Formeel Denken

Herman Geuvers
Deels gebaseerd op het herfst 2002 dictaat van Henk Barendregt en Bas Spitters,
met dank aan het Discrete Wiskunde dictaat van Wim Gielen.
Herfst 2008 herzien en uitgebreid door Engelbert Hubbers en Freek Wiedijk.

9 juli 2013

Inhoudsopgave

Referenties

[1] W. Gielen. Discrete wiskunde voor informatici. Katholieke Universiteit Nijmegen - Subfaculteit Wiskunde 2002.

[2] M. Huth and M. Ryan. *Logic in Computer Science*. Cambridge University Press, 2004. ISBN 0-521-54310-X.

[3] J.F.A.K. van Benthem, H.P. van Ditmarsch, J.S. Lodder, J. Ketting, and W.P.M. Meyer-Viol. *Logica voor informatici*. Pearson Addison-Wesley, Nederland, 2003. ISBN 90-430-0722-6 (dit boek wordt ook gebruikt in het college Beweren en Bewijzen).

[4] R.J. Wilson. *Introduction to Graph Theory*. Pearson Education Limited, fifth edition, 2010. ISBN 978-0-273-72889.

1 Propositielogica

In dit blok behandelen we de propositielogica, ook wel uitspraakrekening genoemd.

1.1 Formele taal en natuurlijke taal

Natuurlijke talen (Nederlands, Engels, Duits,...) zijn niet erg precies. Kijk bijvoorbeeld maar naar de volgende voorbeelden:

- Socrates is een mens. Een mens is sterfelijk. Dus Socrates is sterfelijk.
- Ik ben iemand. Iemand schilderde de Mona Lisa. Dus ik ben de schilder van de Mona Lisa.

De eerste redenering klopt, de tweede niet, maar heeft wel een soortgelijke vorm.

Is de zin

Deze zin is niet waar.

nu waar of niet?

Om dit soort problemen te voorkomen gebruiken we een formele taal. Dit is een soort laboratoriummodel voor de natuurlijke taal. We zullen zien dat we met een heel eenvoudige kunsttaal, toch behoorlijk wat uitspraken en redeneringen heel precies kunnen maken. Dit is bijvoorbeeld erg belangrijk voor het specificeren van programma's; over de specificatie mag immers geen misverstand bestaan.

We zullen nu laten zien hoe we van het Nederlands naar een formele taal gaan. Om te redeneren combineren we vaak een aantal eenvoudige uitspraken, bijvoorbeeld: '*als* het regent *en* ik ben buiten, *dan* word ik nat'. Deze uitspraak is opgebouwd uit de eenvoudige uitspraken 'het regent', 'ik ben buiten' en 'ik word nat'.

1.2 Woordenboek

We kunnen dit ook formeler opschrijven met behulp van een klein woordenboek.

R	het regent
Z	de zon schijnt
RB	er is een regenboog
N	ik word nat
D	ik blijf droog
Bui	ik ben buiten
Bin	ik ben binnen

De zin hierboven wordt dan 'als R en Bui, dan N'. Net zo, kunnen we de volgende zinnen maken: 'als RB, dan Z', 'Z en RB', 'als R en Bin, dan D'.

1.3 Verbindingswoorden

We kunnen ook de verbindingswoorden formeel opschrijven.

Dat doen we op de volgende manier:

Formele taal	Nederlands
$f \wedge g$	f en g
$f \vee g$	f of g
$f \to g$	als f, dan g
$f \leftrightarrow g$	f dan en slechts dan als g[1]
$\neg f$	niet f

De zinnetjes hierboven worden dan 'RB→Z', 'Z∧RB' en '(R∧Bin)→D'.

Nederlands	semi-formeel	Formeel
als het regent *en* ik ben buiten, *dan* word ik nat	als R en Bui, dan N	$(R \wedge \text{Bui}) \to N$
als er een regenboog is, *dan* schijnt de zon	als RB, dan Z	$RB \to Z$
ik ben binnen *of* buiten	Bin of Bui	$\text{Bin} \vee \text{Bui}$

Opgave 1.A Vind zinnen uit de formele taal die hetzelfde betekenen als de volgende zinnen.

