Deze formule zegt: 'als ik weet dat φ het geval is, dan is φ het geval'. Of met andere woorden: alles wat ik weet is waar, vandaar de naam waaronder dit axioma wel bekendstaat. Voor kennislogica is dit axioma plausibel. Het drukt immers een kenmerkende eigenschap van kennis uit, waarin dit begrip verschilt van andere geesteshoudingen zoals 'geloven'. Als je *gelooft* dat iets waar is, is op zich (dat wil zeggen: voor een waarnemer) nog voorstelbaar dat het toch niet zo is, maar als je het daarentegen *weet*, is dat onvoorstelbaar. Het axioma $K\varphi \rightarrow \varphi$ correspondeert met de frame–eigenschap van reflexiviteit, zoals we al in hoofdstuk 13 gezien hebben.

Positieve introspectie

$$K\varphi \rightarrow KK\varphi$$

Dit axioma wordt in de context van kennislogica *positieve introspectie* genoemd en zegt: 'als ik weet dat φ, dan weet ik dat ik dat weet'. Of met andere woorden: ik ben me bewust van al mijn kennis. In hoofdstuk 13 hebben we reeds gezien dat het axioma $K\varphi \rightarrow KK\varphi$ correspondeert met de frame–eigenschap van transitiviteit.

Negatieve introspectie

$$\neg K\varphi \rightarrow K\neg K\varphi$$

Veel logici en informatici onderschrijven ook deze tegenhanger van het principe van positieve introspectie. *Negatieve introspectie* zegt dat 'als ik niet weet dat φ, dan weet ik dat ik dat niet weet'. Of met andere woorden: ik ben me bewust van al mijn onkunde. Dit principe is alleen houdbaar als je een volledig overzicht hebt over alle feiten die relevant zijn voor de beschrijving van de wereld om je heen. In de informatica is deze aanname heel vanzelfsprekend. Dit zegt namelijk dat de structuur van het model – bijvoorbeeld een distributief systeem – bekend is, ook al weet je niet de concrete waarden van de relevante variabelen. Maar in de realiteit van alledag is die aanname niet vanzelfsprekend. Bijvoorbeeld: u bent zich er (waarschijnlijk) niet van bewust dat u niet weet dat de Moa een in Nieuw-Zeeland uitgestorven loopvogel is. Op de toegankelijkheidsrelatie in frames correspondeert *negatieve introspectie* met de eis van 'euclidiciteit':

Euclidiciteit

$$\forall x \forall y\, (Rxy \rightarrow \forall z\, (Rxz \rightarrow Rzy))$$

De laatste drie kennisaxioma's maken die epistemische toegankelijkheid tot een *equivalentierelatie*. Dit sluit natuurlijk aan bij het intuïtieve idee dat de toegankelijkheidsrelatie kennistoestanden verbindt die niet van

elkaar te onderscheiden zijn. Deze axioma's beschrijven dus hoe *ideale agents* ('volmaakte logici'), die niets vergeten, die zich bewust zijn van alle afleidbare gevolgen van hun kennis, en onfeilbaar zijn, kennis verwerken.

Voorbeeld 19.3

In de uitgestrekte wouden van Nieuw-Zeeland komen glimwormen voor. Deze leven in mossige overhangende rotswandjes en je kunt ze natuurlijk alleen in het donker waarnemen. Als je daar bij nacht rondwandelt, met een zaklantaarn, en langs een plek loopt waar ze misschien zitten, moet je natuurlijk wel eerst die lamp uitdoen om dat te kunnen vaststellen. Daarna word je dan soms verrast door een prachtig lampjesbehang op het mos, en soms niet.
Stel p staat voor 'je lantaarn is aan' en q staat voor 'die plek heeft glimwormen', dan krijg je dus alleen uitsluitsel over q als $\neg p$. Stel dat de werkelijke wereld die is, waarbij zowel p als q waar zijn. In die gegeven wereld kun je je voorstellen dat p waar is en q niet waar. Maar je kunt je eveneens die werkelijke wereld zelf voorstellen: p en q beide waar. Je kunt je niet voorstellen dat p niet waar is: je ziet immers dat je lantaarn aan is! Als de situatie p en $\neg q$ was geweest, had ik me vanuit *die* wereld ook die wereld zelf, en de wereld voor p en q, kunnen voorstellen. Verder is het zo, dat als de lantaarn uit was geweest, ik direct had kunnen waarnemen of er glimwormpjes zijn of niet, dus $\neg p$ en q is voorstelbaar als dat zo is, en $\neg p$ en $\neg q$ is voorstelbaar als *dat* zo is. In die zin heb ik kennis van het *hele* model voor deze situatie.
Het linker mogelijke–wereldenmodel hierna geeft een overzicht van al deze mogelijkheden. Omdat de toegankelijkheidrelatie een equivalentierelatie is, kunnen we ook met de eenvoudiger visualisatie rechts volstaan, waarin we alleen hebben aangegeven welke werelden niet van elkaar te onderscheiden zijn, of met andere woorden: dat de drie equivalentieklassen op dit domein zijn: $\{w_1, w_4\}$, $\{w_2\}$ en $\{w_3\}$.

Aan allerlei interessante kenmerken van deze logica gaan we voorbij. Zo is het bijvoorbeeld opmerkelijk dat epistemische beweringen in deze

logica altijd equivalent zijn met een eenvoudiger vorm waarin geen stapelingen van de modale operator voorkomen. Om een voorbeeld te noemen: de bewering $K\neg K\neg KKK\neg K\varphi$ is equivalent met de bewering $\neg K\varphi$, wat eenvoudig met gebruikmaking van voorgaande axioma's af te leiden is.

19.4 KENNISLOGICA VOOR MEER DAN EEN AGENT

We behandelen nu de kennislogica voor meer agents. Als we de kennisoperator K indiceren, waarbij $K_i\varphi$ betekent: 'persoon/processor i weet dat φ', kunnen we spreken over verschillende agents, en hun eventuele interactie. Met iedere kennisoperator K_i correspondeert dan een toegankelijkheidsrelatie R_i. Interactie van agents is vooral van belang in informaticatoepassingen, waar men kennislogica bijvoorbeeld gebruikt om te redeneren over het effect van protocollen tussen groepen processoren. Met 'stapeling' van verschillende operatoren K_i kunnen we uitdrukken dat processoren kennis over elkaar hebben. Bijvoorbeeld, $K_a\neg K_b p$ wil zeggen dat processor a weet dat processor b propositie p (zoals 'de lokale variabele x heeft waarde 3') niet weet, terwijl $K_a(K_b p \vee K_b\neg p)$ tot uitdrukking brengt dat a weet of b p weet.

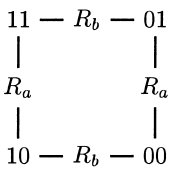
In tegenstelling tot het één–agentgeval, zijn er eigenlijk geen axioma's die in het algemeen de relatie tussen verschillende agents vastleggen, behalve dan principes die direct volgen uit axioma's voor één agent, zoals $K_aK_b\varphi \rightarrow K_b\varphi$. Zo geldt bijvoorbeeld zeker niet in het algemeen dat $K_aK_b\varphi \leftrightarrow K_bK_a\varphi$. Alleen voor beperktere klassen van modellen gelden zekere multi–agentvormen van axioma's, zoals we nu zullen zien. En concreter is het beschrijven van specifieke interactie tussen agents in een gegeven informatiesysteem natuurlijk waar het in dit hoofdstuk allemaal om draait.

Voor processoren is een redelijke aanname, dat ze *ten minste* hun eigen systeemtoestand kennen. Dit valt eenvoudig in modellen voor distributieve systemen af te dwingen: voor elke processor a bestaat de toegankelijkheidsrelatie R_a uit paren van globale systeemtoestanden met dezelfde a–coördinaat. Globale toestanden met een andere lokale a–coördinaat, 'de toestand van a', zijn onbereikbaar voor a. Deze distributieve of geïnterpreteerde systemen, die we ook al in paragraaf 19.2 hebben geïntroduceerd, zijn dus vanuit logisch standpunt niet meer (maar ook niet minder) dan bijzondere gevallen van mogelijke-wereldenmodellen waarbij alle toegankelijkheidsrelaties equivalentierelaties zijn.

Hoe het model er precies uitziet, is afhankelijk van wat processoren *van elkaar* weten. Als processor a toegang heeft tot *alle* informatie van processor b, komt dit overeen met het axioma $K_b\varphi \to K_aK_b\varphi$. Maar heel gebruikelijk is natuurlijk dat de processoren 'helemaal niets' van elkaar weten. Als alle processoren *alleen* hun eigen systeemtoestand kennen, zijn voor iedere processor a *alle* globale toestanden bereikbaar die overeenstemmen in de a–coördinaat. Dit is weer een vrij sterke aanname in het model, die het onderliggende frame karakteriseert. Voor het geval van twee agents a en b correspondeert deze eigenschap met het axioma $\neg K_a\neg K_b\varphi \to K_b\neg K_a\neg\varphi$: als a zich kan voorstellen dat b weet dat φ, dan weet b dat a zich φ kan voorstellen.

Voorbeeld 19.4

Distributieve systemen
Er zijn twee processoren a en b, met elk een lokale Boolese waarde. Atoom p beschrijft dat de waarde van processor a 1 is, en $\neg p$ dat deze waarde 0 is. Atoom q beschrijft de toestand van processor b. Schrijf 10 voor de globale toestand waarin a waarde 1 heeft en b waarde 0, enzovoorts. Beide processoren kennen alleen hun eigen toestand. Dit komt overeen met het volgende model *Vierkant*. Hierin hebben we de werelden namen gegeven die al 'verraden' welke atomen daar gelden, zodat we die extra informatie daarom hebben weggelaten.

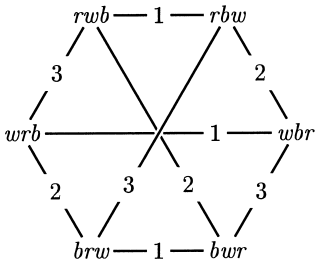


In wereld 11 van het model *Vierkant* geldt bijvoorbeeld dat $K_a p$ (processor a weet dat zijn toestand 1 is), dat $K_a\neg K_b p$ (a weet dat b dat niet weet), en dat $\neg K_a\neg K_b q$ (a houdt voor mogelijk dat b weet dat zijn toestand 1 is). Desgewenst kunt u ook nagaan dat het model *Vierkant* het schema $\neg K_a\neg K_b\varphi \to K_b\neg K_a\neg\varphi$ waar maakt.

Voorbeeld 19.5

Kaartverdelingen
Drie spelers 1, 2 en 3 trekken elk een kaart uit een pak van drie kaarten rood, wit en blauw. Met *rwb* geven we de kaartverdeling weer waarbij speler 1 rood heeft, speler 2 wit, en speler 3 blauw, enzovoorts. Er zijn zes mogelijke kaartverdelingen. Spelers kunnen alleen hun eigen kaart inzien, met andere woorden: ze hebben kennis van hun lokale toestand. Anders dan in het vorige voorbeeld is nu meer van de andere agents bekend. Van de anderen zien ze namelijk dat die ook maar één kaart

hebben, en ze weten dat dat niet hun eigen kaart kan zijn. De kennis van de spelers is weer te geven in het volgende model *Hexa*. Gemakshalve labelen we de verbinding tussen *rwb* en *rbw* met 1 in plaats van R_1, enzovoorts.



Stel atomaire propositie r_1 staat voor 'speler 1 heeft de rode kaart', enzovoorts. In wereld *rwb* van het model *Hexa* geldt bijvoorbeeld dat $K_1 r_1$ (speler 1 kent zijn eigen kaart) en dat $K_1(w_2 \vee b_2)$ (speler 1 weet dat 2 de witte of de blauwe kaart heeft). Overal in het model geldt dat $r_1 \rightarrow \neg K_3 K_1(w_2 \vee b_2)$ (als 1 de rode kaart heeft, dan weet 3 niet dat 1 weet dat 2 de witte of de blauwe kaart heeft).

19.5 ALGEMENE EN GEMEENSCHAPPELIJKE KENNIS

We kunnen ook kennisoperatoren definiëren voor *groepen* processoren. Bijvoorbeeld, in redeneren over de uitvoering van een protocol kan het nuttig zijn dat het systeem bepaalde handelingen pas verricht zodra de hele groep processoren iets te weten is gekomen. Ook kunnen we alleen met zulke operatoren de kennis formaliseren die agents delen over de context (de achtergrondinformatie). Twee varianten in kennis voor een groep processoren zijn algemene kennis en gemeenschappelijke kennis.

DEFINITIE 19.1

Algemene kennis
Als alle leden van een groep G weten dat φ, dan zeggen we dat die groep *algemene kennis* heeft van φ. Dit geven we weer als $E_G\varphi$, en dat staat dus voor de conjunctie van alle $K_a\varphi$ voor a in G.

Een sterker begrip is gemeenschappelijke kennis: elke herhaling van de algemene kennis (iteratie van de E_G–operator) is ook door de hele groep bekend. Dit komt op hetzelfde neer als willekeurige stapeling van individuele kennisoperatoren:

DEFINITIE 19.2

Gemeenschappelijke kennis
Als elke bewering $K_{a_1}...K_{a_n}\varphi$ opgaat voor elke eindige rij $a_1, ..., a_n$ uit een groep G, dan zeggen we dat G *gemeenschappelijke kennis* heeft van φ. Dit geven we weer als $C_G\varphi$.

