE 12.2

Tweede–orde logica
De *tweede–orde logica* is predikaatlogica voorzien van extra kwantoren $\forall P$ over willekeurige predikaten P, en extra kwantoren $\forall f$ over willekeurige functies f.

Individuen worden objecten van de eerste orde genoemd. Predikaten over individuen worden objecten van de tweede orde genoemd. Men spreekt van tweede–orde logica omdat over ten hoogste objecten van de

tweede orde wordt gekwantificeerd. De standaardpredikaatlogica is dus de eerste–orde logica.
De tweede–orde logica breidt de standaardpredikaatlogica wezenlijk uit. Dit weerspiegelt zich in het optreden van echt andere eigenschappen. Bijvoorbeeld, de grotere uitdrukkingskracht wordt gekocht tegen de volgende prijs:

BEWERING 12.1

Onvolledigheid van de tweede–orde logica
Er bestaat *geen volledige bewijsmethode* om geldig redeneren in de tweede–orde logica uitputtend te beschrijven.

Hogere–orde logica

De tweede–orde logica is overigens niet het einde, maar veeleer het begin van een ladder van mogelijkheden. Hogere ordes zijn namelijk heel goed denkbaar: bijvoorbeeld met predikaten over predikaten. Een voorbeeld uit de wiskunde zijn de zogenaamde functionalen: functies over functies, zoals differentiaal– of integraaloperatoren. Een voorbeeld uit de informatica zijn programmatransformaties op programma's, welke laatste zelf al overgangsfuncties vormen op een rekenautomaat. Er zijn ook voorbeelden in ons gewone taalgebruik:

Voorbeeld 12.8

Predikaten van hogere orde in natuurlijke taal
Indien we ons voorstellen dat bijvoorbeeld *eigennamen* zoals 'Marie' en 'De Eiffeltoren' objecten van eerste orde aanduiden, dan geven *werkwoorden* zoals 'werkt' en 'wankelt' eigenschappen van zulke objecten weer. De grammaticale categorie van de *bijwoorden*, zoals 'gestaag' en 'gevaarlijk', opereert weer op werkwoorden en is dus een object van derde orde. Dit proces valt verder voort te zetten met *modificatoren* als 'zeer' en 'uiterst', die weer op bijwoorden opereren en dus objecten van de vierde orde zijn.

In principe hebben we dan logische formalismen nodig die predikaten van elke eindige orde bevatten. Een elegante manier om dit te bereiken is de zogenaamde *lambda–calculus*. Daaraan wijden we hier een aparte korte beschouwing.

12.6 LAMBDA–CALCULUS

Functies

De λ–calculus werd ontwikkeld terwille van een goede formalisering van het centrale begrip *functie* in de grondslagen van de wiskunde, een begrip dat eveneens centraal is gebleken in de informatica. Met functie samenhangende kernbegrippen zijn: *toepassing* of *applicatie* van een functie op een argument, alsmede het creëren van een functie door zogenaamde *abstractie* uit een voorschrift.

λ–Operator

De λ–calculus is een technisch instrument om precies over functies te spreken. Beschouw bijvoorbeeld de uitdrukking $3x + y$. Van deze uitdrukking kunnen verschillende functies gemaakt worden. Om deze verschillen duidelijk te maken, introduceren we een nieuwe notatie, waarin een nieuw symbool voorkomt: de *λ–operator*. De λ–operator functioneert als volgt:

Voorbeeld 12.9

Lambda–abstractie
Door voor een uitdrukking als $3x + y$, met twee vrije variabelen of parameters, het symbool λ te zetten, gevolgd door één of meer variabelen, maken we van dit 'berekeningsvoorschrift' een functie, met als argumenten de achter λ voorkomende variabelen. Om het voorschrift te scheiden van deze variabelen, zetten we er een punt tussen.
Dit levert bijvoorbeeld de volgende functies:

- $(\lambda x \,.\, 3x + y)$ eenplaatsig
- $(\lambda y \,.\, 3x + y)$ eenplaatsig
- $(\lambda x \,.\, (\lambda y \,.\, 3x + y))$ tweeplaatsig

Toepassing van deze functies op een argument gaat als volgt:

- $((\lambda x \,.\, 3x + y)\ 2) = 3 \cdot 2 + y = 6 + y$
- $((\lambda y \,.\, 3x + y)\ 2) = 3x + 2$
- $((\lambda x \,.\, (\lambda y \,.\, 3x + y))\ (2, 5)) = 3 \cdot 2 + 5 = 11$

De uitdrukking $(\lambda x \,.\, (\lambda y \,.\, 3x + y))$ is ook te beschouwen als een functie van getallen naar functies: toepassing op een x–waarde alleen levert een functie van y:

- $((\lambda x \,.\, (\lambda y \,.\, 3x + y))\ 4) = (\lambda y \,.\, 3 \cdot 4 + y)$

λ–Abstractie

Het vormen van de functie $(\lambda x \,.\, P)$ bij een gegeven uitdrukking P, heet *λ–abstractie*. Abstraheren kan behalve op wiskundige berekeningsvoorschriften ook op andere uitdrukkingen toegepast worden, zoals beweringen, en ook op niet–wiskundige uitdrukkingen. Het oorspronkelijke werkterrein van de λ–calculus kan op die manier sterk worden uitgebreid.

Voorbeeld 12.10

Abstractie op logische beweringen
Laten we formules opvatten als uitdrukkingen die een waarheidswaarde opleveren, zoals gebeurde in de hoofdstukken 2 en 7. Predikaten zijn dan functies van individuele objecten naar waarheidswaarden. Met abstractie kunnen we uit formules bijbehorende predikaten maken:
De functie $(\lambda x \,.\, (\lambda y \,.\, x < y))$ is een tweeplaatsige functie van getallenparen naar waarheidswaarden. Deze functie representeert de relatie 'kleiner dan'.