(i) Het regent niet, noch schijnt de zon.
(ii) De zon schijnt, tenzij het regent
(iii) Of de zon schijnt, of het regent. (We bedoelen dus niet allebei)
(iv) Er is alleen een regenboog als de zon schijnt en het regent.
(v) Als ik buiten ben, dan word ik nat, mits het regent.

We denken bij de voegtekens (zoals $\vee, \wedge$) natuurlijk aan de bekende Nederlandse verbindingswoorden. Ook 'of' is zo'n voegwoord, dat van twee beweringen een nieuwe bewering maakt. De betekenis van 'of' is in ons taalgebruik echter een beetje onduidelijk: sommige mensen vinden de bewering '$1 + 1 = 2$ of $2 + 3 = 5$' waar, en andere mensen onwaar. Wat vind jij ervan? Als je erover nadenkt merk je, dat de betekenis van het woordje 'of' in ons taalgebruik dubbelzinnig is. Maar studenten Informatiekunde en Kunstmatige Intelligentie houden niet van dubbelzinnige woordjes, dus wij zullen van de twee gangbare betekenissen van 'of' er één kiezen: we spreken af dat 'A of B' ook waar is in het geval A en B beide waar zijn. In definitie 1.5 wordt dit later nog eens expliciet gemaakt.

Opgave 1.B Kun je $f \leftrightarrow g$ ook uitdrukken met behulp van de andere symbolen?

Opgave 1.C Vertaal de volgende zinnen uit de formele taal naar het Nederlands:

(i) $R \leftrightarrow Z$
(ii) $\text{RB} \to (R \wedge Z)$
(iii) $\text{Bui} \to \neg\,\text{Bin}$
(iv) $\text{Bui} \vee \text{Bin}$

Definitie 1.1 De taal van de uitspraakrekening (of propositielogica) is de volgende formele taal. Zij A een oneindige verzameling van *atomen*:

$$A := \{a, b, c, d, a_1, a_2, a_3, \ldots\}$$

Zij V de verzameling van *voegtekens*:

$$V := \{\wedge, \vee, \to, \leftrightarrow, \neg\}$$

[1] De tekst "dan en slechts dan als" wordt vaak afgekort tot "desda".

Zij H de verzameling van *haakjes*:

$$H := \{(,)\}$$

Dan is het *alfabet* de verzameling $\Sigma := A \cup V \cup H$, waarbij $\cup$ voor de vereniging van twee verzamelingen staat.

Hiermee kunnen *woorden* uit deze taal als volgt worden opgebouwd:

1. Een atoom is een woord.
2. Als f en g woorden zijn, dan zijn $(f \wedge g), (f \vee g), (f \to g), (f \leftrightarrow g)$ en $\neg f$ woorden.
3. Alle woorden worden op deze manier gevormd.

De woorden van deze taal noemen we *proposities*.

Conventie 1.2 Meestal laten we de buitenste haakjes weg, dus we schrijven bijvoorbeeld $a \wedge b$ in plaats van $(a \wedge b)$. De binnenste haakjes mogen we natuurlijk niet weg laten: vergelijk $(R \wedge Z) \to RB$ en $R \wedge (Z \to RB)$. We kunnen wel een notatie *afspreken* waarbij we *sommige* haakjes weglaten en dat doen we dan ook door middel van de volgende afspraak waarin de verschillende *prioriteiten* worden vastgelegd:

- $\neg$ bindt sterker dan $\wedge$
- $\wedge$ bindt sterker dan $\vee$
- $\vee$ bindt sterker dan $\to$
- $\to$ bindt sterker dan $\leftrightarrow$

Dit betekent dat we $\mathrm{Bin} \vee \mathrm{RB} \to \mathrm{Bui} \leftrightarrow \neg\,\mathrm{Z} \wedge \mathrm{R}$ moeten lezen als $((\mathrm{Bin} \vee \mathrm{RB}) \to \mathrm{Bui}) \leftrightarrow (\neg\,\mathrm{Z} \wedge \mathrm{R})$.