Het verschil tussen algemene en gemeenschappelijke kennis lichten we toe met een voorbeeld:

Voorbeeld 19.6

Stippen
Gegeven een situatie waarin twee personen a en b tegenover elkaar staan en beide een stip op het voorhoofd hebben. Ze kunnen die dus wel bij de ander maar niet bij zichzelf zien. De kennis dat minstens een lid van de groep een stip op het voorhoofd heeft, $s_a \vee s_b$, is in deze groep van twee personen algemene kennis maar geen gemeenschappelijke kennis. Dit kunnen we als volgt formaliseren:
Persoon a ziet dat b een stip op het voorhoofd heeft, dat wil zeggen a weet s_b dus a weet $s_a \vee s_b$, met andere woorden: $K_a(s_a \vee s_b)$. Evenzo ziet b de stip op het voorhoofd van a, dus geldt $K_b(s_a \vee s_b)$. Dus $K_a(s_a \vee s_b) \wedge K_b(s_a \vee s_b)$, dat wil zeggen $E_{\{a,b\}}(s_a \vee s_b)$: $s_a \vee s_b$ is algemene kennis.
Persoon a kan haar eigen voorhoofd niet zien, weet dus niet dat zij zelf een stip op het voorhoofd heeft, en weet ook niet dat b dat bij haar kan zien. Dus $K_aK_b(s_a \vee s_b)$ geldt niet. Dus $s_a \vee s_b$ is geen gemeenschappelijke kennis.
Dat algemene en gemeenschappelijke kennis ook echt verschillen in deductieve kracht, zien we aan het volgende:
Stel dat een buitenstaander meedeelt "Minstens een van jullie heeft een stip op het voorhoofd." Hij maakt hiermee $s_a \vee s_b$ tot gemeenschappelijke kennis: $C_{\{a,b\}}(s_a \vee s_b)$. Persoon b zegt daarna "Ik weet niet of mijn voorhoofd bestipt is." Dan kan a nu als volgt rederenen: "Stel dat ik geen stip op mijn voorhoofd heb. Dan weet b dat ik geen stip op mijn voorhoofd heb (*i*). Omdat het gemeenschappelijk bekend is dat een van ons een stip op het voorhoofd heeft, weet b dat een van ons een stip op het voorhoofd heeft (*ii*). Uit (*i*) en (*ii*) volgt dat b weet dat hij zelf een stip op het voorhoofd heeft. Dat is in strijd met wat hij zojuist gezegd heeft. Dus ik heb wel een stip op mijn voorhoofd." Nog iets formeler: "Stel $\neg s_a$. Uit $\neg s_a$ volgt $K_b\neg s_a$. Uit $C_{\{a,b\}}(s_a \vee s_b)$ volgt $K_b(s_a \vee s_b)$ (*iii*). Uit $K_b(s_a \vee s_b)$ en $K_b\neg s_a$ volgt K_bs_b. K_bs_b is in tegenspraak met $\neg(K_bs_b \vee K_b\neg s_b)$. Dus s_a."
Dus, omdat a deze redenering doet: K_as_a. Bij (*iii*) gebruiken we feitelijk dat $K_aK_b(s_a \vee s_b)$. Daarmee hebben we dus inderdaad *meer* dan algemene kennis van $s_a \vee s_b$ nodig gehad. Een soortgelijk verhaal gaat natuurlijk ook op voor b.

Gemeenschappelijke kennis is dus echt krachtiger dan algemene kennis.

De operator C kan beschouwd worden als een soort 'iteratieve afsluiting' van alle K–operatoren, ongeveer zoals de modaliteiten $[\pi^*]$ in de dynamische logica van hoofdstuk 15. Dit kunnen we als volgt inzien. Met gebruikmaking van de equivalentie $[\pi\,;\,\pi']\varphi = [\pi][\pi']\varphi$ betekent $[\pi^*]\varphi$ zoveel als: $[\pi]...[\pi]\varphi$ gaat op voor elk eindig aantal achtereenvolgende uitvoeringen van π. Dit komt al aardig in de richting van de definitie van gemeenschappelijke kennis $C_G\varphi$ als: $K_{a_1}...K_{a_n}\varphi$ gaat op voor elke eindige rij $a_1, ..., a_n$ uit de groep G. De semantiek van deze modale operator is dan ook als volgt:

DEFINITIE 19.3

Semantiek van de gemeenschappelijke–kennisoperator
Voor de overzichtelijkheid zetten we de semantiek van een individuele en van een gemeenschappelijke kennisoperator naast elkaar. In de definitie gebruiken we nogmaals de (reflexieve en) transitivieve afsluiting R^* van een binaire relatie. Dit is de vereniging van willekeurig herhaalde composities van R met zichzelf: $R^* = \bigcup_n R^n = I \cup R \cup (R \circ R) \cup (R \circ R \circ R) \cup ...$ (waarbij I de identiteit is, de verzameling paren (x, x) voor alle elementen x in het domein van de relatie). Met een modale kennisoperator C_G associëren we nu een relatie $R_G := (\bigcup_{a\in G} R_a)^*$. We krijgen nu:

- $M, w \models K_a\varphi \Leftrightarrow$ voor alle $w' \in M$ geldt: als $R_a ww'$, dan $M, w' \models \varphi$
- $M, w \models C_G\varphi \Leftrightarrow$ voor alle $w' \in M$ geldt: als $R_G ww'$, dan $M, w' \models \varphi$

Voorbeeld 19.7

In voorbeeld 19.5 over kaartverdelingen geldt bijvoorbeeld, dat het bij gegeven kaartverdeling *rwb* gemeenschappelijk bekend is dat speler 2 een van de drie kaarten vasthoudt: $Hexa, rwb \models C_{\{1,2,3\}}(r_2 \vee w_2 \vee b_2)$. Deze bewering geldt, omdat we vanuit *rwb* een *pad* van verbindingen naar iedere andere kaartverdeling kunnen maken, en omdat overal in het model een van r_2, w_2, of b_2 geldt.

Inductieaxioma

Voor deze operatoren E en C in de kennislogica zijn volledige bewijssystemen bepaald. Een belangrijke rol hierin speelt het zogenaamde inductieaxioma $C(\varphi \rightarrow E\varphi) \rightarrow (\varphi \rightarrow C\varphi)$. Dit zegt, dat als φ het geval is, en als gemeenschappelijk bekend is ('altijd geldt') dat uit φ algemene kennis van φ volgt, dan mogen we concluderen dat φ gemeenschappelijk bekend is. Voor details, zie opgave 19.4.

De noties van algemene en gemeenschappelijke kennis spelen een belangrijke rol in het redeneren over de effecten van communicatie in een groep van parallel werkende processen. In feite kan het doel van

veel onderlinge communicatie beschreven worden als een poging de 'kwaliteit' van de groepskennis te verbeteren tot gemeenschappelijke kennis.

Impliciete kennis

Nog een andere kennisoperator voor groepen actoren is de zogenaamde *impliciete kennis* of gedistribueerde kennis. Dat de leden van een groep G impliciete kennis hebben van φ, betekent als het ware dat de kennis van φ over de groep verspreid is, en dat de agents deze kennis desgewenst door communicatie naar boven kunnen halen. Bijvoorbeeld, in het model *Hexa* van voorbeeld 19.5 is impliciet bekend wat de kaartverdeling is (ook al weet geen van spelers wat de kaartverdeling is), en in voorbeeld 19.4 is impliciet bekend wat de globale toestand is (ook al weet geen van beide processoren wat deze is). Op impliciete kennis gaan we verder niet in.

19.6 UPDATES IN MULTI–AGENTSYSTEMEN

Als voorbeeld van de reeds in hoofdstuk 16 genoemde mogelijkheid om verschillende soorten modale operatoren te combineren in één taal, breiden we de huidige epistemische logica uit met een ander type modale operator, namelijk een dynamische modale operator $[\varphi]$, die het effect beschrijft van het publiekelijk uitspreken van φ door een van de agents. We nemen aan dat φ waar is, en dat dat gemeenschappelijk bekend is. Zo'n operatie heeft veel overeenkomsten met de 'test op φ' uit paragraaf 15.5 over dynamische logica, waarmee een dynamische operator $[\varphi?]$ correspondeert. Daar zagen we, dat bijvoorbeeld de test $x = SS0?$ slaagt als de bedeling van x in de huidige toestand 2 is, en het resultaat van die test is dat die bedeling onveranderd blijft. Een update functioneert anders:

Voorbeeld 19.8

In model *Hexa* in wereld *rwb* zegt speler 1: "Ik heb rood." Dit komt overeen met een update op r_1. Dit is een 'test' voorzover de geldigheid van r_1 in de wereld *rwb* een voorwaarde is voor het kunnen doen van de uitspraak. Maar tevens worden nu alle werelden onvoorstelbaar waarin 1 rood niet heeft: we verwijderen alle werelden uit ons model waar r_1 niet geldt. Zulke updates mogen ook op epistemische formules, zoals wanneer speler 1 zegt: "Ik weet dat speler 2 mijn kaart niet kent."

Nu lijkt oppervlakkig gezien 'publiekelijk zeggen dat φ geldt' nogal veel op 'gemeenschappelijk bekendmaken dat φ'. Waarom volstaat het niet om de theorie te reviseren die het gegeven probleem beschrijft, en 'simpelweg' $C\varphi$ aan de beschrijving toe te voegen en beweringen die

niet meer geldig zijn te verwijderen? Dit kan niet vanwege de eigenaardigheid dat een bewering niet noodzakelijk meer geldt nadat je deze publiek bekend hebt gemaakt. Een voorbeeld:

Voorbeeld 19.9

In model *Hexa* in wereld *rwb* zegt speler 1: "Speler 2 weet niet dat ik rood heb." Dit komt overeen met een update op $r_1 \wedge \neg K_2 r_1$. Hierna geldt natuurlijk dat speler 2 *wel* weet dat 1 rood heeft. Na uitvoer van de update geldt dus het tegendeel!

We voegen deze updates nu aan onze epistemische taal toe:

DEFINITIE 19.4

Publieke update
Dat na publiek uitspreken van φ geldt dat ψ, geven we weer als $[\varphi]\psi$. We zeggen ook: na update op φ geldt ψ.

DEFINITIE 19.5

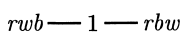
Semantiek van de publieke update
- $M, w \vDash [\varphi]\psi \Leftrightarrow$ als $M, w \vDash \varphi$, dan $M_\varphi, w \vDash \psi$.

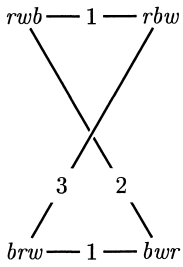
Hierbij is M_φ de beperking van model M, inclusief de toegankelijkheidsrelaties, tot alle werelden waar φ waar is. Met andere woorden: het domein van M_φ is de verzameling $\{w' \in W \mid M, w' \vDash \varphi\}$, en als twee werelden voor een agent a hetzelfde (ononderscheidbaar) waren in M, en die werelden komen eveneens in M_φ voor, dan zijn ze nog steeds voor a hetzelfde.

Een update op φ verandert dus een gegeven informatietoestand M, w in een nieuwe informatietoestand M_φ, w. In het Engels: een update is een 'state transformer'.

Voorbeeld 19.10

Teruggrijpend op de voorbeelden 19.5 en 19.9 hiervoor, kunnen we dus iets formeler opschrijven dat *Hexa*, *rwb* $\vDash [r_1]K_2 r_1$ en dat *Hexa*, *rwb* $\vDash [r_1 \wedge \neg K_2 r_1]K_2 r_1$. Bijvoorbeeld, *Hexa*, *rwb* $\vDash [r_1]K_2 r_1$ geldt, omdat $Hexa_{r_1}$, *rwb* $\vDash K_2 r_1$ (omdat $Hexa_{r_1}$, *rwb* $\vDash r_1$). Het model $Hexa_{r_1}$ bestaat uit de twee werelden waar 1 rood heeft (en dit zijn *dezelfde* twee werelden als die waarin 1 rood heeft en 2 dat niet weet), zie de linkerfiguur hierna. Op vergelijkbare wijze kunnen we berekenen dat *Hexa*, *rwb* $\vDash [\neg w_1]\neg K_2 r_1$. Zie daarvoor de rechterfiguur.





Voorbeeld 19.11

De uitspraken in het stippenvoorbeeld 19.6 zijn natuurlijk eveneens publieke updates. Als beide agents bestipt zijn, geldt dat $[s_a \vee s_b]C_{\{a,b\}}(s_a \vee s_b)$ en ook dat $[s_a \vee s_b][\neg(K_b s_b \vee K_b \neg s_b)]K_a s_a$.

Complexere vormen van dit soort dynamische updates zijn ook mogelijk, en het onderzoek naar dit soort dynamiek van kennis staat momenteel middenin de wetenschappelijke belangstelling. Zie verder opgave 19.5c.

19.7 OPGAVEN

19.1 Formuleer de volgende beweringen in termen van de kennisoperatoren K, E en C:
- Als a weet dat b φ weet, dan weet b dat a weet dat φ.
- a weet dat iedereen die weet dat φ, ook weet dat b weet dat γ.
- Als er iemand is die niet weet dat φ, dan weet a dat φ geen gemeenschappelijke kennis is.
- Als het geen gemeenschappelijke kennis is dat φ, dan is het gemeenschappelijke kennis dat dit geen gemeenschappelijke kennis is.

19.2 Laat met behulp van een tegenvoorbeeld zien dat $K_a K_b \neq K_b K_a$.

19.3 Op een vrolijk avondje bij u thuis kan iedereen weten dat sinterklaas niet bestaat ($E\neg\varphi$), terwijl uw sinterklaasavond er toch heel anders zou uitzien als ook nog $C\neg\varphi$ zou gelden. Leg dit uit.

* 19.4 Beschouw de volgende axiomatisering van gemeenschappelijke kennis in termen van algemene kennis:
1 $C(\varphi \to \gamma) \to (C\varphi \to C\gamma)$
2 $C\varphi \to \varphi$
3 $C\varphi \to EC\varphi$
4 $C(\varphi \to E\varphi) \to (\varphi \to C\varphi)$

Laat zien dat deze axioma's correct zijn met betrekking tot de modellen voor de kennislogica. (Hint: welke eerdere axioma's uit het systeem *DX* herkent u?)

19.5 a Bereken in *Hexa* dat $K_1(K_2r_2 \vee K_2w_2 \vee K_2b_2)$ overal geldt (speler 1 weet dat speler 2 zijn eigen kaart kent).
b Als bij de kaartverdeling *rwb* speler 1 zegt dat hij wit niet heeft, dan weet daarna 2 nog steeds niet dat 1 rood heeft, maar kunnen zowel 1 als 3 zich voorstellen dat 2 dat wel weet. Laat zien dat inderdaad:
- *Hexa*, $rwb \models [\neg w_1]\neg K_2 r_1$
- *Hexa*, $rwb \models [\neg w_1]\neg K_1\neg K_2 r_1$
- *Hexa*, $rwb \models [\neg w_1]\neg K_3\neg K_2 r_1$

*c Speler 1 laat aan speler 2 zijn kaart zien, terwijl 3 'toekijkt'. Dit is een voorbeeld van een update die niet publiek is. Welke informatietoestand resulteert als gevolg van deze actie?

19.6 Deze opgave is een variant op voorbeeld 19.6. Er zijn nu drie agents *a*, *b* en *c*, waarvan *a* en *b* bestipt zijn. Eerst zegt de buitenstaander weer: "Ten minste een van jullie is bestipt." Dan zegt *a* dat hij niet weet of hij bestipt is. Stel dat *b* daarna eveneens zegt dat hij niet weet of hij bestipt is. Bewijs dat *a* nu weet dat hij bestipt is.
Een lichte variant van dit soort stippenvoorbeelden staat in de literatuur ook bekend als het 'modderige kinderen'–probleem. In plaats van een stip hebben de agents, in dit geval kinderen, nu modder op hun voorhoofd, en dat kunnen ze wederom wel bij anderen maar niet bij zichzelf zien.