Voorbeeld 12.11

Abstractie op beweringen in gewone taal
We kunnen hetzelfde doen met beweringen in gewone taal:
Toegepast op 'x is gezond' levert λ–abstractie: $(\lambda x \,.\, x$ is gezond).
Dit is een functie van bijvoorbeeld personen naar waarheidswaarden, die staat voor de eigenschap 'gezond'. Immers, toepassing op een argument levert een bewering die waarheidswaarde 0 of 1 heeft:

$((\lambda x \,.\, x$ is gezond) Guus) = Guus is gezond

Niets verhindert ons om te abstraheren vanuit andere zinsposities! Zo definieert $(\lambda P \,.\, P(\text{Guus}))$ een eigenschap van predikaten: namelijk 'gelden voor Guus'. En evenzo betekent $(\lambda P \,.\, \forall x\, (Qx \rightarrow Px))$: 'gelden voor iedere Q'.

We introduceren de λ–calculus nu formeel:

DEFINITIE 12.3

Taal en syntaxis van de λ–calculus
Het alfabet van de λ–calculus bestaat uit:
a variabelen: $x, y, z, \ldots$
b constanten: $a, b, c, \ldots$
c één logisch symbool: λ
d hulpsymbolen: .,) en (

De syntaxis geeft een inductieve opbouw van *termen*:

e elke variabele en elke constante is een term

f als t_1 en t_2 termen zijn, dan ook $(t_1\ t_2)$ (applicatie)

g als t een term is en x een variabele, dan is $(\lambda x\ .\ t)$ een term (abstractie)

Voorbeeld 12.12

Termen

- $(a\ x)$
- $(\lambda x\ .\ (x\ a))$
- $(\lambda x\ .\ ((\lambda y\ .\ (b\ (y\ x)))\ (\lambda z\ .\ z)))$

Er zijn overigens ook andere notatieconventies in omloop, met haakjes ingezet op andere plaatsen. Zo schrijft men applicaties ook vaak meer traditioneel als $t_1(t_2)$ – 'functie t_1 toegepast op argument t_2' – of zonder haakjes als $t_1\ t_2$.

λ–Conversie

De bedoelde interpretatie van termen is als *functies*. Functies kunnen zelf ook weer argument voor een functie zijn. Gegeven deze semantische interpretatie, die we hier overigens niet verder formeel zullen uitwerken, liggen enkele *bewijsregels* voor de hand om te opereren met identiteiten tussen lambda–termen. Zo mogen we 'gelijken vervangen door gelijken', net als in de equationele logica. Het meest kenmerkend en nuttig is een omvorming die bekendstaat als *conversie*. Deze regelt het functioneel gedrag van abstractietermen:

DEFINITIE 12.4

λ–conversie

$((\lambda x\ .\ t_1)\ t_2) = [t_2/x]t_1$ mits t_2 vrij voor x in t_1

In woorden: toepassen van de functie $(\lambda x\ .\ t_1)$ op een argument t_2 heeft hetzelfde resultaat als substitutie van t_2 voor x in t_1. De conditie is nodig omdat λ–bindingen dezelfde problemen kunnen vertonen als de eerdere kwantorbindingen van hoofdstuk 7.

Voorbeeld 12.13

- $((\lambda x\ .\ (f\ x))\ a) = [a/x](f\ x) = (f\ a)$
- $((\lambda x\ .\ (\lambda y\ .\ (x\ (g\ y))))\ f) = [f/x](\lambda y\ .\ (x\ (g\ y))) = (\lambda y\ .\ (f\ (g\ y)))$

λ–Calculus

Tezamen met nog enkele eenvoudige reducties voor identiteiten ontstaat aldus een systeem van rekenregels om λ–termen te reduceren tot eenvoudiger vormen. Dat uiteindelijke systeem heet de λ–calculus.

λ–Reductie

We zeggen dat term t_1 *reduceert tot* t_2, indien t_2 uit t_1 te verkrijgen is door λ-conversiestappen.

In de λ–calculus kan verrassend veel worden uitgedrukt. Men kan namelijk aantonen dat alle effectief berekenbare functies (hierover meer in hoofdstuk 14) in de λ–calculus gesimuleerd kunnen worden. Niettemin is de voornaamste functie van de λ–calculus in de meeste toepassingen toch eerder om de combinatorische mogelijkheden van bestaande formalismen uit te breiden. Voorbeelden hiervan uit de informatica en de taalkunde zullen we verderop kort bespreken.

Theorie van de λ–calculus: normaalvormen

Klassieke resultaten over de λ–calculus ademen een iets andere sfeer dan wat we tot nu toe bij onze logische systemen tegenkwamen. Ze betreffen veelal eigenaardigheden van reducties. Een belangrijke rol spelen de termen die niet verder met behulp van conversie te reduceren zijn. Zulke termen heten *normaalvormen*.

Voorbeeld 12.14

- De term $(((\lambda x\,.\,(x\ x))\ (\lambda y\,.\,y))\ z)$ reduceert eerst tot $(((\lambda y\,.\,y)\ (\lambda y\,.\,y))\ z)$, en die weer tot $((\lambda y\,.\,y)\ z)$ en die ten slotte tot z: een normaalvorm.
- De term $(x\ (\lambda y\,.\,y))$ is niet verder te reduceren en is dus in normaalvorm.
- Niet elke term heeft overigens een normaalvorm:
$((\lambda x\,.\,(x\ x))\ (\lambda x\,.\,(x\ x)))$ reduceert tot zichzelf en deze reductie stopt dus nooit.