Deze conventie alleen is nog niet duidelijk genoeg: zij beschrijft alleen hoe de impliciete haakjes staan als er *verschillende* voegtekens worden gebruikt, maar niet waar de haakjes staan als *dezelfde* voegtekens gecombineerd worden. Moeten we $\mathrm{Bui} \to \mathrm{R} \to \mathrm{N}$ nu als $(\mathrm{Bui} \to \mathrm{R}) \to \mathrm{N}$ of als $\mathrm{Bui} \to (\mathrm{R} \to \mathrm{N})$ lezen? Dit wordt bepaald door de *associativiteit* van de operatoren.

Conventie 1.3 De voegtekens $\wedge$, $\vee$, $\to$ en $\leftrightarrow$ zijn *rechts associatief*. Dat wil zeggen dat als $v \in \{\wedge, \vee, \to, \leftrightarrow\}$, dan moet

$$A \;\; v \;\; B \;\; v \;\; C$$

gelezen worden als

$$A \;\; v \;\; (B \;\; v \;\; C)$$

Merk op dat dit een keuze is die bij dit vak zo gemaakt is. In andere logicaboeken kan een andere keuze worden gemaakt.

Opmerking 1.4 (Deze opmerking heeft pas betekenis na blok 3, "Talen en automaten"; bekijk hem dan nog eens goed.) De formele definitie met behulp van een *contextvrije grammatica* is als volgt: $\Sigma = A \cup V \cup H$, ofwel

$$\Sigma = \{a, b, c, \ldots, \wedge, \vee, \to, \leftrightarrow, \neg, (,)\}.$$

$$S \;\to\; a \;\mid\; \neg S \;\mid\; (S \wedge S) \;\mid\; (S \vee S) \;\mid\; (S \to S) \;\mid\; (S \leftrightarrow S)$$

Hier mag a ieder element van A zijn.

1.4 Betekenis en waarheidstabellen

De zin 'als a en b, dan a' is waar, wat je ook voor a en b invult. We zouden nu ook willen zeggen dat de zin '$a \wedge b \to a$' waar is[2], maar we hebben nog niet gedefinieerd wat dat betekent: '$a \wedge b \to a$' is zomaar een zin in een taal. We gaan nu definiëren wat de betekenis van een zin is, in het bijzonder, wanneer een zin in de taal van de logica *waar* is.

Voor de atomen kunnen we aan eenvoudige uitspraken denken zoals '2=3', 'Jolly Jumper is een paard' of 'het regent'. In de klassieke logica, waar we ons in dit college toe beperken, nemen we aan dat alle atomen of waar zijn of niet waar zijn. We maken ons geen zorgen om het feit dat we soms niet weten of de zin waar of niet waar is, zoals bij de zin, 'op 1 januari 2050 regent het in Nijmegen'.

De waarheid van de atomen wordt bepaald door hun interpretatie in een *model*. Bijvoorbeeld '2=3' is niet waar in het model van de natuurlijke getallen. 'Jolly Jumper is een paard' is waar in het model van het stripboek Lucky Luke. De zin 'het regent' was niet waar in Nijmegen op 17 september 2002.

Bekijk nu eens de zin '$a \wedge b$'. We willen dat deze zin alleen waar is in een model als zowel a als b waar zijn in dat model. Als we nu een lijstje maken met alle mogelijke waarden voor a en voor b, dan kunnen we ook bepalen wat de mogelijke waarden zijn voor $a \wedge b$.

In de informatica schrijven we vaak 1 voor *waar* en 0 voor *onwaar*. De logische operaties zijn de elementaire operaties op bits.