* 19.7 Gegeven zijn twee gehele getallen x en y met $1 < x < y$ en $x + y \leq 100$. *Som* krijgt de som van die getallen te horen en *Product* het product. Al het voorgaande is gemeenschappelijk bekend bij *Som* en *Product*. Nu volgt dit gesprek:
- *Product* zegt: "Ik weet de getallen niet."
- *Som* zegt: "Dat wist ik al."
- *Product* zegt: "Nu weet ik ze wel."
- *Som* zegt: "Nu weet ik ze ook."

Bepaal de getallen x en y.

* 19.8 Uit zeven kaarten 0, 1, 2, 3, 4, 5, 6 trekken Anna en Bert er ieder drie, en Cees krijgt de resterende kaart. Al het voorgaande is gemeenschappelijke kennis. Kunnen Anna en Bert elkaar openlijk van hun kaarten op de hoogte brengen, zonder dat Cees van een van hun kaarten leert wie

die kaart heeft? Neem voor het gemak aan dat Anna de kaarten 0, 1 en 2 heeft, Bert de kaarten 3, 4 en 5, en Cees dus kaart 6.

a Als een buitenstaander zegt: "Als Bert 0 niet heeft, dan heeft Anna 0, 1 en 2, en als Anna 3 niet heeft, dan heeft Bert 3, 4 en 5," dan geldt daarna dat Anna en Bert elkaars kaarten hebben geleerd, maar Cees geen van die van hen. Leg dit uit.

b Als daarentegen Anna tegen Bert zegt: "Als jij 0 niet hebt, dan heb ik 0, 1 en 2," en Bert tegen Anna zegt: "Als jij 3 niet hebt, dan heb ik 3, 4 en 5," dan weet Cees wél wat de kaartverdeling is. Waarom?

c Het is evenmin een oplossing, als Anna zegt: "Ik heb 0, 1 en 2 allemaal wel, of ik heb ze allemaal niet," en Bert tegen Anna zegt: "Ik heb 3, 4 en 5 allemaal wel, of ik heb ze allemaal niet," want ook dan leert Cees de kaartverdeling. Leg uit waarom.

c Niettemin, het antwoord op de gestelde vraag is: "Ja, dit kan." Er zijn meerdere oplossingen. Geef een oplossing voor het probleem.

Verwerking van natuurlijke taal

Hoofdstuk 20

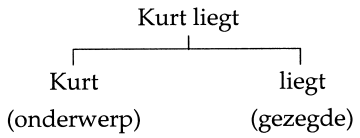
Verwerking van natuurlijke taal

20.1 VOORGESCHIEDENIS

Sinds het begin van de kunstmatige intelligentie als zelfstandige tak van de informatica heeft men zich tot taak gesteld computers natuurlijke taal te doen begrijpen en produceren. In de jaren vijftig leidde dit tot projecten om machinaal teksten van de ene taal in de andere te vertalen. Deze eerste fase is echter stukgelopen op de naïeve veronderstelling dat men door syntactische woord–voor–woordvervanging kan vertalen, zonder rekening te houden met eigenaardigheden van complexe grammaticale constructies en – vooral – van verschillende betekenisaspecten in de betrokken talen. Aan de constructie van een vertaalmachine moet blijkbaar een exacte analyse van grammaticale en betekenisstructuren in natuurlijke talen voorafgaan. En hier komt de logica weer in het spel: reeds eeuwenlang bestaat contact tussen logica en taalkunde over de onderliggende vorm van taaluitingen.
Aan de vorm van taal vallen twee perspectieven te onderscheiden, grammaticale en logische vorm: taalkundigen zijn geïnteresseerd in grammaticale structuren, terwijl logici zich richten op die vormen voor zinnen die hun rol in geldig redeneren goed weergeven (binnen de taalkunde noemt men dit de logische vorm). Dit kan tot verschillen in weergave leiden:

Voorbeeld 20.1

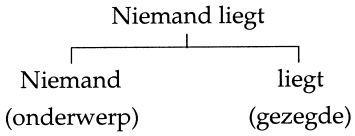
De zin 'Kurt liegt' heeft als 'taalkundige vorm':



Dit sluit goed aan bij zijn logische vorm:

$L(k)$

De zin 'Niemand liegt' heeft dezelfde grammaticale structuur als 'Kurt liegt':



De logische vorm is echter beslist anders:

$\neg\exists x\, L(x)$

In termen van betekenis: in het tweede geval is er geen 'object' waarvan beweerd wordt dat het liegt.

Het onderscheid tussen grammaticale en logische vorm is lang beschouwd als een diepe kloof tussen taalkunde en logica. We kunnen beide gezichtspunten echter met elkaar verzoenen door op een systematische manier logische betekenissen te *koppelen* aan taalkundige structuren. Er bestaan diverse manieren om dit te bewerkstelligen.

Interfaces

Ook buiten de sfeer van vertaling tussen natuurlijke talen is de toenadering tussen taalkunde en logica van belang. Op vele plaatsen wenst men tegenwoordig 'natuurlijke taal interfaces' aan te brengen tussen gebruikers en een computersysteem. Dit kan onder andere met een vertaalslag tussen de (natuurlijke) taal van de gebruiker en de logische vormen waarmee de computer raad weet.
Bijvoorbeeld, bij bevragen in natuurlijke taal van een gegevensbank moet men vertalen van gewone taal naar predikaatlogische formules (of een ander formalisme). Met die vertaling kan de gegevensbank dan geconfronteerd worden (zoals bijvoorbeeld bij het bevragen van logische programma's, zie hoofdstuk 16). De uitkomsten kunnen eventueel weer voor de gebruiker in natuurlijke taal terugvertaald worden.
Nu hebben we in hoofdstuk 6 wel de 'kunst' besproken van vertalen tussen predikaatlogica en natuurlijke taal, maar een echte mechanische procedure was dat zeker niet. Men is tegenwoordig veel dieper doorgedrongen in de hiermee gemoeide taalkundige, logische en computationele details. In dit hoofdstuk geven we een glimp van een methodiek op dit gebied, te weten de zogenaamde 'categoriale grammatica'. Die benadering sluit goed aan bij het reeds ontwikkelde logische apparaat.

Overigens heeft deze interesse voor natuurlijke taal vanuit de informatica tot gevolg gehad dat sommige onderzoekers geneigd zijn natuurlijke talen als programmeertalen te beschouwen (zie hoofdstuk 21 voor een concreet voorbeeld).

20.2 TAALVORM IN CATEGORIALE GRAMMATICA

Categoriale grammatica

De kerngedachte van de zogenaamde categoriale grammatica is dat het mechanisme in natuurlijke taal dat complexe uitdrukkingen opbouwt, overeenkomt met het *toepassen van functies* op argumenten. Bijvoorbeeld, in de zin 'Kurt liegt' staat de eigennaam 'Kurt' voor een individueel object, waarop het predikaat 'liegt' wordt toegepast om een waarheidswaarde te leveren. Het werkwoord 'liegt' is dus een functionele categorie die objecten als argument neemt en waarheidswaarden levert. Waarheidswaarden corresponderen met de categorie der zinnen van de taal. In de zin 'Niemand liegt' daarentegen staat de naamwoordelijke uitdrukking 'niemand' niet voor een individueel object maar voor een functie. Die functie wordt toegepast op eigenschappen als 'liegt' om een waarheidswaarde te leveren, te weten: 'waar' als die eigenschap voor geen mens opgaat en 'onwaar' anders.
Ook 'hogere' soorten functies treden op, zoals de voorgaande functie voor 'niemand' die zelf op andere 'lagere' functies opereert. Dit procédé is zelfs nog verder herhaalbaar. Het verband tussen grammaticale categorieën en functionele typen werd in hoofdstuk 12 uitgelegd.

Argumentposities

In standaardlogica wordt de positie van een argument louter door afspraak bepaald. In natuurlijke talen echter is syntactische volgorde doorgaans heel belangrijk. Daarom gebruiken we de volgende 'codering': functionele uitdrukkingen die *links* een argument van type A willen om een waarde van type B te leveren, hebben type $A \rightarrow B$; functionele uitdrukkingen die *rechts* een argument van type A willen om een waarde van type B te leveren, hebben type $B \leftarrow A$.

We geven als voorbeeld nu eerst een fragment van natuurlijke taal met een bijbehorende keuze van functionele categorieën, daarna twee grammaticale analyses in dit fragment.

Voorbeeld 20.2

Grammaticale categorieën

grammaticale categorie	*voorbeeld*	*type*
eigennamen	Kurt, Alfred	E
zinnen		Z
onovergankelijke werkw.	liegt, hoest	$E \rightarrow Z$
overgankelijke werkw.	slaat, bemint	$(E \rightarrow Z) \leftarrow E$
zelfstandige naamwoorden	kind, programma	ZN
naamwoordsgroep	niemand, ieder kind	$Z \leftarrow (E \rightarrow Z)$
determinatoren	geen, ieder	$(Z \leftarrow (E \rightarrow Z)) \leftarrow ZN$

Voorbeeld 20.3

Grammaticale analyse

$$\dfrac{\begin{array}{cc}\text{Kurt} & \text{liegt} \\ E & E \rightarrow Z\end{array}}{Z}$$

$$\dfrac{\dfrac{\begin{array}{cc}\text{Geen} & \text{kind} \\ (Z \leftarrow (E \rightarrow Z)) \leftarrow ZN & ZN\end{array}}{Z \leftarrow (E \rightarrow Z)} \qquad \begin{array}{c}\text{liegt} \\ E \rightarrow Z\end{array}}{Z}$$

We zien het volgende mechanisme van taalkundige ontleding aan het werk. Aan enkelvoudige woorden in een uitdrukking worden functionele categorieën toegekend, waarna wordt geprobeerd om voor de hele uitdrukking een categorie 'af te leiden' door gebruik van twee regels van functietoepassing:

$$\dfrac{A \qquad A \rightarrow B}{B} \qquad \dfrac{B \leftarrow A \qquad A}{B}$$

Dit eenvoudige schema kan bij herhaling zeer complexe zinsstructuren verantwoorden. Bijvoorbeeld, we krijgen *oneindig* veel naamwoordsgroepen wanneer we deze recursief opbouwen met behulp van voorzetsels:
ieder kind *met* een programma
ieder kind *met* een programma *voor* ieder kind
enzovoorts.

20.3 ONTLEDEN ALS AFLEIDEN

In het voorgaande werd de gangbare taalkundige spreekwijze gevolgd van 'afleiden' van een grammaticale structuur. In feite heeft dat in logisch perspectief een diepere zin: de voorgaande methode *is* inderdaad een vorm van logisch afleiden! Immers, wanneer we de functionele pijlen lezen als implicaties, dan komen de twee categoriale combinatieregels beide neer op de afleidingsregel Modus Ponens, ofwel, de 'implicatie–eliminatie' in de natuurlijke deductie (zie de hoofdstukken 4 en 10). Met andere woorden, categoriaal ontleden komt neer op het zoeken naar een logische afleiding van de gewenste categorie (op te vatten als een formule met implicaties) uit een rij categorieën die als aannames kunnen worden beschouwd. Categoriaal ontleden leent zich dus voor implementatie via de diverse bewijsmethoden voor de propositielogica.

We geven in deze paragraaf enkele illustraties van kwesties over taalkundig ontleden als afleiden met natuurlijke deductie.

Voorbeeld 20.4

Om te beginnen kiezen we als eenvoudig categoriaal systeem de syntaxis van de propositielogica zelf. We werken met een 'basistype' F voor formules, functioneel type $F \leftarrow F$ voor eenplaatsige logische connectieven ($\neg$) en functioneel type $(F \rightarrow F) \leftarrow F$ voor tweeplaatsige logische connectieven ($\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$).
Hoe kunnen we laten zien dat de uitdrukking $p \wedge \neg q$ een formule is? We schrijven de overeenkomstige rij categorieën op:

p	F
$\wedge$	$(F \rightarrow F) \leftarrow F$
$\neg$	$F \leftarrow F$
q	F

Hieruit leiden we de categorie F af:

$$\dfrac{\dfrac{F}{\dfrac{(F \rightarrow F) \leftarrow F \qquad \dfrac{F \leftarrow F \qquad F}{F}}{F \rightarrow F}}{F}}{F}$$

Voorbeeld 20.5

Ambiguïteit
Een dergelijke afleiding is niet altijd uniek. Zo heeft de uitdrukking $\neg p \wedge q$ twee verschillende afleidingen (vergelijk voorbeeld 2.8):

a
$$\dfrac{\dfrac{\dfrac{(F \to F) \leftarrow F \qquad F}{F} \qquad F \to F}{F \leftarrow F} \qquad F}{F}$$

b
$$\dfrac{\dfrac{F \leftarrow F \qquad F}{F} \qquad \dfrac{(F \to F) \leftarrow F \qquad F}{F \to F}}{F}$$

Deze twee afleidingen staan voor twee grammaticale leeswijzen – in de taalkunde heel gebruikelijk – voor onze uitdrukking. Deze vertegenwoordigen intuïtief twee verschillende mogelijke betekenissen, afhankelijk van het bereik van de operator $\neg$:

a $\neg(p \wedge q)$ 'groot bereik'
b $(\neg p \wedge q)$ 'klein bereik'

De regel voor implicatie–introductie

Een voor de hand liggende vraag is of categoriaal ontleden niet meer is dan het herhaald toepassen van de $\to$E–regel, of dat ook andere regels van natuurlijke deductie een rol spelen, met name de *introductieregel* voor implicatie $\to$I. Dit laatste is inderdaad het geval:

Voorbeeld 20.6

Implicatie–introductie
In een uitdrukking als 'slaat ieder kind' willen we het overgankelijk werkwoord 'slaat' combineren met de naamwoordsgroep 'ieder kind', zodanig dat een uitdrukking ontstaat die zelf weer net zo kan functioneren als een onovergankelijk werkwoord. Om overbodige complexiteit te vermijden zien we in dit voorbeeld even af van de richting waarin functionele categorieën hun argumenten toegeschoven krijgen en lezen alles van links naar rechts.
Er dreigt een lastig probleem: geen der categorieën $E \to (E \to Z)$ (van 'slaat') en $(E \to Z) \to Z$ (van 'ieder kind') kan de ander als argument nemen, zodat functietoepassing niet werkt. Met behulp van de $\to$I–regel valt niettemin een natuurlijke deductie te construeren die het gewenste

resultaat $E \rightarrow Z$ ('slaat ieder kind') oplevert. We nemen daartoe tijdelijk een hulpaanname E aan die we later weer intrekken:

$$\dfrac{\dfrac{\dfrac{\dfrac{1}{E} \qquad E \rightarrow (E \rightarrow Z)}{E \rightarrow Z}\,{\rightarrow}\mathrm{E} \qquad (E \rightarrow Z) \rightarrow Z}{Z}\,{\rightarrow}\mathrm{E}}{E \rightarrow Z}\,{\rightarrow}\mathrm{I},\,[-1]$$

In het bewijssysteem voor ontleden in natuurlijke taal spelen dus beide bewijsregels voor implicatie uit hoofdstuk 4 een rol.