Een van de fundamentele theoretische resultaten over de λ–calculus is dat normaalvormen uniek zijn, op alfabetische varianten in hun variabelen na. Dit volgt uit een andere (onbewezen) eigenschap van de λ–calculus:

STELLING 12.2

Confluentie
Als een term t reduceert tot zowel t_1 als t_2, dan is er een term t' waartoe beide weer te reduceren zijn.

Het computationele belang van confluentie – 'samenloop' – wordt ook in de informatica benadrukt: we kunnen kennelijk indeterministisch reduceren, zonder de weg naar het gewenste eindresultaat te verliezen.

Voorbeeld 12.15

Confluentie
Beschouw de term $((\lambda x\,.\,((\lambda y\,.\,(y\ x))\ a))\ b)$. Deze term kan gereduceerd worden tot $((\lambda y\,.\,(y\ b))\ a)$ en tot $((\lambda x\,.\,(a\ x))\ b)$, maar elk van deze twee termen reduceert weer tot $(a\ b)$.

Functioneel programmeren

De λ–calculus wordt toegepast in programmeertalen waarin declaratief functies worden gedefinieerd door middel van stelsels vergelijkingen, die eventueel recursief de te definiëren functies aanroepen.

LISP

Een bekend voorbeeld van deze zogenaamde functionele programmeertalen is LISP ('LISt Processing'). Een programmaregel in LISP is een gestructureerde lijst van functies. De structuur van zo'n lijst komt sterk overeen met de structuur van een term uit de λ–calculus.
LISP kan met enige historische rechtvaardiging worden beschouwd als een implementatie van de λ–calculus. Er zijn overigens wel diverse hulpfuncties toegevoegd aan LISP terwille van verhoogde bruikbaarheid, zoals conditionele uitdrukkingen. Ook niet–declaratieve hulpfuncties zoals bedelingen zijn toegevoegd. Wij zullen niet verder ingaan op LISP.

12.7 GETYPEERDE LAMBDA–CALCULUS

De zojuist gepresenteerde versie van de λ–calculus komt het meest voor in de theoretische literatuur. Niettemin zijn er ook vele toepassingsgebieden waar we het formalisme aan strengere eisen willen laten voldoen, met name aan de eis dat een functie slechts kan opereren op argumenten die 'eenvoudiger' zijn dan die functie zelf. Een natuurlijke manier om dit te bereiken is door middel van *typering*. Hierbij worden alle objecten en termen ingedeeld naar zogenaamde *typen*, die als het ware verschillende niveaus van het universum beschrijven. Typering is een belangrijk hulpmiddel in diverse programmeertalen (bijvoorbeeld de functionele programmeertaal ML), onder meer als test op coherentie van programma's.
Ook natuurlijke taal vertoont een typestructuur. We kunnen dit demonstreren met het volgende eenvoudige technische apparaat:

DEFINITIE 12.5

Typen
Typen ontstaan via de volgende inductieve definitie:
a e (entity) en t (truth value) zijn typen.
b Als a en b typen zijn, dan ook $(a \rightarrow b)$ (functioneel van a naar b).

Voorbeeld 12.16

Typen
e
$(e \rightarrow t)$
$(t \rightarrow (t \rightarrow t))$
$((e \rightarrow t) \rightarrow (e \rightarrow t))$

Getypeerde λ–calculus

Met behulp van deze typen kunnen we opnieuw λ–termen definiëren. Om te beginnen krijgt in de getypeerde λ–calculus elke variabele x een eigen type a: notatie x^a. Getypeerde λ–termen worden inductief gedefinieerd (bij typen laten we de buitenste haakjes vaak weg):

DEFINITIE 12.6

Getypeerde termen

a Variabelen en constanten van type a zijn termen van type a.
b Als t_1 een term van type $a \rightarrow b$ is en t_2 een term van type a, dan is $(t_1\ t_2)$ een term van type b.
c Als t een term van type b is en x een variabele van type a, dan is $(\lambda x\ .\ t)$ een term van type $a \rightarrow b$.

De beperkingen die dit teweegbrengt, zien we aan een term als $(\lambda x\ .\ (x\ x))$. Omdat x op zichzelf wordt toegepast, is er namelijk geen typering voor x te geven die uit $(\lambda x\ .\ (x\ x))$ een acceptabele getypeerde term maakt.

Belangrijke theoretische eigenschappen van de ongetypeerde λ–calculus blijven opgaan en soms zelfs in versterkte vorm. In tegenstelling tot in de ongetypeerde λ–calculus geldt nu het volgende:

STELLING 12.3

Normaalvorm
Elke term in de getypeerde λ–calculus heeft een normaalvorm.

Semantiek

Een semantiek voor de ongetypeerde λ–calculus is nogal ingewikkeld, omdat er niet eenvoudig structuren te vinden zijn waarin functies en argumenten zich gelijksoortig gedragen. Een semantiek voor de getypeerde λ–calculus is simpeler te geven. Het basistype e verwijst naar een domein van individuele objecten, en t naar het domein der waarheidswaarden. Vervolgens worden hierover inductief alle verdere typendomeinen gebouwd via 'functieruimten':

$$D^{a \rightarrow b} = \{f \mid f\colon D^a \rightarrow D^b\}$$

Typen in natuurlijke taal

In de natuurlijke taal doen functionele typen zich nu voor als de semantische tegenhanger van traditionele grammaticale categorieën. Om te beginnen een eenvoudig voorbeeld:

Voorbeeld 12.17

Een eigennaam als 'Marie' is van type e: entiteit. De zin 'Marie loopt' is een ware of onware bewering, dus van type t. Het intransitief werkwoord 'loopt' is daarom van type $e \rightarrow t$: 'loopt' is een functie met als argument de 'entiteit' Marie en als functiewaarde 'Marie loopt': type t.