We krijgen dan de volgende waarheidstabel:

x	y	$x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

'Als ..., dan ...' is eveneens een methode om van twee beweringen een nieuwe bewering te maken. Ook hier geeft ons gangbare taalgebruik geen duidelijk antwoord op de vraag, wanneer 'als A, dan B' waar is. Laten we bijvoorbeeld eens kijken naar het geval, dat A onwaar is en ook B onwaar is. Is 'als A, dan B' in dat geval waar? Wat vind je ervan? Denk bijvoorbeeld eens aan:

'als $1 + 1 = 3$, dan $2 + 2 = 6$'
'als ik van het Erasmusgebouw spring, dan verander ik in een vogeltje'
'als ik hier iets van snap, dan heet ik Alpje'

We zullen aan alle onduidelijkheid een einde maken door een waarheidstabel *af te spreken* (we schrijven '$A \to B$' in plaats van 'als A, dan B'). Dit staat allemaal in de volgende definitie.

Definitie 1.5 De *waarheidstabellen* voor de logische voegtekens zijn als volgt

x	$\neg x$
0	1
1	0

x	y	$x \wedge y$
0	0	0
0	1	0
1	0	0
1	1	1

x	y	$x \vee y$
0	0	0
0	1	1
1	0	1
1	1	1

[2]Let op: $a \wedge b \to a$ moeten we dus lezen als $(a \wedge b) \to a$

x	y	$x \to y$
0	0	1
0	1	1
1	0	0
1	1	1

x	y	$x \leftrightarrow y$
0	0	1
0	1	0
1	0	0
1	1	1

Met behulp van de waarheidstabellen kunnen we de waarde van een complexe propositie uitrekenen als we de waarde van de atomen kennen. Zo kunnen we ook voor een complexe propositie een waarheidstabel maken die alle mogelijkheden weergeeft.

Voorbeeld 1.6 De waarheidstabel van $a \vee b \to a$.

a	b	$a \vee b$	a	$(a \vee b) \to a$
0	0	0	0	1
0	1	1	0	0
1	0	1	1	1
1	1	1	1	1

Opgave 1.D Schrijf de waarheidstabellen op van:

(i) $a \vee \neg a$
(ii) $(a \to b) \to a$
(iii) $a \to (b \to a)$
(iv) $a \wedge b \to a$
(v) $a \wedge (b \to a)$
(vi) $\neg a \to \neg b$

1.5 Modellen en waarheid

In de introductie spraken we over *modellen* en waarheid in een model. In de waarheidstabellen hebben we gezien dat de '*waarheid*' van een propositie volledig bepaald wordt door de waarden die we aan de atomen toekennen. Een model in de propositielogica is dan ook eenvoudigweg een toekenning van waarden ($\{0, 1\}$) aan de atomen.

Definitie 1.7 Een *model* in de propositielogica is een *waardentoekenning* of *valuatie* van de atomen: een functie $v : A \to \{0, 1\}$.

Om de waarde van een propositie f te bepalen hoeven we niet de waarden van alle atomen te weten, maar alleen van de atomen die in f voorkomen. We zullen daarom een model vaak gelijkstellen aan een *eindige waardentoekenning*.

Voorbeeld 1.8 In het model v met $v(a) = 0$ en $v(b) = 1$ heeft de propositie $a \vee b \to a$ de waarde 0.

Conventie 1.9 Als v een model is, dan schrijven we ook wel $v(f)$ voor de waarde van f. We zeggen f is *waar in een model* v als $v(f) = 1$.

Dus $a \vee b \to a$ is niet waar in v als $v(a) = 0$ en $v(b) = 1$, maar $a \to (b \to a)$ is wel waar in zo'n model.

Definitie 1.10 Als een propositie f waar is in ieder model (dat wil zeggen dat er in de waarheidstabel van f alleen maar 1-en staan), dan noemen we de propositie *logisch waar*. Notatie $\models f$. Een logisch ware propositie heet ook wel een *tautologie*.

Als een propositie f *niet* logisch waar is, kan dat genoteerd worden met $\not\models f$.