Voorkomens en volgorde

Verdere subtiliteiten hebben te maken met een belangrijk verschil in gezichtspunt: in grammaticale ontleding gaat het om *geordende* reeksen van *voorkomens* van woorden, in logisch afleiden tot nu toe om *verzamelingen* aannames, zonder zorg om hun volgorde of aantal voorkomens. Tot nu toe waren principes als de volgende triviaal geldig:

- Uit $\varphi_1, \varphi_2 \vdash \psi$ volgt $\varphi_2, \varphi_1 \vdash \psi$.
- Uit $\varphi \vdash \psi$ volgt $\varphi, \varphi \vdash \psi$.

Maar voor ontleden zijn beide implausibel: de eerste zou bijvoorbeeld elke permutatie binnen correcte zinnen ook herkennen ('verdraaien'), en de tweede zou willekeurig herhalen van een woord in een zin toestaan ('stotteren'). Een precieze logica voor natuurlijke taal vormt nog steeds onderwerp van onderzoek (zie hoofdstuk 21 voor de daarbij opgekomen 'logica van voorkomens').

20.4 SEMANTIEK VAN CATEGORIALE UITDRUKKINGEN

In ontleden als afleiden wordt de belangrijke logische informatie niet zozeer gedragen door een 'vlakke' uitdrukking (die we in de afleidingsboom onderaan terugvinden), maar door de bijbehorende afleidingen naar uiteindelijke categorieën. Zoals we zagen is het daarbij heel goed mogelijk, dat een uitdrukking verschillende afleidingen naar eenzelfde categorie krijgt. Dit introduceert een interessante nieuwe gedachte omtrent onze eerdere bewijssystemen: niet 'alle wegen naar het doel' van een geldige gevolgtrekking zijn equivalent. Verschillende bewijzen kunnen verschillende semantische informatie dragen.
'Verschil in betekenis' kunnen we preciseren door aan de hand van categoriale afleidingen systematisch *berekeningsvoorschriften* op te bouwen voor de semantische waarde der ontlede uitdrukking. We noteren de interpretatie van een uitdrukking met categorie A als $\beta(A)$.

We gaan nu twee semantische regels definiëren voor de categoriale ontleedregels →E en →I.

Semantiek van de →E–regel

De interpretatie van de →E–regel is bijzonder eenvoudig. De motivering voor deze regel was immers *functietoepassing*:

DEFINITIE 20.1

→E betekent functietoepassing

$$\frac{A \qquad A \to B}{B} \qquad \frac{\beta(A) \qquad \beta(A \to B)}{\beta(A \to B)(\beta(A))}$$

Voor $B \leftarrow A$ gaat het evenzo.

Voorbeeld 20.7

Voor de eerdere twee afleidingen van $\neg p \wedge q$ construeert men eenvoudig de bijbehorende semantische waarden. De volgende analyse correspondeert met de lezing $(\neg p \wedge q)$:

$$\frac{\dfrac{\beta(\neg) \qquad \beta(p)}{\beta(\neg)(\beta(p))} \qquad \dfrac{\beta(\wedge) \qquad \beta(q)}{\beta(\wedge)(\beta(q))}}{\beta(\wedge)(\beta(q))(\beta(\neg)(\beta(p)))}$$

Semantiek van de →I–regel

De semantische duiding van de introductieregel →I vergt een geavanceerder begrip, afkomstig uit de lambda–calculus (zie hoofdstuk 12). Semantisch geeft een afleiding van ψ uit een stel aannamen $\Sigma + \varphi$ ons een systematisch recept om een betekenis voor ψ op te bouwen met behulp van betekenissen voor de uitdrukkingen in Σ plus één voor φ. Met andere woorden, we hebben een methode om een betekenis voor ψ te vinden bij gegeven betekenis voor φ en gelijkblijvende 'achtergrond' voor Σ. Maar dan kunnen we juist een functie *introduceren*, en wel door *λ–abstractie*:

DEFINITIE 20.2

→–Introductie betekent functieabstractie
Als een afleidingsboom voor ψ een voorschrift V heeft opgeleverd waarin betekenissen voor Σ en voor φ voorkomen (met de betekenis van φ bijvoorbeeld aangegeven door een variabele x), dan is $\lambda x\ .\ V$ het voorschrift om een betekenis van $\varphi \to \psi$ te verkrijgen uit de betekenissen van Σ alleen.

Voorbeeld 20.8

Een concreet voorbeeld van deze semantische regel (zie voorbeeld 20.6):

$$\dfrac{\dfrac{\overset{1}{x_E} \qquad \beta(\text{slaat})}{\beta(\text{slaat})(x_E)} \qquad \beta(\text{ieder kind})}{\beta(\text{ieder kind})(\beta(\text{slaat})(x_E))}$$
$$\dfrac{\beta(\text{ieder kind})(\beta(\text{slaat})(x_E))}{\lambda x_E.\, \beta(\text{ieder kind})(\beta(\text{slaat})(x_E))}\ [-1]$$

Met andere woorden, we hebben 'slaat' een tijdelijk argument x gegeven, dat we later weer intrekken. Als betekenis houden we de eigenschap over om ieder kind te slaan.

Er valt veel meer te zeggen over de precieze berekeningswijze hier en de speciale klasse der logische betekenisvoorschriften die zich in natuurlijke talen nu echt voordoen, met andere woorden over de 'logische uitdrukkingskracht' van natuurlijke talen. Maar als introductie is het voorgaande genoeg geweest.

20.5 IMPLEMENTATIE

Voor het implementeren van de voorgaande ideeën zijn er diverse methoden. Historisch het bekendst is logisch programmeren (zie hoofdstuk 16), waar 'parsing as deduction' (ontleden als afleiden) een bekend toepassingsgebied vormt. Het essentiële idee is de volgende representatie.

Implementatie in logische programma's

We beschrijven een rij uitdrukkingen via binaire predikaten tussen 'posities' in de rij. Bijvoorbeeld, de rij 'Geen programma werkt' laat zich vertalen in drie basisfeiten:

- 'Geen'(1, 2), 'programma'(2, 3), 'werkt'(3, 4)

De predikaten tussen aanhalingstekens geven aan dat het bijbehorende woord tussen de twee genoemde argumentposities staat. De toekenningen van basistypen aan woorden vertalen we vervolgens in regels als:

- $\forall x\, \forall y$ ('programma'$(x, y) \rightarrow$ '$ZN'(x, y)$)
- $\forall x\, \forall y$ ('werkt'$(x, y) \rightarrow$ '$E \rightarrow Z'(x, y)$)

De combinatieregels van de categoriale grammatica voegen ten slotte, in het eenvoudigste geval van functietoepassing met een argument links, alle instanties toe van het volgende schema:

- $\forall x\, \forall y\, \forall z$ (('$A'(x, y) \wedge$ '$A \rightarrow B'(y, z)) \rightarrow$ '$B'(x, z)$)

Analoog voor zijn rechter tegenhanger.

Ontleden via een dergelijk programma komt dan neer op het stellen van de atomaire JA/NEE–vraag '$A'(1, n)$? A is hier de gewenste eindcategorie, en n is het totaal aantal posities ingenomen door de invoer.
In de berekening van het antwoord op deze vraag zal de resolutie een 'weerleggingsboom' van mogelijkheden nagaan, waarbij alle mogelijke grammaticale analyses gevonden worden.

Voorbeeld 20.9

Propositielogica
Beschouw een logisch programma voor de eerdere beschrijving van propositionele formules. Met invoerfeiten
- '$\neg'(1, 2)$, '$p'(2, 3)$, '$\wedge'(3, 4)$, '$q'(4, 5)$

en ontleedvraag
- '$F'(1, 5)$?

test resolutie de unificatie met geschikte feiten of programmaregels. Wat uiteindelijk succes oplevert zijn zowel aanroepen van een regel voor negatie, via een eerste opsplitsing in doelen:
- '$F \leftarrow F'(1, x)$, '$F'(x, 5)$

als het aanroepen van een regel voor conjunctie, via een eerste opsplitsing in doelen
- '$F'(1, x)$, '$F \rightarrow F'(x, 5)$.

Merk overigens op dat de boom ook falende takken bevat – bij ongelukkige keuze van een rekenregel – zoals de oneindige regressie:
- '$F'(1, x)$, '$F \rightarrow F'(x, 5)$
- '$F'(1, y)$, '$F \rightarrow F'(y, x)$, '$F \rightarrow F'(x, 5)$
- '$F'(1, z)$, '$F \rightarrow F'(z, y)$, '$F \rightarrow F'(y, x)$, '$F \rightarrow F'(x, 5)$

enzovoorts.

Een eigenaardigheid van dit proces is nog dat we werken met een *eindig* Herbrand–universum, te weten de n relevante posities, met alleen predikaten daarover. Dit maakt het ontleedprobleem toch vrij eenvoudig, althans vergeleken met andere in hoofdstuk 16 genoemde toepassingen.

Verfijningen: berekening van syntactische structuur

De mogelijke grammaticale structuren van een uitdrukking blijven bij deze manier van programmeren impliciet. We kunnen de machine echter ook dwingen om de syntactische structuur expliciet zichtbaar te maken. Hiertoe dient een voor logisch programmeren typische techniek: voeg aan de eerdere predikaten een *nieuwe argumentplaats* toe die als 'venster' op de oude berekening kan dienen. Bijvoorbeeld, we kunnen werken met nieuwe predikaten '$A'(x, y, u)$: 'van positie x tot positie y loopt een uitdrukking van categorie A met structuur u'.

Een syntactische applicatieoperator *app* op het termuniversum bouwt dan een passende structuur op in het meest rechtse argument, via regels als de volgende:

$$\forall x\, \forall y\, \forall z\, \forall u\, \forall v((\text{'}A\text{'}(x, y, u) \wedge \text{'}A \rightarrow B\text{'}(y, z, v)) \rightarrow \text{'}B\text{'}(x, z, app(v, u)))$$

De ontleedvraag is dan niet langer een JA/NEE–vraag, maar een echte berekening van een u, zodat $\text{'}A\text{'}(1, n, u)$.

Verfijningen: berekening van semantische structuur

Met eenzelfde techniek kunnen we ook *semantische* informatie laten berekenen parallel aan de zinsontleding. In dat geval lezen we het laatste argument als een voorschrift voor interpretatie, opgesteld in een logisch formalisme. Nog een andere mogelijkheid is om naast de positie voor de syntactische constructie nog een vierde argument toe te voegen voor de interpretatie.

Voorbeeld 20.10

Categoriale betekenissen opgebouwd
We voorzien woorden van een categorie en van bedoelde betekenis in de vorm van een bijbehorende constante term, bijvoorbeeld:

$$\text{'programma'}(x, y) \rightarrow \text{'}ZN\text{'}(x, y, \text{PROGRAMMA})$$

Verder dwingen we de eerdere functionele combinatieregel tot het creëren van een functietoepassing waar *'app'* nu staat voor semantische functietoepassing. Dit leidt voor een eerdere uitdrukking als 'Geen programma werkt', met als categoriale bewijsboom:

$$\dfrac{\dfrac{(Z \leftarrow (E \rightarrow Z)) \leftarrow ZN \qquad ZN}{Z \leftarrow (E \rightarrow Z)} \qquad E \rightarrow Z}{Z}$$

tot het volgende antwoord op de vraag $Z(1, 4, u)$:

$$u = app(app(\text{GEEN}, \text{PROGRAMMA}), \text{WERKT})$$

In het geval dat we in een categoriale analyse ook de afleidingsregel van implicatie–introductie hebben gebruikt, zal dit eenvoudige procédé overigens niet toereikend zijn en moeten er in de termen rechts ook op een geschikte manier λ–abstracties worden ingebouwd.

Er is nog een andere manier waarop λ–abstractie in het spel kan komen. Wellicht zal men het zoëven berekende voorschrift niet geschikt vinden, omdat de kwantor 'geen' geen expliciete betekenis heeft. Willen we dat wel, dan moet reeds in de oorspronkelijke toekenning een voorziening worden getroffen die latere argumenten voor deze functie 'voorziet'. En daarvoor is λ–abstractie geschikt:

$$\forall x\, \forall y\, (\text{'geen'}(x, y) \rightarrow \text{'}(Z \leftarrow (E \rightarrow Z)) \leftarrow ZN\text{'}(x, y, \text{'}\lambda P\, .\, \lambda Q\, .\, \neg\exists x\, (P(x) \wedge Q(x))\text{'}))$$

De eerder berekende semantische term krijgt nu in plaats van 'GEEN' deze λ–uitdrukking. De uiteindelijke semantische waarde laat zich dan berekenen met λ–*conversie* (zie hoofdstuk 12):

$$\begin{aligned}&(\lambda P\, .\, \lambda Q\, .\, \neg\exists x\, (P(x) \wedge Q(x)))(\text{PROGRAMMA})(\text{WERKT})\\ &= (\lambda Q\, .\, \neg\exists x\, (\text{PROGRAMMA}(x) \wedge Q(x)))(\text{WERKT})\\ &= \neg\exists x\, (\text{PROGRAMMA}(x) \wedge \text{WERKT}(x))\end{aligned}$$

Hiermee besluiten we ons korte overzicht van meer concrete programmeertechnieken voor de omzetting van natuurlijke taal in logische formalismen.

Er zij overigens nog vermeld dat de categoriale grammatica ook heel anders geïmplementeerd kan worden dan in deze paragraaf is gebeurd, aangezien onze eerdere calculus voor categorieën ook rechtstreeks kan worden aangepakt als een toepassing van automatisch bewijzen van stellingen.
Bovendien doen zich nog vele andere mogelijkheden voor van programmeertaken in dit gebied.
Zo kunnen wij bij de ontledingsvraag '$A'(x, y, u, v)$?' (posities x en y, syntaxis u en semantiek v) de vraag van *taalproductie* stellen: gegeven een logische betekenis v, hoe die tussen gegeven posities x en y onder woorden (= u) te brengen?

Hiermee zijn overigens nog lang niet alle aspecten van natuurlijke taalverwerking in de kunstmatige intelligentie uitgeput.