Dit leidt ons tot de volgende lijst. Deze lijst komt tot stand door met typen van zinsdelen te 'puzzelen' zoals in het voorgaande voorbeeld.

grammaticale categorie	*voorbeeld*	*functioneel type*
eigennaam	Marie	e
zin	Marie loopt	t
intransitief werkwoord	loopt	$e \rightarrow t$
transitief werkwoord	vindt	$e \rightarrow (e \rightarrow t)$
bijwoord	langzaam	$(e \rightarrow t) \rightarrow (e \rightarrow t)$
naamwoordelijke uitdrukking	elk kind	$(e \rightarrow t) \rightarrow t$
determinator	elk	$(e \rightarrow t) \rightarrow ((e \rightarrow t) \rightarrow t)$
voorzetselgroep	met elk kind	$(e \rightarrow t) \rightarrow (e \rightarrow t)$
voorzetsel	met	$((e \rightarrow t) \rightarrow t) \rightarrow ((e \rightarrow t) \rightarrow (e \rightarrow t))$

Getypeerde λ–termen in natuurlijke taal

De rol van λ–termen bij dit alles is als volgt. Met enkelvoudige woorden in een categorie corresponderen constanten van het bijbehorend type. De semantische betekenis van complexe uitdrukkingen kan dan worden weergegeven door een term van een passend type, die is ontstaan door parallel met de syntactische constructies geschikte operaties van functionele applicatie en abstractie uit te voeren (in hoofdstuk 19 vindt u hier voorbeelden van).
Dat laatste betekent overigens dat de λ–calculus behalve als handig formalisme ook als bewijssysteem voor natuurlijke taal gebruikt kan worden. Immers, het berekenen van de gelijkheid van λ–termen voor complexe uitdrukkingen komt neer op het bepalen of die uitdrukkingen *synoniem* zijn, dat wil zeggen dezelfde betekenis hebben.

Dit besluit onze schets van de λ–calculus als een speciaal maar ruim toepasbaar systeem van logica. In hoofdstuk 19 komen we terug op de formele verwerking van natuurlijke taal met behulp van getypeerde λ–calculus.
We gaan nu over tot het bespreken van nog twee andere richtingen van uitbreiding van de predikaatlogica.

12.8 UITBREIDINGEN: MODALE LOGICA

Een geheel ander soort uitbreiding dan de hogere–orde logica ontstaat door het onderkennen van een andersoortige beperking aan de

predikaatlogica. Wat tweede– en hogere–orde logica's onveranderd laten, is de kern van de eerdere predikaatlogische semantiek: we spreken in principe over dezelfde structuren $\boldsymbol{D} = \langle D, \boldsymbol{R}, \boldsymbol{O} \rangle$, al doen we er dan sterkere beweringen over. Of, in het geval van λ–calculus, we bespreken een vaste semantische hiërarchie over dit basisdomein.
Er zijn echter ook diverse vormen van redeneren waarbij dat statische semantische beeld niet langer voldoet. Met name geldt dit voor zogenaamde *intensionele* uitdrukkingen in natuurlijke taal zoals 'weten', 'kunnen' en 'moeten', en voor uitdrukkingen die een *tijdsverloop* aangeven zoals verleden en toekomstige werkwoords–'tijden'. Gemeenschappelijk aan de betekenis van zulke uitdrukkingen is de vergelijking tussen een aantal *verschillende* structuren: denkbare of gewenste alternatieve situaties, gebeurtenissen door de tijd heen, enzovoort. We willen een logische taal die naar zulke alternatieve situaties kan verwijzen. We krijgen zo'n taal door de predikaatlogica uit te breiden met zogenaamde 'modale operatoren'. Met modale operatoren kunnen beweringen worden gedaan over waarheid in alle of sommige alternatieve situaties. Die uitbreiding van de predikaatlogica heet *modale logica*. De predikaatlogische semantiek en geldigheid wordt dus in modale logica gegeneraliseerd. Een en ander vindt u terug in hoofdstuk 13.

12.9 UITBREIDINGEN: MEERWAARDIGE LOGICA

Nog een andere uitbreiding van de standaardlogica ontstaat als we het aantal voorkomende waarheidswaarden uitbreiden. Alle tot nu toe behandelde logica's waren tweewaardig (met waarden 0 en 1). We kunnen echter ook meerdere waarheidswaarden toestaan. Aldus ontstaat de zogenaamde *meerwaardige logica*.
Een voorbeeld hiervan is driewaardige logica waar naast de waarden 0 en 1 een derde waarde voorkomt, aangegeven met bijvoorbeeld # of 1/2 (als waarde tussen 0 en 1). De betekenis van # is afhankelijk van de toepassing: denk bijvoorbeeld aan 'onbekend', 'nog niet bekend' of 'ongedefinieerd'. Een voorbeeld uit de informatica: als het test-programma 'is φ waar?' termineert, dan is bepaald of φ waar of onwaar is. Als het niet termineert, dan is de waarde van φ onbekend.

Interpretatie van logische symbolen

In meerwaardige logica zijn er door de fijnere onderscheiding in waarden meer keuzes voor logische operaties. Dit komt tot uitdrukking in de verschillende mogelijkheden voor waarheidstabellen. We geven mogelijke driewaardige tabellen voor $\neg$, $\wedge$ en $\vee$:

$\neg$	
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	#	#

Deze tabellen sluiten aan bij de intuïtieve betekenis van de notie onbekend. Een voorbeeld: $\varphi \vee \psi$ is waar als φ waar is en ψ onbekend, want mocht ψ alsnog bekend worden, dan geldt $\varphi \vee \psi$ onafhankelijk van de waarheid van ψ.
Bijvoorbeeld, de tabel voor $\vee$ sluit niet aan bij de notie ongedefinieerd: $\varphi \vee \psi$ is ongedefinieerd als φ waar is en ψ ongedefinieerd. Dit is op grond van de compositionaliteit van betekenis: de betekenis van het geheel is niet gedefinieerd als de betekenis van een der delen niet gedefinieerd is.
Ook voor de andere connectieven zijn andere tabellen mogelijk.