Opgave 1.E Welke van de volgende proposities zijn logisch waar?

(i) $a \vee \neg a$
(ii) $a \to (a \to a)$
(iii) $a \to a$
(iv) $(a \to b) \to a$
(v) $a \to (b \to a)$
(vi) $a \wedge b \to a$
(vii) $a \vee (b \to a)$
(viii) $a \vee b \to a$

Opgave 1.F Laat f en g proposities zijn. Ga na of de volgende uitspraken kloppen. Verklaar je antwoorden.
(i) Als $\models f$ en $\models g$, dan $\models f \wedge g$.
(ii) Als niet $\models f$, dan $\models \neg f$.
(iii) Als $\models f$ of $\models g$, dan $\models f \vee g$.
(iv) Als (als $\models f$, dan $\models g$), dan $\models f \to g$.
(v) Als $\models \neg f$, dan niet $\models f$.
(vi) Als $\models f \vee g$, dan $\models f$ of $\models g$.
(vii) Als $\models f \to g$, dan (als $\models f$, dan $\models g$).
(viii) Als $\models f \leftrightarrow g$, dan ($\models f$ dan en slechts dan als $\models g$).
(ix) Als ($\models f$ dan en slechts dan als $\models g$), dan $\models f \leftrightarrow g$.

1.6 Logisch equivalent

Definitie 1.11 Twee proposities zijn *logisch equivalent* als f waar is in een model, dan en slechts dan als g waar is in dat model.
Iets preciezer geformuleerd: f en g zijn logisch equivalent als voor ieder model v geldt dat $v(f) = 1$ dan en slechts dan als $v(g) = 1$. Dat wil zeggen dat f en g dezelfde waarheidstabel hebben.
Als f en g equivalente proposities zijn schrijven we ook wel $f \equiv g$.

Vaak kunnen we een propositie vervangen door een eenvoudigere equivalente propositie. Bijvoorbeeld $a \wedge a$ is logisch equivalent met a, dus $a \wedge a \equiv a$.

Opgave 1.G Laat zien dat de volgende proposities telkens logisch equivalent aan elkaar zijn.

(i) $(a \wedge b) \wedge c$ en $a \wedge (b \wedge c)$
(ii) $(a \vee b) \vee c$ en $a \vee (b \vee c)$

In dit geval doen de haakjes er dus eigenlijk niet toe. Soms laten we in zo'n geval de haakjes ook wel weg. In conventie 1.3 hadden we afgesproken dat alle binaire voegtekens rechts associatief zijn. Hier zie je dus dat voor $\wedge$ en $\vee$ die keuze arbitrair is: hadden we afgesproken $\wedge$ en $\vee$ als links associatief te beschouwen, dan had dat geen invloed gehad op het al dan niet waar zijn van een propositie.[3] In opgave 1.D hebben we gezien dat het voor $\to$ wel van belang is dat $\to$ als rechts associatief moet worden beschouwd!

Opmerking 1.12 De proposities $a \wedge b$ en $b \wedge a$ zijn wiskundig gezien logisch equivalent. In het Nederlands is dit niet altijd eenduidig, met de zin 'ze trouwde en kreeg een kind' bedoelen we waarschijnlijk wat anders dan met de zin 'ze kreeg een kind en trouwde'.

Hieronder volgen enkele equivalenties die de distributie van de $\neg$, $\wedge$ en $\vee$ operatoren over haakjes weergeven. De eerste twee hebben een naam en heten de wetten van *De Morgan*.

[3] We hebben overigens voor rechts associatief gekozen omdat de bewijsassistent Coq die bij Beweren en Bewijzen wordt gebruikt, dat ook doet.

1. $\neg(f \wedge g) \equiv \neg f \vee \neg g$.
2. $\neg(f \vee g) \equiv \neg f \wedge \neg g$.
3. $f \wedge (g \vee h) \equiv f \wedge g \vee f \wedge h$.[4]
4. $f \vee g \wedge h \equiv (f \vee g) \wedge (f \vee h)$.