20.6 OPGAVEN

20.1 a Geef een categoriale afleiding van de volgende zinnen:
- Kurt slaat elk kind
- niemand hoest
- elk kind bemint Alfred

- elk programma liegt
- Kurt bemint niemand

b Geef verschillende categoriale afleidingen van de zin:

- elke jongen houdt van een meisje

c Geef een categoriale analyse van de woorden 'of', 'terwijl' en 'met', en eveneens van de woorden 'is' en 'bang' zoals die voorkomen in de zin 'Kurt is bang'.

d Geef semantische ontledingen van de zinnen onder a en b in termen van de lambda–calculus. Hoe zou de Prolog–parser dit bewerkstelligen?

20.2 De zogenaamde 'termlogische' taal is de predikaatlogische taal zonder kwantificatiesymbolen. Deze taal bevat dus:

- individuele variabelen en constanten
- predikaatsymbolen
- logische connectieven

a In voorbeeld 20.4 is een categoriale analyse van de propositielogische taal beschreven. Geef een soortgelijke analyse van een termlogische taal met als enige predikaatsymbool het identiteitsteken.

b Laat met behulp van de hiervoor gevonden categoriale analyse zien dat de volgende rijtjes formules zijn van deze taal. We gebruiken de Poolse notatie zoals in opgave 5.6:

- $= x\ x$
- $\rightarrow = x\ y = x\ y$
- $\wedge = x\ y = y\ z = x\ z$

20.3 a Bewijs dat het het type $(E \rightarrow Z) \rightarrow Z$ afleidbaar is onder aanname van E.

b Bedenk dat E semantisch gezien de categorie der objecten is en Z de categorie der waarheidswaarden. De afleiding onder a laat dus zien dat objecten ook categorie $(E \rightarrow Z) \rightarrow Z$ bezitten. Waarmee wordt een object in dit geval semantisch geïdentificeerd?

c Bewijs dat voor willekeurige typen A, B en C, het type $(A \rightarrow B) \rightarrow (A \rightarrow C)$ afleidbaar is onder aanname van $B \rightarrow C$.

d Laat nu zien dat negatie en conjunctie, gedefinieerd als operatoren op *zinnen* (waarheidswaarden), ook kunnen dienen als operatoren op onovergankelijke werkwoorden (predikaten).

Blok VII

Tot besluit

Vooruitblik

Hoofdstuk 21

Vooruitblik

In dit boek is een eerste kennismaking met de moderne logica gegeven, als een algemene benadering van redeneren in de ruimste zin des woords, met het daartoe ontwikkelde technische instrumentarium. Bovendien zijn enkele doorkijkjes gegeven naar moderne toepassingen, vooral in de informatica, maar ook enigszins in de wiskunde en zelfs hier en daar de taalkunde.
Goed toepassen is echter geen zaak van louter eenrichtingsverkeer. Zo heeft de toepassing van de logica in de wiskunde sinds het midden van de negentiende eeuw de logica zelf verrijkt en grondig veranderd. Diverse logici verwachten tegenwoordig dat iets dergelijks zou kunnen voortkomen uit het steeds intensievere contact met de informatica. Wat betreft verrijking van de logica door de informatica kunnen we in ieder geval al enkele verschijnselen noteren. Zo werpt de informatica een nieuw licht op oude discussies over het belang van alternatieve logische benaderingen, zoals de discussie over klassieke versus intuïtionistische logica in hoofdstuk 12. Evenzo is vanuit de informatica meer aandacht gekomen voor de gedetailleerde structuur van bestaande systemen, zoals het predikaatlogische fragment van de Horn–clausules uit hoofdstuk 16 illustreerde.
Maar er is ook een indringender manier waarop computationele invloeden zich kunnen doen gevoelen. In het voorgaande bleven we aannemen dat althans de basisstructuur van onze standaardlogische systemen dezelfde blijft in het contact met de informatica: de 'stukken' staan misschien anders in de toepassing, maar het blijven wel dezelfde stukken. Ook dit valt echter te relativeren, en aan enkele voorbeelden van een dergelijke heroverweging is dit laatste hoofdstuk gewijd. Lezers die meer van vastigheid houden, kunnen op dit punt beter het boek sluiten. Maar zelfs exacte wetenschap is nu eenmaal net zo vaak het in twijfel trekken van bestaande zekerheden als die steeds maar weer propageren.

21.1 LOGICA VAN DYNAMISCHE INTERPRETATIE

In de logische folklore heeft reeds lang een idee bestaan dat men als een alternatief zou kunnen beschouwen voor de standaardvisie waarop dit boek is gebaseerd. Namelijk, men zou de betekenis van proposities ook kunnen lokaliseren, niet in hun waarheidscondities, maar in de dynamische rol die zij spelen wanneer de erdoor uitgedrukte informatie wordt toegevoegd aan een kennistoestand. In dat perspectief, bijvoorbeeld, is een conjunctie een sequentiële instructie voor 'updating', het bijwerken van informatie, en een disjunctie heeft juist meer iets van een keuze. En redeneren zou verklaard kunnen worden in de volgende trant:

We verwerken de informatie uit achtereenvolgende aannames, en testen dan of de conclusie in de aldus bereikte kennistoestand wordt afgedwongen.

Aldus gezien worden logische talen meer een soort 'cognitieve' programmeertalen, en het ligt voor de hand hier analogieën uit de informatica te exploiteren. Illustraties hiervan zijn te vinden in recente theorieën van 'propositionele dynamica' langs de lijnen gesuggereerd door onze bespreking van 'krimpende informatietoestanden' in hoofdstuk 2. We geven hier echter een ander recent voorbeeld, dynamische (predikaat)logica, dat niet rechtstreeks ingaat op informatieverwerking, maar veeleer op 'variabelenbeheer'. Dit voorbeeld illustreert heel fraai een omkering van het perspectief in hoofdstuk 15, waarin een dynamische, imperatieve programmeertaal onderscheiden wordt van de predikaatlogica als statische declaratieve logische beschrijvingstaal. Overigens wordt de term 'dynamische logica' – enigszins verwarrend – zowel voor deze logica van dynamische interpretatie als voor de dynamische modale logica zoals in hoofdstuk 15 gebruikt.

In de natuurlijke taal zijn persoonlijke voornaamwoorden, met hun wisselende bindingen door een tekst heen, een voor de hand liggend analogon van de variabelen in de predikaatlogica. Maar er lijkt een kloof te gapen tussen persoonlijke voornaamwoorden en variabelen, zodra we ons verdiepen in de details van de vertaling van natuurlijke taal naar predikaatlogische formules.

Voorbeeld 21.1

Wanverhoudingen

a De tekst 'Een man kwam. Hij schoot.' legt een binding tussen de geïntroduceerde man en 'hij' over de eerste zinsgrens heen. In de

predikaatlogische vertaling daarentegen stopt het bereik van de corresponderende kwantor $\exists$ bij de punt: de natuurlijke weergave is immers: $\exists x\ (Mx \wedge Kx)\ .\ Sx$.

b Zo'n binding is niet mogelijk wanneer we de volgorde van de zinnen omdraaien: in de tekst 'Hij schoot. Een man kwam.' is binding tussen 'hij' en 'een man' niet grammaticaal. Maar waarom hier niet, en hiervoor wel?

c Een universele kwantor bindt juist niet over zinsgrenzen heen, getuige de incorrecte tekst 'Iedere man kwam. Hij schoot.', die we zonder bindingsperikelen rechtstreeks kunnnen vertalen tot: $\forall x\ (Mx \rightarrow Kx)\ .\ Sx$.

Een aantal auteurs heeft voorgesteld dat we hier een betere aansluiting verkrijgen zodra we de logische representatietaal zelf dynamisch opvatten, met de mogelijkheid van wisselende bedelingen; net zoals we dat dus deden in de operationele semantiek voor imperatieve programmeertalen. Een elegant systeem van deze aard krijgen we als we predikaatlogische formules φ interpreteren in onze eerdere modellen M als overgangsrelaties $[[\varphi]]$ tussen bedelingen. Men kan hierbij als oorzaak denken aan variërende bedelingen tijdens het nagaan van gekwantificeerde beweringen. We krijgen dan een inductieve interpretatie met kenmerkende gevallen als de volgende:

1 $[[Px]] = \{\langle b, b\rangle \mid M, b \vDash Px\}$
atomaire beweringen fungeren gewoon als tests.
2 $[[\varphi \wedge \psi]] = [[\varphi]] \circ [[\psi]]$
conjunctie wordt relationele compositie (dat wil zeggen 'sequentie').
3 $[[\exists x\ \varphi]] = \{\langle a, b\rangle \mid$ er is een object $d \in D$ zodat $\langle a[x \mapsto d], b\rangle \in [[\varphi]]\}$
we doen een 'random assignment' voor een geschikte x en moeten daarna φ succesvol verwerken.

Deze uitleg verklaart de binding in voorbeeld 21.1a. Wanneer we namelijk 'en' en de punt in onze tekst beide sequentieel lezen, dan kunnen we berekenen dat $[[\exists x\ (Mx \wedge Kx)\ .\ Sx]]$ die relatie tussen bedelingen is, die van een willekeurige b overgaat op een $b[x \mapsto d]$ waarin zowel Mx, Kx als Sx waar zijn. Er is immers geen tussenliggende instructie geweest die de waarde in het x–register nog zou veranderen. Maar omgekeerd is duidelijk dat in de verwerking van $Sx\ .\ \exists x\ (Mx \wedge Kx)$ de waarde van x wel veranderd kan zijn door de kwantor, zodat identificatie in dit geval niet mogelijk is.

De verklaring van het uitblijven van binding in voorbeeld 21.1c vergt een behandeling van andere logische operatoren. Hierop wordt verder niet ingegaan.

Binnen het perspectief van dynamische interpretatie zijn vele aspecten van de predikaatlogica te heroverwegen.

Geldig gevolg

Als notie van geldig gevolg kunnen we bijvoorbeeld overgaan op:

$\varphi_1, ..., \varphi_n \models \psi$ als in elk model M de relatie $[[\varphi_1]] \circ ... \circ [[\varphi_n]]$ bevat is in $[[\psi]]$.

Deze notie van geldig gevolg blijkt zich geheel anders te gedragen dan de gebruikelijke, met name vanwege de gevoeligheid voor formuleringen in de tekst. Zo zal uiteraard het effect van een rij aannames gaan afhangen van hun *volgorde*: iets wat in al onze systemen tot nu toe nooit het geval was.

Logische operatoren

Een ander interessant punt is het opkomen van meer keuzen voor het definiëren van logische operatoren. Zo zijn er *twee* natuurlijke kandidaten voor conjunctie: naast de hier gekozen sequentiële ook een meer parallelle. Parallelle conjunctie komt neer op een Boolese operatie en kunnen we interpreteren als 'succesvolle overgangen voor beide beweringen tegelijk':

$[[\varphi \wedge \psi]] = [[\varphi]] \cap [[\psi]]$

Meer in het algemeen hebben we nu te maken met een relatiealgebra van mogelijke operatoren (zie paragraaf 12.4).

Syntaxis

Ten slotte kan men vanuit dit perspectief zelfs nog eens terugkomen op de predikaatlogische *syntaxis*. De voorgaande interpretatie van de existentiële kwantor in $\exists x\ \varphi$ heeft namelijk hetzelfde effect wanneer er een conjunctie zou hebben gestaan van de vorm $\exists x \wedge \varphi$. Sterker nog, vanuit dit gezichtspunt is er iets voor te zeggen de syntaxis van de predikaatlogica te herschikken:

a basisinstructies zijn atomaire tests, deterministische assignments $x := t$ en random assignments $x := ?$
b constructies maken we met $\wedge$ en $\neg$ en eventuele nieuwe kandidaten, maar niet langer met kwantoren.

De bedoeling van deze bespreking is overigens beslist niet om te suggereren dat de standaardpredikaatlogica nu van de baan is. Zij wil alleen aantonen dat zelfs binnen het traditionele kerngebied nog wel

degelijk ruimte is voor nieuwe gezichtspunten. In de praktijk is er veel voor te zeggen een dynamische predikaatlogica te hanteren in *combinatie* met de oude predikaatlogica, die dan dient als maat voor de klassieke 'informatieve inhoud' van beweringen.

Voor recente ontwikkelingen in deze logica van dynamische interpretatie verwijzen we naar het *Handbook of Logic and Language*, zie de literatuurlijst in appendix 3.

21.2 NIET–STANDAARD REDENEREN

Een ander vast aspect van de predikaatlogica is binnen de kunstmatige intelligentie onderwerp van discussie geworden. Een gedachte die daar gaandeweg is opgekomen (en die overigens ook wortels heeft in de geschiedenis van de logica zelf), is dat er niet een unieke notie van 'geldig gevolg' is, maar eerder een *familie* van zulke noties, in verschillende mate van 'strengheid'. En dan zou een belangrijke taak voor de logica juist zijn het opsporen en bestuderen van zulke verschillende begrippen en hun semantische achtergrond.

Minimaal gevolg

Een belangrijke bron van zulke verschijnselen is verwant aan onze eerdere notie van 'minimaal model' in de semantiek van logisch programmeren. Vaak zijn niet alle modellen van een verzameling aannames voor ons even plausibel. We kunnen voorkeuren, 'preferenties' hebben, bijvoorbeeld voor situaties die zo specifiek mogelijk zijn, zodat de kans op het voorkomen van uitzonderingen voor onze heuristische regels zo gering mogelijk is. Een voorbeeld hiervan is het redeneren met inductie: uit een aantal geobserveerde gevallen willen we tot een regelmaat besluiten, wat we doen door alleen die gevallen en geen andere in het universum van onze modellen op te nemen. Technisch valt dit te omschrijven volgens de methode van circumscriptie, die in hoofdstuk 18 reeds kort aangestipt is (zie definitie 18.9):

Een formule φ *volgt minimaal* uit Σ, notatie $\Sigma \models_{\min} \varphi$, indien φ waar is in alle modellen voor Σ met een zo klein mogelijk domein (dat wil zeggen, in die M met $M \models \Sigma$ waarvan geen submodel met echt kleiner domein Σ nog waar maakt).

Er zijn ook varianten waar de preferentierelatie tussen de modellen anders totstandkomt, bijvoorbeeld door minimaliseren van predikaten.

Het effect van redeneren met minimaal gevolg is de winst aan geldige conclusies: doordat we minder modellen voor Σ in aanmerking nemen dan in de standaarddefinitie, vallen vele potentiële tegenvoorbeelden af.

Voorbeeld 21.2

$\forall x\, Px$ volgt minimaal uit $\exists x\, Px$, in notatie: $\exists x\, Px \models_{\min} \forall x\, Px$.

Minimaal gevolg is niet–monotoon

De prijs hiervoor is wel dat, bij groei van informatie, eerdere conclusies wellicht moeten worden opgegeven: niet–monotonie.