Geldigheid

Behalve de interpretatie van logische symbolen is een ander probleem de geldigheid. In een meerwaardig systeem is bijvoorbeeld de waarde van $\varphi \vee \neg\varphi$ niet altijd 1, maar wel nooit 0.
We zullen hier niet verder op deze en andere consequenties ingaan.

Meer dan drie waarheidswaarden

Driewaardige logica is een specifiek geval van *n–waardige logica* waarbij n verschillende waarheidswaarden gebruikt worden. In het algemeen kunnen we hier denken aan een sprongsgewijs oplopende 'schaal van waarheid' (met waarden $0, 1/(n-1), 2/(n-1), \ldots, (n-2)/(n-1), 1$).

Oneindig veel waarheidswaarden: vage logica

Ten slotte is er het speciale geval dat er oneindig veel waarheidswaarden zijn. In dit geval kan elk getal uit het interval $[0, 1]$ als waarheidswaarde dienen. Er is dan een continu verloop van waarheidswaarden. De aldus ontstane logica heet *vage logica* of *fuzzy logica*. Fuzzy logica wordt onder andere toegepast voor het redeneren met onzekerheid in expertsystemen, voor patroonherkenning in geschreven of gesproken taal en ook in de taalwetenschap voor vage kwantoren zoals 'veel', 'sommige' en 'bijna geen'.

Overigens is het ontwerpen van uitbreidingen van de predikaatlogica een open proces, dat nog steeds voortgaat. Onze bespreking heeft echter wel de belangrijkste genres geïllustreerd.

12.10 ALTERNATIEVEN: INTUÏTIONISTISCHE LOGICA

Wellicht het meest intrigerend vanuit een meer filosofisch gezichtspunt is het voorkomen van logische systemen die niet op de predikaatlogica variëren of erop voortbouwen, maar die pretenderen er een beter

alternatief voor te bieden. Dat de klassieke predikaatlogica niet altijd bevredigend functioneert zien we aan het volgende voorbeeld:

Voorbeeld 12.18

Met behulp van het 'principe van uitgesloten derde' $\varphi \vee \neg\varphi$ (een propositielogische stelling) kunnen we weliswaar abstract bewijzen dat er niet–rationale reële getallen p en q zijn zodat p^q rationaal is, maar helaas hebben we daarmee nog geen concrete constructie gevonden.

Bewijs

Beschouw $\sqrt{2}^{\sqrt{2}}$. Dit getal is ofwel rationaal ofwel niet–rationaal ($\varphi \vee \neg\varphi$). Als het rationaal is, dan zijn we klaar: neem $p = q = \sqrt{2}$. Als het niet rationaal is, dan is $(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{(\sqrt{2} \cdot \sqrt{2})} = \sqrt{2}^2 = 2$ wel rationaal en zijn we ook klaar: neem $p = \sqrt{2}^{\sqrt{2}}$ en $q = \sqrt{2}$. □

We hebben wel het gevraagde bewezen, maar geen constructie gegeven van de gevraagde p en q! Dit drukt precies het bezwaar tegen de klassieke logica uit.

Intuïtionische logica

Het bekendste alternatief voor de predikaatlogica is de zogenaamde *intuïtionistische* logica. Deze logica baseert haar uitleg van de betekenis der logische constanten niet op het waarheidsbegrip, zoals in dit boek tot nu toe in hoofdzaak is gebeurd, maar op het begrip 'constructieve bewijsbaarheid'.
Bijvoorbeeld, een intuïtionist zal een existentiële bewering (zoals in het voorgaande voorbeeld) als volgt lezen: $\exists x\, P(x)$ drukt het vermogen uit om een *voorbeeld* te *construeren* van een object dat bewijsbaar de eigenschap P heeft. En evenzo schuilt de constructieve betekenis van een implicatie $\varphi \to \psi$ in het aangeven van een *methode* om bewijzen van φ om te zetten in bewijzen voor ψ.
Wat dan kan gebeuren is een uiteenlopen van oordelen over de geldigheid van eenzelfde, of althans eender geformuleerd, logisch principe.

Zo erkent de klassieke predikaatlogica, maar mist de intuïtionistische logica, het principe van dubbele negatie:

$$\neg\neg\varphi \leftrightarrow \varphi$$

Constructief is 'weerleggen van de ontkenning' van een bewering niet equivalent met aantonen van die bewering! Dit verschil in bewijskracht zagen we reeds in hoofdstuk 4, waar de intuïtionistische propositielogica werd gedefinieerd als klassieke natuurlijke deductie zonder de negatieregel ¬E* voor 'bewijs uit het ongerijmde'. In het bewijs van het

principe van dubbele negatie moesten we immers de $\neg$E*–regel gebruiken. Ook het principe van uitgesloten derde $\varphi \vee \neg\varphi$ is niet constructief bewijsbaar.

Evenzo erkent de klassieke logica de geldigheid van 'Plato's wet', die door de intuïtionistische logica ontkend wordt:

$\exists y\ (\exists x\ \varphi \rightarrow [y/x]\varphi)$

Ondanks de genoemde constructieve lezing van $\exists$, zijn de kwantor-regels van natuurlijke deductie overigens in beide logische systemen dezelfde.

Semantiek

Klassieke en intuïtionistische logica lopen dus uiteen in bewijskracht. Maar ook in vele andere opzichten verschillen ze. Zo past de meest gangbare *semantiek* voor intuïtionistische logica beter in het modale kader van hoofdstuk 14: ze is nauwer gelieerd aan het modale begrip 'kennis' dan aan waarheid op zich.