Opgave 1.H Laat f en g proposities zijn. Is de volgende uitspraak waar? $f \equiv g$ dan en slechts dan als $\models f \leftrightarrow g$.

1.7 Logisch gevolg

In het Nederlands volgt uit 'het regent en de zon schijnt' dat 'de zon schijnt'. Net zo willen we dat uit $a \wedge b$ volgt dat a. Dat maken we nu precies.

Definitie 1.13 Een propositie g is een *logisch gevolg* van de propositie f als g waar is in ieder model waarin f waar is. Anders gezegd: een propositie g is een logisch gevolg van de propositie f als er op alle plaatsen waar in de waarheidstabel van f een 1 staat, er in de waarheidstabel van g ook een 1 staat. Notatie $f \models g$.

Als g *geen* logisch gevolg van f is, kan dat genoteerd worden als $f \not\models g$.

Opgave 1.I Zijn de volgende uitspraken waar?

(i) $a \wedge b \models a$
(ii) $a \vee b \models a$
(iii) $a \models a \vee b$
(iv) $a \wedge \neg a \models b$

Stelling 1.14 *Laat f en g proposities zijn. Dan geldt:*

$$\models f \to g \text{ dan en slechts dan als } f \models g.$$

Het bewijs van deze stelling zou je zelf moeten kunnen bedenken. Vergelijk bijvoorbeeld opgave 1.F en opgave 1.I.

Opmerking 1.15 Merk op dat $\models$ en $\equiv$ niet in de definitie van de proposities voorkomen. Deze tekens zijn dan ook geen onderdeel van proposities, maar worden gebruikt om een uitspraak *over* bepaalde proposities te doen. In het bijzonder betekent dit dat strings als $(f \equiv g) \to (\models g)$ en $\neg \models f$ geen betekenis hebben en in dus ook niet mogen worden opgeschreven.

Opmerking 1.16 Voor meer informatie over logica kun je bijvoorbeeld de boeken [3] en [2] bekijken.

[4] NB: Volgens de haakjesconventie is dit $(f \wedge g) \vee (f \wedge h)$.

1.8 Belangrijke begrippen

2 Predikaatlogica

"Een vrouw is gelukkig als zij van iemand houdt
en een man is gelukkig als iemand van hem houdt."

We gaan het niet hebben over de juistheid van deze zin, maar over hoe we deze formeel kunnen opschrijven. Dat zal als voordeel hebben dat we beter kunnen analyseren of dit soort zinnen inderdaad waar zijn.

We springen meteen in het diepe en geven de formalisering van bovenstaande zin.

Woordenboek

V	verzameling van (alle) vrouwen
M	verzameling van (alle) mannen
$H(x, y)$	x houdt van y
$G(x)$	x is gelukkig

Vertaling

$$\forall v \in V \Big[(\exists x{\in}(M \cup V) H(v, x)) \to G(v)\Big] \wedge \forall m \in M \Big[(\exists x{\in}(M \cup V) H(x, m)) \to G(m)\Big] \quad (1)$$

De $\forall$ staat hier voor "voor alle" en de $\exists$ staat voor "er is een". En de $\in$ staat voor de gebruikelijke "is element van" en $\cup$ staat voor "vereniging". Als we deze formule (bijna) letterlijk terugvertalen naar natuurlijke taal staat er

> Voor iedere vrouw geldt dat als er een persoon is van wie zij houdt dan is zij gelukkig en voor iedere man geldt dat als er een persoon is die van hem houdt dan is hij gelukkig.

Ga zelf na dat dit hetzelfde zegt als de zin waar we mee begonnen zijn en ga ook na dat je dit uit de formule kunt aflezen. Je ziet dat we ook andere haakjes $[\_]$ gebruiken. Dat is alleen voor de leesbaarheid, zodat je beter kunt zien welk beginhaakje bij welk eindhaakje hoort. Omdat een woordenboek de mogelijkheid geeft om een uitspraak te vertalen, noemen we zo'n lijstje ook wel een *interpretatie*.