Voorbeeld 21.3

Niet geldig is bijvoorbeeld: $\exists x\, Px$, $\exists x\, \neg Px \models_{\min} \forall x\, Px$.

Minimaal gevolg is niet–transitief

Het lijkt erop alsof de logische prijs voor zulke noties van minimaal gevolg tamelijk hoog is. Niet–monotonie betekent reeds het verdwijnen van het vertrouwde klassieke verschijnsel van 'cumulatie van conclusies'. Maar ook bijvoorbeeld 'schakeling van conclusies' verdwijnt, want, zoals eenvoudig valt na te gaan, minimaal redeneren is eveneens niet–transitief:

Voorbeeld 21.4

$\exists x\, Px \wedge \exists x\, \neg Px \models_{\min} \exists x\, Px$ en $\exists x\, Px \models_{\min} \forall x\, Px$, maar $\exists x\, Px \wedge \exists x\, \neg Px \not\models_{\min} \forall x\, Px$.

Echter, als de prijs zo hoog is, waarom zou de standaardlogica ertoe neigen zulke ongewenste verschijnselen in zich op te nemen? Hier bedriegt de schijn: bij nadere beschouwing blijken niet–standaard noties van gevolg wel degelijk allerlei interessante eigenschappen over te houden: we moeten alleen wat beter kijken dan in het standaardgeval nodig was.

Minimaal gevolg is voorzichtig monotoon

Minimaal gevolg is bijvoorbeeld nog steeds 'voorzichtig monotoon'. Er valt te bewijzen dat conjunctie met eenmaal bereikte conclusies geen geldige gevolgen aantast:

Als $\Sigma \models_{\min} \varphi$ en $\Sigma \models_{\min} \psi$, dan $\Sigma \cup \{\varphi\} \models_{\min} \psi$.

Minimaal gevolg is voorzichtig transitief

Evenzo is minimaal gevolg 'voorzichtig transitief':

Als $\Sigma \models_{\min} \varphi$ en $\Sigma \cup \{\varphi\} \models_{\min} \psi$, dan $\Sigma \models_{\min} \psi$.

Conditionele beweringen

Schuilt er een systeem achter dergelijke eigenschappen? Het antwoord is bevestigend en komt uit een reeds bestaande tak van de standaardlogica, te weten de studie van 'conditionele beweringen' in de modale logica. Traditioneel werd daar reeds lang gewerkt met de uitleg van

implicaties in modellen van mogelijke werelden, met als ordening een preferentierelatie van 'relatieve nabijheid' van een wereld ten opzichte van de wereld van evaluatie. Ter onderscheiding van de gewone implicatie gebruiken we hier de pijl $\Rightarrow$ in plaats van $\rightarrow$:

$w \models \varphi \Rightarrow \psi$ desda *voor alle dichtstbijzijnde* v met $v \models \varphi$ geldt ook $v \models \psi$.

Deze semantiek geeft aanleiding tot de volgende 'minimale logica' van conditioneel redeneren:

C1	$\vdash \varphi \Rightarrow \varphi$	reflexiviteit
C2	$\varphi \Rightarrow \psi \vdash \varphi \Rightarrow (\psi \vee \xi)$	verzwakking
C3	$\varphi \Rightarrow \psi, \varphi \Rightarrow \xi \vdash \varphi \Rightarrow (\psi \wedge \xi)$	conjunctie
C4	$\varphi \Rightarrow \psi, \xi \Rightarrow \psi \vdash (\varphi \vee \xi) \Rightarrow \psi$	disjunctie
C5	$\varphi \Rightarrow \psi, \varphi \Rightarrow \xi \vdash (\varphi \wedge \psi) \Rightarrow \xi$	voorzichtige monotonie

Dit blijkt nu ook de logica te zijn voor de hiervoor beschreven relatie van minimaal gevolg. Soortgelijke resultaten zijn te zoeken voor andere families van niet–standaard gevolg. Met andere woorden, het hiervoor genoemde ruimere programma van onderzoek naar varianten op het geldigheidsbegrip heeft zelfs binnen de standaardpredikaatlogica wel degelijk perspectieven.

Voor recente ontwikkelingen in niet-standaard redeneren verwijzen we naar het *Handbook of Logic in Artificial Intelligence and Logic Programming*, zie de literatuurlijst in appendix 3.

21.3 LOGICA VAN VOORKOMENS

Als laatste voorbeeld van een mogelijke paleisrevolutie kiezen we een meer bewijstheoretisch thema. In de behandeling van natuurlijke deductie in de hoofdstukken 4 en 10 is herhaaldelijk opgemerkt dat logisch redeneren zich niet bekommert om aantallen *voorkomens* van gegevens: we gebruiken een aanname net zo vaak als we haar nodig hebben. Maar er zijn wel degelijk redenen om hier soms een scherpere bril op te zetten. Wanneer we namelijk denken aan het rekenproces binnen een bewijs, dan telt wel degelijk iedere aanroep van een aanname als een handeling met kosten. Zoals Jean–Yves Girard, een van de pioniers op dit gebied, pleegt te zeggen: 'kopiëren kost ook geld'. Aldus ontstaat het idee om deductie nog eens te heroverwegen als het manipuleren van *voorkomens* van formules. Sequenten $\Sigma \circ \psi$ (zie paragraaf 11.4) moeten we dan uiteraard gaan lezen met voor Σ niet

langer een verzameling formules, maar een 'multi–set' of 'bag' van voorkomens van formules. En bijvoorbeeld de regel $\rightarrow$I gaat dan als volgt luiden:

Trek een specifiek voorkomen van φ in (niet allemaal tegelijk!) om tot $\varphi \rightarrow \psi$ te concluderen.

De afleidbare sequenten zullen hiermee sterk veranderen, omdat nu de 'multipliciteit' van formules gaat tellen.

Voorbeeld 21.5

Wel afleidbaar blijft $\circ\ (\varphi \rightarrow \psi) \rightarrow ((\psi \rightarrow \xi) \rightarrow (\varphi \rightarrow \xi))$, maar niet afleidbaar wordt $\circ\ (\varphi \rightarrow (\psi \rightarrow \xi)) \rightarrow ((\varphi \rightarrow \psi) \rightarrow (\varphi \rightarrow \xi))$.

Naast een computationele motivering heeft het bijhouden van voorkomens ook diverse andere goede gronden. Een voorbeeld hiervan is het volgende.

Categoriale grammatica en logica met voorkomens

Bij de verwerking van natuurlijke taal in hoofdstuk 20 werd gebruikgemaakt van categoriale grammatica's die categorieën voor zinsdelen combineren via functietoepassing. Dit is een voorbeeld van een bekende logische analogie tussen *implicaties* $\varphi \rightarrow \psi$ en *functies* van argumenten van type φ naar waarden van type ψ.
Modus Ponens ($\rightarrow$E) op het eerste niveau correspondeert met functietoepassing op het andere, en de analogie zet zich verder voort tussen implicatie–introductie ($\rightarrow$I) en lambda–abstractie. Bij deze analogie komen we dan weer uit op een voorkomensvariant van natuurlijke deductie. We lezen immers een sequent zoals $\varphi, (\varphi \rightarrow \psi) \circ \psi$ als 'functie $\varphi \rightarrow \psi$ neemt argument φ om waarde ψ af te leveren'. Dit is niet mogelijk met de sequent $\varphi, \varphi, (\varphi \rightarrow \psi) \circ \psi$. Er blijft nu immers een argument φ ongebruikt over.

Lineaire logica

Het resulterende systeem van natuurlijke deductie is een natuurlijke logica die ver 'beneden' de systemen van hoofdstuk 4 ligt. Deze logica staat bekend als *lineaire logica*.
Net als bij de logica van dynamische interpretatie is er een uitbreiding van het aantal natuurlijke logische operatoren. Zo heeft een voorkomenslogica weer minstens twee niet–equivalente kandidaten voor 'conjunctie', waarvan we de verschillende introductieregels tonen:

a een bewijs voor $\Sigma \circ \varphi$ en een bewijs voor $\Sigma \circ \psi$ combineren tot een bewijs voor $\Sigma \circ \varphi \wedge_1 \psi$;
b een bewijs voor $\Sigma \circ \varphi$ en een bewijs voor $\Delta \circ \psi$ combineren tot een bewijs voor $\Sigma \cup \Delta \circ \varphi \wedge_2 \psi$.

Ook ontbreken weer diverse globale eigenschappen van standaard-afleidbaarheid, zoals monotonie en in dit geval ook contractie van gelijke voorkomens van aannames (uit een bewijs voor $\varphi, \varphi \circ \psi$ concluderen tot een bewijs voor $\varphi \circ \psi$).
Deze verschijnselen zijn ten dele weer te begrijpen in samenhang met het dynamische perspectief in paragraaf 21.1. Een typische voorkomenslogica ontstaat namelijk ook wanneer we formules lezen als beschrijvingen van binaire overgangsrelaties op een of ander domein (zeg van informatietoestanden). Dan kunnen we de eerdere relatiealgebra weer in het spel brengen, en geldigheid definiëren middels de volgende afspraak:

De compositie der relaties voor de aannames van een sequent moet bevat zijn in die voor de conclusie.

In dat geval gaan voorkomens weer essentieel tellen. Bijvoorbeeld, met φ correspondeert de relatie $[[\varphi]]$, met een herhaling φ, φ daarentegen de compositie $[[\varphi]] \circ [[\varphi]]$. Dat is in het algemeen niet dezelfde relatie, tenzij $[[\varphi]]$ transitief en dicht is.

Met andere woorden, zelfs schijnbaar triviale aspecten van logische syntaxis en bewijstheorie, zoals voorkomens van formules versus formules zelf, blijken nog onvermoede vragen te verbergen.

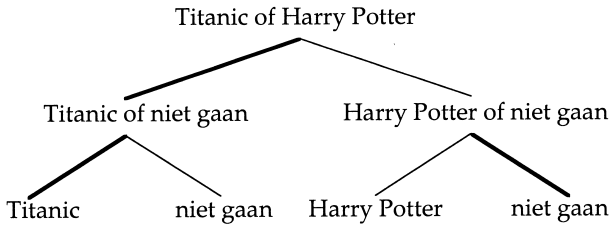
21.4 LOGICA EN SPELTHEORIE

Een recente ontwikkeling in de logica is het onstaan van verbanden met *speltheorie*. Speltheorie is een tak van de economie waarin wordt onderzocht hoe agents, eventueel op basis van onvolledige informatie, rationele keuzes maken en verantwoorden. In het algemeen gaat het hier om situaties waarin agents uit meerdere acties kunnen kiezen op grond van een 'persoonlijke voorkeur' (in het Engels: *preference relation*, dit lijkt op een 'kleiner dan'–relatie op de verzameling mogelijke acties), die bepaald wordt door de berekende gevolgen (de winst, of uitkomst) van die acties. Hierbij is een probleem dat de 'beste' keuze vaak niet alleen bepaald wordt door de directe gevolgen van die actie, maar ook door de keuzes van andere agents. Een voorbeeld:

Voorbeeld 21.6

Strategisch kiezen
Drie onafscheidelijke vrienden Anna, Bert en Cees moeten kiezen uit twee films voor een avondje uit: Titanic en Harry Potter. Uiteraard kunnen ze ook nog besluiten om niet naar de film te gaan. Ze doen dit

volgens de volgende procedure: eerst kiezen ze met meerderheid van stemmen naar welke film ze gaan (als ze al gaan), daarna kiezen ze met meerderheid van stemmen of ze wel of niet naar die film gaan. De 'winst' van deze verkiezingsprocedure is als het ware 'wel of niet je zin krijgen'. De voorkeur van Anna is: *Titanic* > *niet gaan* > *Harry Potter*, die van Bert: *niet gaan* > *Titanic* > *Harry Potter*, en die van Cees: *Harry Potter* > *Titanic* > *niet gaan*.



Als iedereen volgens zijn eigen voorkeur kiest, wordt in de eerste ronde voor Titanic gekozen, en in de tweede ronde ook voor Titanic (zie de figuur, de vette lijnen geven de meerderheidsuitkomsten weer). Dit vindt Bert niet leuk, want hij ging nog liever *niet* uit dan naar Titanic! Aangezien Bert dit van tevoren aan ziet komen, kiest hij in de eerste ronde niet voor Titanic maar voor Harry Potter. Daarna wordt dan in de tweede ronde 'niet gaan' gekozen. Voor Bert is het dus voordelig om niet te kiezen volgens zijn voorkeur, maar om zogenaamd *strategisch* te kiezen. Echter, Cees voorziet dat Bert strategisch voor Harry Potter gaat kiezen, en besluit daarom in de eerste ronde Titanic te kiezen, dus evenmin volgens zijn persoonlijke voorkeur. Daarmee wordt dan in de tweede ronde Titanic gekozen, niet zijn eerste voorkeur, maar dat deed hij liever dan helemaal niets. Houdt dit nu op?

Nash-evenwicht

Een *strategie* is een vaste speelwijze die een antwoord geeft op elke handeling van de tegenspeler(s). Een *Nash-evenwicht* (*Nash equilibrium*) is een stel strategieën waarbij geen enkele speler er beter van wordt door zijn strategie te veranderen, aangenomen dat de anderen aan hun strategie vasthouden. Er zijn allerlei methoden om zulke evenwichten te bepalen, en zelfs onderling te combineren (zogenaamde gemengde strategieën). Hiermee valt inderdaad een beste oplossing te vinden voor het probleem hiervoor. We geven geen details.

In dit voorbeeld 'kiest ieder voor zich'. In het algemeen wordt het nog ingewikkelder als agents ook nog kunnen samenwerken in *coalities*.

Dit was maar een voorbeeld ter illustratie van de speltheorie als vakgebied, waarbij overigens onmiddelijk duidelijk is dat hier van bekende structuren – bomen – en bekende relaties – preferentierelaties zijn lineaire ordeningen – sprake is. We verwachten dus deze met logische technieken te kunnen beschrijven. Er is inderdaad een snel groeiende en vruchtbare relatie tussen logica en speltheorie. Net als bij relaties met andere vakgebieden, is er ook hier weer sprake van tweerichtingverkeer.