Existentie–eigenschap

Ook op metaniveau zijn de klassieke en de intuïtionistische logica verschillend. Zo weerspiegelt de constructieve lezing van de existentiële kwantor zich in de zogenaamde *existentie–eigenschap*.
Een formule $\exists x\ \varphi$ heeft klassiek een niet–constructieve betekenis. Concreet hebben we dit gezien in voorbeeld 12.18. Daarin hebben we klassiek bewezen dat er getallen p en q zijn waarvoor een bewering geldt, maar we hebben geen concrete getallen gevonden, met andere woorden: geen constructie gegeven. Verder merken we dit bij geldige principes als Plato's wet $\exists y\ (\exists x\ \varphi \rightarrow [y/x]\varphi)$, waarbij in het algemeen geen enkele term t expliciet valt aan te geven zodat $\exists x\ \varphi \rightarrow [t/x]\varphi$ klassiek geldig wordt. Intuïtionistisch daarentegen geldt de existentie–eigenschap:

Indien $\vdash \exists x\ \varphi$, dan is er een term t waarvoor $\vdash [t/x]\varphi$.

Met name de existentie–eigenschap maakt de intuïtionistische logica in ieder geval een nuttig systeem voor die wetenschapsgebieden waar constructief redeneren vereist is. Volgens sommigen valt de informatica daar grotendeels binnen! Immers, aantonen dat problemen oplosbaar zijn via de strengere normen van de constructieve logica, heeft kennelijk een bonus in de vorm van een expliciete term die algoritmische informatie verschaft over de oplossing.

Wanneer men intuïtionistisch bijvoorbeeld kan bewijzen dat er voor elke rij getallen een gesorteerde versie bestaat, dan kan uit dat bewijs een sorteeralgoritme worden afgelezen, en hoeven we de correctheid van dat algoritme niet ook nog eens apart aan te tonen zoals in de gebruikelijke programmeerpraktijk.

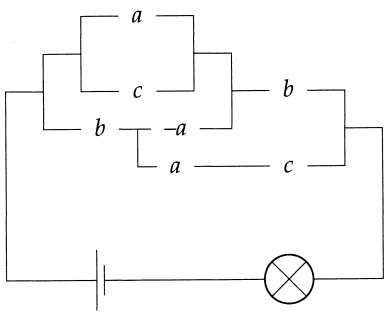
Overigens moet men zich bij het voorgaande niet voorstellen dat de logica in elkaar bestrijdende kampen uiteenvalt. Om te beginnen liggen de scheidslijnen tussen 'alternatieve logica's' soms wat vaag. Zo zijn er, om eens een ander voorbeeld te noemen, diverse systemen ontwikkeld voor de implicatie 'als ... dan ...' zoals die functioneert in redeneren in natuurlijke taal, die het gedrag hiervan getrouwer verantwoorden dan de waarheidstabellen van hoofdstuk 2. Of men dit nu als 'uitbreidingen' of 'alternatieven' voor de standaardpredikaatlogica moet beschouwen is grotendeels een kwestie van smaak. Bovendien is er nog een tweede vervagende factor: door middel van geschikt gekozen *vertalingen* is redeneren in een gegeven logisch systeem vaak te reduceren tot redeneren in een ander logisch systeem. Hiermee ontstaat op een dieper niveau toch weer eenheid in de logica. Hier gaan we niet verder op in.

12.11 TOT BESLUIT

Het geconstateerde veelvoud aan logische mogelijkheden voor redeneersystemen is vanuit de informatica geheel begrijpelijk en ter zake. Dat variaties in notatie en beperkingen tot kleine fragmenten uiterst nuttig kunnen zijn, leert de ontwikkeling van programmeertalen. En uitbreidingen, zoals hogere–orde logica of modale logica, zijn ook opgekomen in de studie van programmaverwerking. Zie voor dat laatste de bespreking van dynamische logica in de hoofdstukken 13 en 15. Ten slotte heeft ook het informaticaonderzoek zelf reeds alternatieven opgeworpen voor de standaardpredikaatlogica, zoals 'niet–monotone logica's', die binnen de kunstmatige intelligentie ontstonden voor de weergave van plausibel redeneren. Over deze omgekeerde invloed van informatica naar logica komen we in hoofdstuk 21 nog uitvoeriger te spreken.

12.12 OPGAVEN

12.1 Beschouw de volgende blauwdruk voor een netwerk van elektrische schakelingen:



a Geef een Boolese omschrijving van dit netwerk met behulp van de Boolese operaties +, · en –.
b Bewijs met behulp van de Boolese axioma's dat dit netwerk te vereenvoudigen is tot de schakeling met alleen schakelaar b.

12.2 Motiveer de genoemde axioma's voor procesalgebra (zie voorbeeld 12.5).

12.3 In natuurlijke taal komen kwantoren gerelativeerd voor. We zeggen bijvoorbeeld 'Alle vaders vinden hun eigen kind het mooist' en niet 'Voor alle mensen geldt dat als een mens een vader is, dan vindt hij zijn kind het mooist'.
Geef aan hoe we gerelativeerde kwantoren in meersoortige logica kunnen behandelen.

12.4 Geef normaalvormen voor elk van de volgende λ–termen:
i $((\lambda x\,.\,((\lambda y\,.\,(y\ x))\ (\lambda z\,.\,z)))\ a)$
ii $((\lambda z\,.\,(z\ (z\ x)))\ (\lambda y\,.\,(y\ a)))$

12.5 Een λ–term t is *typeerbaar* als het mogelijk is om elke variabele die in t voorkomt, van een type te voorzien, zodanig dat daardoor ook t een type krijgt. Bijvoorbeeld, de term $(\lambda x\ .\ (x\ y))$ is typeerbaar. Geef x type $e \to t$ en y type e, dan heeft $(x\ y)$ type t en $(\lambda x\ .\ (x\ y))$ type $(e \to t) \to t$. Maar $(\lambda x\ .\ (x\ x))$ is niet typeerbaar: een term $(t_1\ t_2)$ heeft per definitie alleen een type als t_1 type $a \to b$ heeft en t_2 type a. Dit is niet mogelijk als t_1 en t_2 dezelfde term (hier: x) zijn.
Kunt u een methode geven om uit te maken of een willekeurige gegeven term t typeerbaar is?

* 12.6 Beschouw de volgende analogie tussen getypeerde λ–termen en afleidingen:

- een term $(t^{a \to b}\ t^a)$ correspondeert met →E:

$$\frac{a \qquad a \to b}{b}$$

- een term $\lambda x^a\ .\ t^b$ correspondeert met →I:

$$\begin{array}{c} a \\ \vdots \\ b \\ \hline a \to b \end{array} [-a]$$

a Geef een afleiding van $(e \to ((e \to t) \to (e \to t)))$ uit $(((e \to t) \to t) \to (t \to t))$ (met behulp van →E en →I) en de bijbehorende λ–term.
b Geef een afleiding die correspondeert met de volgende λ–term: $(\lambda x^{((e \to t) \to t) \to t}\ .\ (\lambda y^e\ .\ (x^{((e \to t) \to t) \to t}\ (z^{e \to (e \to t))}\ y^e))))$.

* 12.7 Laat zien dat de zogenaamde 'Wet van Peirce' intuïtionistisch geen stelling is:

$$\not\vdash_I ((\varphi \to \psi) \to \varphi) \to \varphi$$

Modale logica

Hoofdstuk 13

Modale logica

13.1 INLEIDING

De propositielogica en de predikaatlogica stellen ons in staat om stukjes werkelijkheid te modelleren en daar zinvolle beweringen over te doen. Toch hebben deze logica's hun beperkingen. Zo zijn waarderingen en modellen in principe vrij statische begrippen. Ze zijn momentopnamen van de (wiskundige) wereld of van bijvoorbeeld een computerprogramma. Maar in werkelijkheid is de wereld, en zeker een programma, in beweging. Tijdens de verwerking van een programma zullen bijvoorbeeld regelmatig de waarden van variabelen veranderen en daarmee ook de waarden van eventuele functies. Ook in het dagelijkse leven komen uitdrukkingen voor die aan verandering onderhevig zijn. Beschouw bijvoorbeeld de uitdrukking 'de premier van Nederland'. De *betekenis* hiervan staat vast: de premier is de eerste minister, de leider van het kabinet dat Nederland bestuurt. Wat echter niet vaststaat is het individu waarnaar deze uitdrukking *verwijst*. In 1990 verwijst 'de premier van Nederland' naar het individu R. Lubbers, maar in 2001 naar W. Kok.
Het kan ook voorkomen dat we niet alleen de werkelijke wereld, maar ook mogelijke alternatieven willen modelleren. Terwijl een waardering of een predikaatlogisch model weergeeft hoe de situatie er op een bepaald moment uitziet, had die situatie er op dat moment ook anders uit kunnen zien. Hoewel in 2002 Engeland niet tot de eurolanden behoort, is het *mogelijk* dat Engeland eerder wel besloten had om mee te doen. Anders gezegd: een waardering of model geeft een stukje werkelijkheid weer, maar het is niet *noodzakelijk* dat de wereld er op dat moment zo uitziet. Een ander model (waarin er dertien eurolanden zijn) had ook werkelijk kunnen zijn. Dit aspect van mogelijkheid of noodzakelijkheid van beweringen heet *modaliteit*. Ook om dit te kunnen analyseren is een aanpassing van de standaardlogica nodig. De logica waarin we deze modaliteiten kunnen uitdrukken heet 'modale logica'. Deze logica werd oorspronkelijk ontwikkeld om modaliteiten te modelleren. Tegenwoordig wordt modale logica breed toegepast binnen

bijvoorbeeld informatica, kunstmatige intelligentie, taalkunde en wiskunde.
In dit hoofdstuk presenteren we de basiseigenschappen van modale logica, waarbij de nadruk ligt op modale propositielogica, met af en toe een verwijzing naar de toepassingen. Op een aantal van deze toepassingen gaan we in de hoofdstukken 15, 18 en 19 uitvoerig in.

13.2 TAAL VAN DE MODALE LOGICA

We beginnen op het eenvoudigste niveau. De taal van de *modale propositielogica* is een uitbreiding van de taal van de propositielogica met twee nieuwe logische symbolen: de *modale operatoren* $\Box$ ('noodzakelijk', 'blokje') en $\Diamond$ ('mogelijk', 'ruitje').

DEFINITIE 13.1

Alfabet
De taal van de modale propositielogica heeft een alfabet met
a een verzameling propositieletters;
b logische symbolen: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow, \Box, \Diamond$;
c hulpsymbolen:) en (.

Formules in de modale propositielogica ('modale formules') worden zoals steeds inductief opgebouwd:

DEFINITIE 13.2

Modale formules
a elke propositieletter is een formule;
b als φ en ψ formules zijn, dan zijn $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \rightarrow \psi)$, $(\varphi \leftrightarrow \psi)$, $\Box\varphi$ en $\Diamond\varphi$ ook formules;
c niets anders is een formule.