Invloed van speltheorie op logica

Aan de ene kant biedt speltheorie een andere fundering van zaken zoals logische semantiek. De waarheid van een formule in een model kan ook gezien worden als de uitkomst van een *spel* dat tussen twee spelers gespeeld wordt. De ene speler probeert de formule te ontkrachten, en de andere probeert deze te verdedigen. Geldigheid komt dan overeen met het bestaan van een winnende strategie voor de verdediger. We geven weer een voorbeeld:

Voorbeeld 21.7

Speltheoretische semantiek
In hoofdstuk 9 hebben we met semantische tableaus systematisch onderzocht of een gevolgtrekking geldig is. Hierbij werd een tegenmodel geconstrueerd. Op enigszins vergelijkbare wijze kunnen we de geldigheid van een formule in een concreet model bepalen, waarbij we de reductieregels in de vorm gieten van zetten in een *spel* dat tussen twee spelers gespeeld wordt. We noemen de ene speler *Zij* en de andere *Hij*. Dit spel is zogenaamd 'zero sum': als de ene speler wint, verliest de ander. Als *Zij* een winststrategie heeft voor het spel, is de bewering waar, en anders onwaar.
Stel ons concrete model is een blokkenwereld waarin een kleine gele kubus g op een grote rode kubus r ligt, en een grote blauwe kubus b_1 op een kleine blauwe kubus b_2 ligt. Is het waar dat iedere kubus die op een andere ligt, klein of blauw is? Met andere woorden: geldt $\forall x \exists y\ (Op(x, y) \rightarrow (Kl(x) \vee Bl(x)))$? *Zij* beweert van wel, en *Hij* van niet. Omdat de eerste kwantor een universele kwantor is, mag *Hij* eerst een object kiezen in het model – *Zij* moet immers een willekeurig object kunnen verdedigen – en hij kiest de grote blauwe kubus b_1. We krijgen dus $Op(b_1, y) \rightarrow (Kl(b_1) \vee Bl(b_1))$. Omdat de tweede kwantor een existentiële kwantor is, mag *Zij* nu de keuze maken. Zij kiest voor de kleine blauwe kubus. We krijgen nu $Op(b_1, b_2) \rightarrow (Kl(b_1) \vee Bl(b_1))$. De implicatie beschouwen we als een disjunctie $\neg Op(b_1, b_2) \vee (Kl(b_1) \vee Bl(b_1))$. Ook nu mag *Zij* weer kiezen, want een disjunctie is waar als een van beide disjunten waar is. Zij kiest voor het tweede alternatief, blijft aan zet, en van $Kl(b_1) \vee Bl(b_1)$ kiest zij voor $Bl(b_1)$. En deze atomaire bewering is waar. *Zij* wint!

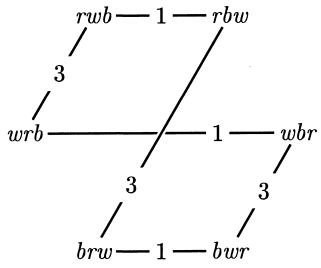
Dit spel kunt u ook echt spelen als een computerspelletje in het programma *Tarski's World* (zie de literatuurlijst), op dit soort modellen met blokken en andere figuren. Zo'n speltheoretisch gefundeerde semantiek blijkt ook weer aanvullende vragen op te roepen, die zelfs tot uitbreidingen van de predikaatlogica leiden. In hoofdstuk 16 over logisch programmeren zagen we dat we kwantorafhankelijkheden in de predikaatlogica expliciet kunnen maken in tweede-orde logica, met Skolem–functies (overigens beschrijft die Skolem–functie precies de winststrategie). In de uitdrukking $\forall x\, \exists y\, \forall z\, \exists w\, \varphi$ hangt y van x af, en w van x en z. In de predikaatlogica kunnen we echter niet uitdrukken dat *zowel* y alleen van x afhangt *als* w alleen van z. In de IF–semantiek (voor *Independence Friendly*) van Hintikka en Sandu kan dit wel. Deze is een generalisering van de speltheoretische semantiek in voorbeeld 21.7. Deze semantiek wordt besproken in het *Handbook of Logic and Language*, zie de literatuurlijst.

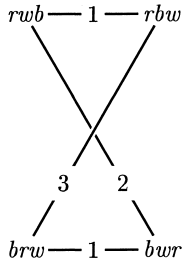
Invloed van logica op speltheorie

Aan de andere kant biedt de logica een formalisering van de speltheorie, zowel van algemene begrippen als van concrete spelen. We geven een voorbeeld van het laatste.

Voorbeeld 21.8

Kaartspelen: het tonen van een kaart
In hoofdstuk 19 werden kaartverdelingen genoemd als concrete multi-agentsystemen met een interessante dynamiek. We kunnen natuurlijk ook kaart*spelen* gaan analyseren, en belanden zo in de speltheorie. Vaak is het doel van het spel om erachter te komen wie een bepaalde kaart heeft, of wat de kaartverdeling is. Voor we de Nash-evenwichten van zulke spelen kunnen berekenen, dienen we de precieze informatie-veranderingen in zetten te beschrijven. De dynamische fijnstructuur van deze handelingen blijkt complex.
De situatie waarin ieder van drie spelers één kaart heeft, is behandeld in voorbeeld 19.5. Stel dat speler 1 aan speler 2 zijn kaart laat zien, terwijl speler 3 'toekijkt' (zie opgave 19.5c). Ook al weet speler 3 niet welke kaart 1 aan 2 laat zien, het zal duidelijk zijn dat 3 nu weet dat 2 de kaartverdeling kent. Met andere woorden: wat de kaartverdeling ook is, 2 kan zich alleen *die* verdeling nog voorstellen: het resultaat is weergegeven in de linkerfiguur hierna. Hierin geldt, dat 3 weet dat 2 de kaart van 1 weet: $K_3(K_2 r_1 \vee K_2 w_1 \vee K_2 b_1)$.





Voorbeeld 21.9

Kaartspelen: het winnen van het spel
De rechterfiguur geeft de informatietoestand weer nadat 1 gezegd heeft dat hij wit niet heeft. Stel nu eens dat het doel van dit spel met drie kaarten is, wie het eerst de kaartverdeling kent. En stel dat de kaartverdeling *rwb* is. We nemen aan dat speler 1, die net zei dat hij wit niet had, daarom nog steeds aan zet is. Hij weet nu nog niet wat de kaartverdeling is. Hij kan immers *rwb* en *rbw* niet van elkaar onderscheiden. Speler 1 laat dus 2 aan zet. Speler 2 weet evenmin wat de kaartverdeling is. Zij kan immers *rwb* en *bwr* niet van elkaar onderscheiden. Maar als zij dit zegt, is dat informatief voor speler 1! Voor speler 1 is immers wereld *rwb* waarin 2 niet kan winnen, nu te onderscheiden van wereld *rbw* waarin 2 wel had kunnen winnen. Maar dat betekent dat 1 nu *wel* kan winnen, en wel precies omdat 2 zei dat ze *niet* kon winnen. Voor speler 3 maakt het in dit geval weinig uit: die kon daarvoor ook al winnen. Als hij dat echter voor zijn beurt publiek had gemaakt, was ook dat voor 1 voldoende geweest om de kaartverdeling te leren. Maar niet voor 2...

Een 'echt' kaartspel waarin dergelijke acties voorkomen is *Cluedo*: hierin worden 21 kaarten over 6 spelers verdeeld. Er zijn drie typen kaarten: verdachten, kamers en wapens. Het doel van het spel is om een moord op te lossen, concreet: wie heeft het gedaan, waarmee, en in welke kamer. Dit zijn kaarten die blindelings getrokken worden en 'op tafel' liggen. Wie het eerst weet wat die kaarten zijn, heeft gewonnen.

De logica heeft ook bijgedragen aan de formalisering van het algemeen begrippenkader van de speltheorie. Zo zijn spelbomen (zoals in voorbeeld 21.6) en hun strategische evenwichten goed te formaliseren. Ook redeneren over coalities en hun macht is in een modale logica fraai beschreven. Verder is er momenteel veel belangstelling voor combinaties van logica en waarschijnlijkheid. Het ligt in de lijn der verwachting dat dit deel–vakgebied zich snel zal blijven ontwikkelen.

21.5 TOT BESLUIT

Hiermee besluiten we dit laatste hoofdstuk. De bedoeling van de vier gegeven voorbeelden is niet om te beweren dat de standaardlogica reeds geheel door de informatica omgevormd zou zijn, of zelfs maar zou moeten worden. Immers, totaal toegeven aan computationele invloeden zou de logica wel eens kunnen beroven van die voordelen van eenvoud en overzichtelijkheid op grond waarvan haar toepassing oorspronkelijk juist begonnen was. Maar we hebben wel willen zeggen dat er geen vaste grenzen bestaan tussen logica en informatica, zoals lezers van dit boek hopelijk ook nog verder zullen ondervinden.

De moderne informatica evolueert nog steeds, en ieder decennium ziet nieuwe thema's ontstaan in deze snel expanderende wetenschap van informatiestructuren en informatieoverdracht. Lag oorspronkelijk de nadruk sterk op rekenmachines op zich en hun programmatuur, thans zijn veel bredere onderwerpen centraal komen te staan. Voorbeelden uit de afgelopen jaren zijn gedistribueerd rekenen door groepen van agents, met het internet als spectaculairste voorbeeld, beeldverwerking en multimedia, en mens–machine interactie. Praktische en fundamentele vragen die daarbij rijzen, betreffen nieuwe verschijnselen als informatie en interactie in groepen processoren, beveiliging van gegevens en communicatie, of het temmen van grootschalige statistische (over-)informatie. Deze vragen lopen over van de traditionele kerninformatica naar de kunstmatige intelligentie, en zelfs soms naar de cognitieve psychologie. In het bijzonder leiden ze ook tot nieuwe contacten met de logica, zoals enkele voorbeelden van kennis en communicatie in ons boek al hebben aangegeven. Bovendien maakt de snel groeiende rekenkracht van moderne computers het steeds beter mogelijk om logische bewijs- en testmethoden rechtstreeks computationeel uit te voeren, zodat ook een grensgebied is ontstaan van computationele experimentatie met vele soorten logische talen en calculi, waardoor realistische 'bewijsmachines' mogelijk worden. Ook deze zogenaamde 'computationele logica' is inmiddels een vakgebied op zich.

Appendices

Algemene begrippen

Appendix 1

Algemene begrippen

1 VERZAMELINGEN

In het vervolg zijn A en B verzamelingen.

A en B zijn *gelijk* ($A = B$) als voor alle x: $x \in A \Leftrightarrow x \in B$.
A is een *deelverzameling* van B ($A \subseteq B$) als voor alle x: $x \in A \Rightarrow x \in B$.
A is een *echte deelverzameling* van B ($A \subset B$) als: $A \subseteq B$ en er is een $x \in B$ met $x \notin A$.
De *machtsverzameling* $P(A)$ van A is $\{ x \mid x \subseteq A\}$.
De *lege verzameling* $\varnothing$ is de verzameling die geen elementen bevat.
De *doorsnede* $A \cap B$ is $\{x \mid x \in A$ en $x \in B\}$.
De *vereniging* $A \cup B$ is $\{x \mid x \in A$ of $x \in B\}$.
Het *verschil* $A \setminus B$ is $\{x \mid x \in A$ en $x \notin B\}$.
Het *complement* A^c is $\{x \mid x \notin A\}$.

In het algemeen beschouwen we het complement van een verzameling A ten opzichte van een grotere verzameling U. In dat geval geldt $A^c = U \setminus A$.

2 RELATIES

Een *geordend paar* (a, b) is een geordend rijtje van twee objecten.
Het *cartesisch product* $A \times B$ van twee verzamelingen A en B is $\{(a, b) \mid a \in A$ en $b \in B\}$.
Een deelverzameling R van het cartesisch product $A \times B$ heet een *relatie* R 'van A naar B' of 'op $A \times B$'. Een relatie is dus een verzameling geordende paren. Als $R \subseteq A \times A$, dan heet R een relatie op A.
Het *domein* van een relatie R is $\{x \mid$ er is een y zodat $(x, y) \in R\}$.
Het *beeld* van een relatie R is $\{y \mid$ er is een x zodat $(x, y) \in R\}$. In plaats van $(x, y) \in R$ schrijven we ook wel $R(x, y)$ of Rxy of xRy.
Omdat relaties verzamelingen zijn, kunnen ook op relaties operaties toegepast worden, zoals $\cap$, $\cup$ en ^{c}.
De *converse* $R^\smile$ van R is $\{(x, y) \mid (y, x) \in R\}$.

De *compositie* $R_1 \circ R_2$ van R_1 en R_2 is $\{(x, y) \mid$ er is een z zodat $(x, z) \in R_1$ en $(z, y) \in R_2\}$.

Mogelijke eigenschappen van tweeplaatsige relaties op een verzameling A (voor alle R, x, y, z):

R is *transitief* als: $(x, y) \in R$ en $(y, z) \in R \Rightarrow (x, z) \in R$.
R is *reflexief* als: voor alle $x \in A$, $(x, x) \in R$.
R is *symmetrisch* als: $(x, y) \in R \Rightarrow (y, x) \in R$.
R is *lineair* als: $(x, y) \in R$ of $(y, x) \in R$ of $x = y$.
R is *antisymmetrisch* als: $(x, y) \in R$ en $(y, x) \in R \Rightarrow x = y$.
R is *irreflexief* als: voor alle $x \in A$, $(x, x) \notin R$.
R is *asymmetrisch* als: $(x, y) \in R \Rightarrow (y, x) \notin R$.
R is *dicht* als: $(x, y) \in R \Rightarrow$ er is een w zodat $(x, w) \in R$ en $(w, y) \in R$.
R is een *lineaire orde* als R transitief, irreflexief en lineair is.
R is een *partiële orde* als R transitief, reflexief en antisymmetrisch is.
R is een *equivalentierelatie* als R transitief, reflexief en symmetrisch is.

Equivalentierelaties verdelen het domein in zogenaamde *equivalentie-klassen*. Elke equivalentieklasse van een relatie R bestaat uit de objecten die door R met elkaar verbonden zijn.

Met behulp van geordende paren kunnen ook langere geordende rijtjes gemaakt worden. Bijvoorbeeld, als (x, y) een geordend paar is en z één of ander object, dan is $((x, y), z)$ ook een geordend paar. Het geordend paar $((x, y), z)$ noteren we ook wel als het geordend *drietal* (x, y, z). Een *n–plaatsige relatie* R is een verzameling geordende n–tallen (met n vast). Notatie: $(x_1, ..., x_n) \in R$ of $R(x_1, ..., x_n)$ of $Rx_1 ... x_n$.

3 FUNCTIES

Een *n–plaatsige functie* f is een $(n + 1)$–plaatsige relatie zodat voor alle $x_1, x_2, ..., x_n$ en y: als $(x_1, ..., x_n, y) \in f$, dan is y uniek.
Notatie: $f(x_1, ..., x_n) = y$.

In het vervolg beschouwen we eenplaatsige functies.
Als $f \subseteq A \times B$, dan heet f een functie *van* A *naar* B, notatie $f: A \rightarrow B$.
De functieruimte B^A is $\{f \mid f: A \rightarrow B\}$.
Het *beeld* van f noteren we ook als $f[A]$.
Het *origineel* van f, notatie $f^{-1}[B]$, is $\{x \mid$ er is een $y \in B$ zodat $f(x) = y\}$.
Een functie f is *partieel* als $f^{-1}[B] \subset A$.
Een functie f is *surjectief* als $f[A] = B$.