De manier waarop we de modale operatoren $\Box$ en $\Diamond$ lezen zal afhangen van het doel waarvoor we ons met modale logica bezighouden. De 'klassieke' leeswijze is 'noodzakelijk' voor $\Box$ en 'mogelijk' voor $\Diamond$. Maar wanneer we modale logica gebruiken om processen in de tijd te beschrijven kunnen we $\Box$ lezen als 'voortaan' en $\Diamond$ als 'eens (in de toekomst)', en in het geval dat we het gedrag van een computerprogramma π onderzoeken lezen we $\Box\varphi$ als 'na elke uitvoering van π geldt φ', en $\Diamond\varphi$ als 'er is een uitvoering van π waarna φ geldt'. In kennislogica wordt de operator $\Box$ gebruikt voor 'weten dat'. Als we een specifieke toepassing voor ogen hebben, gebruiken we vaak een speciale notatie, zoals G (going to be) voor $\Box$ in tijdslogica, en $[\pi]\varphi$ om aan te geven dat φ geldt na elke uitvoering van het programma π. In de hoofdstukken 15, 18 en 19 leest u meer over speciale toepassingen van modale logica.

Voorbeeld 13.1

Een aantal voorbeelden van formules, met voor de eerste drie formules een mogelijke leeswijze:

- $\Box p$ 'noodzakelijk p'
- $\Box (p \rightarrow q)$ 'na elke uitvoering van het programma geldt $p \rightarrow q$'
- $(\Diamond p \vee \neg p)$ 'eens p of nu $\neg p$'
- $\Diamond (p \rightarrow \Box \Diamond \Box p)$

Bereik van modale operatoren

Evenals bij de kwantoren is ook bij $\Box$ en $\Diamond$ het *bereik* van belang. In $\Box (p \rightarrow q)$ is het bereik van $\Box$ de formule $p \rightarrow q$; in $\Box p \rightarrow q$ is het bereik van $\Box$ slechts p. Dat de betekenis van beide formules verschilt zullen we in de volgende paragraaf zien, wanneer we de semantiek behandelen. In modale interpretaties van beweringen in natuurlijke taal is het bereik van modale operatoren vaak onduidelijk. De zin 'als A, dan moet B het geval zijn' (bijvoorbeeld: 'als het regent, dan moeten de pannen nat worden') kunnen we modaal weergeven als $A \rightarrow \Box B$ en als $\Box(A \rightarrow B)$.

13.3 SEMANTIEK

Mogelijke werelden

Om modale formules te interpreteren wordt de semantiek van de propositielogica uitgebreid. In de propositielogica legt een waardering een toestand vast van de 'wereld'. In de modale logica wordt ook rekening gehouden met andere 'mogelijke werelden' en daarmee met meerdere waarderingen. Denk bij het begrip 'mogelijke wereld' bijvoorbeeld aan verschillende toestanden in de tijd, of verschillende geheugentoestanden van een computer. Ons model bevat dus een verzameling werelden (toestanden), met op elk van die werelden een waardering. Onderling zijn de werelden van ons model verbonden door middel van een toegankelijkheidsrelatie. Klassiek wordt deze toegankelijkheidsrelatie geïnterpreteerd als 'vanuit wereld w is wereld w' voorstelbaar'. Wanneer we zeggen dat iets *mogelijk* is, bedoelen we dat we ons *een* situatie voor kunnen stellen waarin dat het geval is. Als iets daarentegen *noodzakelijk* zo is, is het in *iedere* voorstelbare situatie het geval. Dit leidt tot een *mogelijke–wereldensemantiek* die werkt met de volgende 'patronen':

DEFINITIE 13.3

Mogelijke–wereldenmodel
Een mogelijke–wereldenmodel (kort: 'model') $M = \langle W, R, V \rangle$ voor de modale propositielogica bestaat uit:

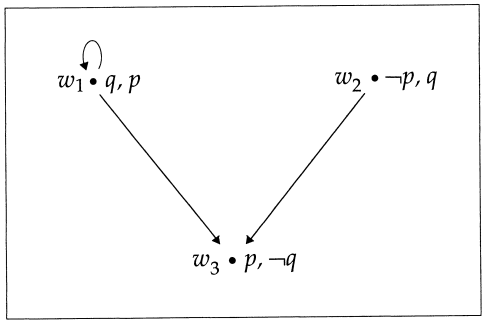
a een niet–lege verzameling W van mogelijke werelden;
b een binaire toegankelijkheidsrelatie R tussen mogelijke werelden in W;

c een waardering V die in elke mogelijke wereld w aan elke propositieletter p een waarde $V_w(p)$ geeft.

Een mogelijke–wereldenmodel is grafisch weer te geven. Een mogelijke wereld geven we aan met een punt en de toegankelijkheidsrelatie R tussen de werelden door middel van pijlen. Tevens geven we bij de werelden voor ieder atoom aan of dat atoom of de negatie van dat atoom daarin waar is.

Voorbeeld 13.2

Beschouw het volgende model $M = \langle W, R, V \rangle$ met:
$W = \{w_1, w_2, w_3\}$
$R = \{(w_1, w_1), (w_1, w_3), (w_2, w_3)\}$
$V_{w_1}(p) = V_{w_1}(q) = V_{w_2}(q) = V_{w_3}(p) = 1$
$V_{w_2}(p) = V_{w_3}(q) = 0$
In een plaatje:



Waarheidsdefinitie

Met behulp van de nu volgende waarheidsdefinitie kunnen we voor een modale formule φ in elke wereld w van een model M bepalen of deze formule geldig is in w. Is dit het geval dan schrijven we

$M, w \models \varphi$

Of φ waar is in w hangt af van de waarden van subformules van φ in verschillende mogelijke werelden van M. De precieze waarheids-definitie voor de modale propositielogica verloopt inductief als volgt:

DEFINITIE 13.4

Waarheidsdefinitie

a $M, w \models p \quad \Leftrightarrow \quad V_w(p) = 1$ voor alle propositieletters p
b $M, w \models \neg\varphi \quad \Leftrightarrow \quad M, w \not\models \varphi$
c $M, w \models \varphi \wedge \psi \quad \Leftrightarrow \quad M, w \models \varphi$ en $M, w \models \psi$
de overige connectieven gaan evenzo