Een functie f is *injectief* als: $f(x) = f(y) \Rightarrow x = y$.
Een functie f is *bijectief* als f surjectief en injectief is.

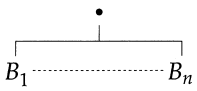
Deze begrippen zijn te generaliseren naar meerplaatsige functies.

4 BOMEN

Een *boom* is een abstracte grafische vorm. We geven een informele definitie:
De volgende vorm is een boom:

•

Als $B_1, ..., B_n$ bomen zijn, dan ook:



Hierbij kan n zowel eindig als oneindig zijn. Ook kunnen we de constructiestap oneindig vaak toepassen.
Elk symbool • in een boom heet een *knoop*. Een boom bevat één knoop die geen knopen 'boven' zich heeft. Deze knoop heet de *wortel*. Een knoop die geen knopen 'onder' zich heeft, heet een *blad* of *eindknoop*. Een 'pad' van de wortel naar een blad heet een *tak*. De *lengte* van een tak is het aantal knopen op die tak minus 1. Een boom heet *eindig* als elke tak een eindige lengte heeft. Een boom heet *oneindig* als er een tak met oneindige lengte is. Een *eindig vertakkende* boom is een boom die bij iedere knoop in *eindig* veel paden splitst. Een *deelboom* is een deel van een boom dat zelf ook boom is.

5 WISKUNDIG BEWIJZEN

Notatie
Als B volgt uit A, dan noteren we dat ook wel als $A \Rightarrow B$. Als B uit A volgt én A uit B volgt, dan zijn A en B equivalent; dit noteren we wel als $A \Leftrightarrow B$, en ook wel als 'A desda B' (desda is een afkorting van 'dan en slechts dan als'). Het bewijs voor een bewering sluiten we vaak af met het teken □. Dit staat voor 'hiermee is het gevraagde bewezen'.

Keten van gevolgtrekkingen
Als B volgt uit A en C volgt uit B, dan volgt C uit A.

Contrapositie
Als (niet A) volgt uit (niet B), dan volgt B uit A.

Bewijs uit het ongerijmde
Gegeven is A. Stel B geldt niet. Als uit beide volgt dat A niet geldt, dan is dat in tegenspraak (*ongerijmd*) met het gegeven A. Dus B geldt wel. Dus volgt B uit A.

Natuurlijke inductie
Als een bewering over natuurlijke getallen geldt voor 0, en als we bovendien uit de aanname dat de bewering geldt voor een willekeurig getal n, kunnen afleiden dat deze ook geldt voor het getal $n + 1$, dan mogen we concluderen dat de bewering geldt voor alle natuurlijke getallen.

Appendix 2

Historisch overzicht

Dit boek is geschreven zonder expliciete naamsvermeldingen en 'credits'. Hier volgen alsnog enkele namen en data die belangrijke bijdragen in de geschiedenis van het vak markeren. Het toenemen van de naamsdichtheid in dit overzicht naarmate de geschiedenis vordert wil overigens geenszins zeggen dat grote geleerden tegenwoordig steeds ruimer voorhanden zijn, maar slechts dat we voor het meer recente verleden nog niet weten hoe te schiften.

1 De geschiedenis van de logica als wetenschap begint in de vierde eeuw voor Christus, met Aristoteles als uitvinder van de syllogistiek, en de school der Stoa als oorsprong van de propositielogica.
Een volgende bloeitijd is pas in de Middeleeuwen, op het gebied van de filosofisch toegepaste logica. Deze zogenaamde Scholastiek werd voornamelijk onderwezen aan de universiteiten van Oxford en Parijs. Een belangrijke wegbereider voor de meer wiskundige vakoriëntatie van de moderne tijd is rond 1700 Gottfried Wilhelm Leibniz, met zijn project voor een alomvattende formele taal (de 'characteristica universalis') en een daarbij behorende algoritmische test voor geldig redeneren 'calculus ratiocinator'.

2 De propositielogica in haar moderne vorm is bedacht door George Boole (1847), het predikaatlogische formalisme door Gottlob Frege (1879), met verwante ideeën van Charles Peirce (1880). De semantiek van de predikaatlogica gaat terug op Alfred Tarski (1933), die ook later daaruit de logische modeltheorie zou initiëren. Semantische tableaus werden ingevoerd door Evert Beth (1955). Natuurlijke deductie, en daarmee het begin van de logische bewijstheorie, werd gecreëerd door Gerhard Gentzen (1934). De belangrijkste impuls tot een significante ontwikkeling van logische metatheorie gaf Kurt Gödel (1930) met zijn volledigheidsstelling.

3 Equationele logica ontstond in het werk van Garrett Birkhoff (1935), lambda–calculus in dat van Alonzo Church (1941) met wortels in de

typentheorie van Bertrand Russell (1908), intuïtionistische logica in dat van Arend Heyting (1929). Modale logica in de stijl van dit boek werd onder meer ontwikkeld door Rudolf Carnap (1947) en Saul Kripke (1959). Speciale onderwerpen kennen hun eigen pioniers, zoals tijdslogica (Arthur Prior (1957)), kennislogica (Jaako Hintikka (1962)), dynamische logica (Vaughan Pratt (1976)).

4 De klassieke analyse van effectieve berekenbaarheid stamt van Alan Turing (1936), met verdere belangrijke bijdragen van Stephen Kleene (1936), de schepper van de recursietheorie, en Emil Post (1944). Operationele semantiek voor imperatieve programma's in onze zin is afkomstig van Tony Hoare (1969). De verbreiding van een bijbehorende programmeerdiscipline heeft meer te danken aan Edsger Dijkstra (1976). Een belangrijk 'denotationeel' alternatief loopt via modellen voor de lambda–calculus, te danken aan Dana Scott (1969). De ontwikkeling van stellingbewijzen berust op logische ideeën van Jacques Herbrand (1928) en Thoralf Skolem (1920), terwijl de beslissende aanzet voor de resolutiemethode stamt van J. A. Robinson (1965). Logisch programmeren als tak van informatica dankt veel aan Robert Kowalski (1974).

5 Temporele logica werd binnen de informatica geïntroduceerd door Amir Pnueli (1976), en binnen de kunstmatige intelligentie onder meer door James Allen en Patrick Hayes (1983). Originele computationele toepassingen van kennislogica vinden we bij Joseph Halpern (1984) en Yoav Shoham (1988). Pioniers op het gebied van niet–standaard redeneren in de informatica zijn onder meer Ray Reiter (1980), John McCarthy (1980) en Dov Gabbay (1984). De meest invloedrijke logicus op de grens van de taalkunde is Richard Montague (1970) geweest. Een bekende hedendaagse figuur op dit gebied is Hans Kamp (1984). Grote namen van de categoriale grammatica zijn Kazimierz Ajdukiewicz (1935) en Joachim Lambek (1958).

6 Enkele belangrijke gangmakers in een mogelijke 'computationele wending' binnen de logica zelf zijn Jon Barwise (1985), Jean–Yves Girard (1987) – bekende voortrekker van de voorkomenslogica, Yuri Gurevich (1988) en Peter Gärdenfors (1989). Het hier gepresenteerde systeem van dynamische interpretatie komt van Jeroen Groenendijk en Martin Stokhof (1990).

Nog recenter onderzoek is ten dele behandeld in de gereviseerde hoofdstukken, en wordt genoemd in het literatuuroverzicht hierna.

Appendix 3

Literatuurlijst

Om te beginnen geven we enkele aanbevolen (andere) leerboeken. Daarna een lange lijst referenties over de speciale onderwerpen die zoal in het boek aan bod zijn gekomen. Tot slot een lijst van toonaangevende tijdschriften op het gebied van de logica.

Aanbevolen leerboeken

- Logica meer voor wiskundigen

Bell, J.L. en M. Machover, *A Course in Mathematical Logic*, North–Holland, Amsterdam, 1977.

- Logica meer voor taalkundigen

Gamut, L.T.F., *Logica, Taal en Betekenis*, Het Spectrum, Utrecht/Antwerpen, 1982 (2 delen).

Gamut, L.T.F., *Logic, Language & Meaning*, Chicago University Press, Chicago, 1991 (2 delen).

Barwise, J. en J. Etchemendy, *Language, Proof and Logic* (bevat Tarski's World), CSLI Publications, Stanford, 2000.

- Logica meer voor informatici

Huth, M. en M. Ryan, *Logic for Computer Science*, Cambridge University Press, Cambridge, 2000.

- Overzichtswerken

Barwise, J. (redacteur), *Handbook of Mathematical Logic*, North–Holland, Amsterdam, 1977.

Gabbay, D. en F. Guenthner (redacteuren), *Handbook of Philosophical Logic*, Reidel, Dordrecht, 1983–1989 (4 delen).

Gabbay, D. en anderen (redacteuren), *Handbook of Logic in Logic Programming and Artificial Intelligence*, Oxford University Press, Oxford, vanaf 1993 (vijf delen).

Gabbay, D. en anderen (redacteuren), *Handbook of Logic in Computer Science*, Oxford University Press, Oxford, vanaf 1992 (6 delen).

Speciale onderwerpen

- Intuïtionisme

Troelstra, A. en D. van Dalen, *Constructivism in Mathematics*, deel I en II, North–Holland, Amsterdam, 1989.

- Lambda–calculus

Barendregt, H.P., *The Lambda Calculus. Its Syntax and Semantics*, North–Holland, Amsterdam, 1984 (tweede editie).

- Theoretische informatica

Abelson, H. en G.J. Sussman, *Structure and Interpretation of Computer Programs*, MIT Press, Cambridge MA, 1985.

Bakker, J. de, *A Mathematical Theory of Program Correctness*, Prentice Hall International, Englewood Cliffs NJ, 1980.

Papadimitriou, Ch., *Computational Complexity*, Addison-Wesley, Reading MA, 1994.

Kaldeway, A., *Programming, the derivation of algorithms*, Prentice Hall, Londen, 1990.

Harel, D., *Algorithmics, the spirit of computing*, Addison-Wesley, Reading MA, 1992.

Leeuwen, J. van en anderen (redacteuren), *Handbook of Theoretical Computer Science*, Elsevier Science Publishers, Amsterdam, 1990.

- Logisch programmeren

Bratko, I., *Prolog Programming for Artificial Intelligence*, Addison Wesley, Wokingham, 1986.

Lloyd, J.W., *Foundations of Logic Programming*, Springer, Berlijn, 1987 (tweede editie).

- Temporele logica

Benthem, J.F.A.K. van, *The Logic of Time*, Reidel, Dordrecht, 1991 (tweede editie).

Shoham, Y., *Reasoning about Change. Time and Causation from the Standpoint of Artificial Intelligence*, MIT Press, Cambridge MA, 1988.

- Recursietheorie en berekenbaarheid:

Rogers, H., *Theory of Recursive Functions and Effective Computability*, McGraw–Hill, New York, 1967.

- Bewijstheorie

Troelstra, A.S., *Lectures on Linear Logic*, CSLI Lecture Notes 29, CSLI Publications, Stanford, 1992.

Troelstra, A.S., en H. Schwichtenberg. *Basic Proof Theory*, Cambridge University Press, 2000 (tweede editie).

- Inductieve definities

Moschovakis, Y.N., *Elementary Induction on Abstract Structures*, North–Holland, Amsterdam, 1974.

- Modale logica

Hughes, G.E. en M.J. Cresswell, *A Companion to Modal Logic*, Methuen, Londen, 1984.

Blackburn, P., M. de Rijke en Y. Venema, *Modal Logic*, Cambridge University Press, Cambridge, 2001.

- Dynamische logica

Harel, D., *First Order Dynamic Logic*, Lecture Notes in Computer Science, Volume 68, Springer, Berlijn, 1979.

Goldblatt, R., *Logics of Time and Computation*, CSLI Lecture Notes No. 7, Stanford, 1992 (tweede editie).

Harel, D., D. Kozen en J. Tiuryn, *Dynamic Logic*, MIT Press, Cambridge MA, 2000.

- Intensionele logica en informatica

Benthem, J.F.A.K. van, *A Manual of Intensional Logic*, CSLI Lecture Notes No. 1, Stanford, 1988 (tweede editie).

- Kennislogica en multi-agentsystemen

Fagin, R. en anderen, *Reasoning about Knowledge*, MIT Press, Cambridge MA, 1995.

Meyer, J.-J.Ch. en W. van der Hoek, *Epistemic Logic for AI and Computer Science*, Cambridge University Press, Cambridge, 1995.

Wooldridge, M., *An introduction to MultiAgent Systems*, Wiley, Chichester, 2002.

- Niet–standaard redeneren

Gardenfors, P., *Knowledge in Flux. Modelling the Dynamics of Epistemic States*, MIT Press, Cambridge MA, 1988.

- Modeltheorie

Chang, C. en H. Keisler, *Model Theory*, North–Holland, Amsterdam, 1973.

Hodges, W., *A Shorter Model Theory*, Cambridge University Press, Cambridge, 1997.

- Logica en natuurlijke taal

Benthem, J.F.A.K. van, *Essays in Logical Semantics*, Reidel, Dordrecht, 1986.

Moortgat, M., *Categorial Investigations. Logical and Linguistic Aspects of the Lambek Calculus*, Foris, Dordrecht, 1988.

Benthem, J.F.A.K. van, en A. ter Meulen (redacteuren), *Handbook of Logic and Language*, Elsevier Science Publishers, Amsterdam, 1997.

- Geschiedenis van de logica

Kneale, W. en M. Kneale, *The Development of Logic*, Clarendon, Londen, 1962.

Bochenski, I.M., *Ancient Formal Logic*, North–Holland, 1968.

Tijdschriften

- Logica

The Journal of Symbolic Logic, gepubliceerd door the Association for Symbolic Logic, Baltimore.

The Journal of Philosophical Logic, Kluwer, Dordrecht.

Studia Logica, Kluwer, Dordrecht.

- Logica en wiskunde

Fundamenta Mathematica, gepubliceerd door PWN, Warschau.

Annals of Pure and Applied Logic, North–Holland, Amsterdam.

- Logica en taalkunde

Linguistics and Philosophy, Reidel, Dordrecht.

Computational Linguistics, MIT Press, Cambridge MA.

- Logica en informatica

Theoretical Computer Science, Elsevier Science, Amsterdam.

Logic and Computation, Oxford University Press, Oxford.

Artificial Intelligence, North–Holland, Amsterdam.

Journal of Logic, Language en Information, Kluwer, Dordrecht.