2.1 Predikaten, relaties en individuen

Toen we nog propositielogica deden konden we de zin

Als Sharon gelukkig is, dan is Koos niet gelukkig

als volgt formaliseren.

Woordenboek

SG	Sharon is gelukkig
KG	Koos is gelukkig

Vertaling

$$SG \to \neg KG$$

Maar als er nu ook nog Joris en Maud zijn, dan krijgen we wel een lang woordenboek met de extra proposities JG en MG.

We zetten daarom een vergrootglas op de uitspraak

Sharon is gelukkig

en analyseren deze als bestaande uit het *predikaat* $G(\ )$ toegepast op het *subject* s (Sharon) en dat schrijven we als $G(s)$. Het voordeel is dat we dan meteen kunnen schrijven

$$G(k),\ G(j) \text{ en } G(m)$$

voor de subjecten k, j en m. Een andere term die gebruikt wordt om een subject aan te duiden is *individu*.

Ook voor de subjectnamen gebruiken we het woordenboek om precies aan te geven welk subject er bij welke naam hoort. Bovendien geven we hier aan tot welk *domein* het subject behoort.

Woordenboek

s	Sharon $\in$ "vrouwen"
k	Koos $\in$ "mannen"
j	Joris $\in$ "mannen"
m	Maud $\in$ "vrouwen"

Het voordeel is nu nog niet duidelijk. In plaats van vier proposities SG, KG, JG, MG hebben we nu één predikaat G en vier subjecten s, k, j, m dat is dus in totaal vijf symbolen. Maar onze vier spelers kunnen nog meer eigenschappen hebben:

Woordenboek

$L(x)$	x is lang
$K(x)$	x is knap
$A(x)$	x is aardig
$I(x)$	x is intelligent

Bovendien zijn er naast de predikaten (eigenschappen) ook nog (binaire) relaties. Bijvoorbeeld:

$$H(x,y):\ x \text{ houdt van } y.$$

We kunnen de zin

Sharon is een intelligente knappe vrouw;
er is een aardige lange man die van zo'n vrouw houdt

nu formaliseren met de formule

$$\begin{aligned} I(s) \ &\wedge\ K(s) \\ &\wedge\ \exists x \in M \left[A(x) \wedge L(x) \wedge \exists y \in V\ [H(x,y) \wedge K(y) \wedge I(y)] \right]. \end{aligned}$$

Of bedoelen we misschien

$$\begin{aligned} I(s) \ &\wedge\ K(s) \\ &\wedge\ \exists x \in M\ [L(x) \wedge A(x) \wedge H(x,s)]? \end{aligned}$$

Merk op dat we de twee beweringen in de twee zinnen tot één formule maken door er een $\wedge$ tussen te zetten. Dat Sharon een vrouw is ligt besloten in het woordenboek, dus die informatie hoeven we niet meer toe te voegen (door middel van "$s \in V$" of iets dergelijks).

Of we de eerste of de tweede formalisatie verkiezen hangt ervan af hoe we de verwijzing "zo'n vrouw" vertalen. (Het kan slaan op "Sharon", maar ook op "intelligente knappe vrouw".) Hier gaat de logica zelf niet over, maar de logica maakt wel expliciet dat er hier een keuze gemaakt moet worden; de ambiguïteit in de natuurlijke taal wordt expliciet gemaakt.

Opgave 2.A Geef twee mogelijke vertalingen voor de volgende zin.

Sharon houdt van Maud; een aardige man houdt van zo'n knappe vrouw.

We gaan nu de andere kant op; een formele uitspraak ontcijferen.

Opgave 2.B Vertaal de onderstaande formules naar een zin in het Nederlands.

(i) $\exists x \in M \Big[L(x) \wedge \exists v \in V \, [K(v) \wedge I(v) \wedge H(x,v)] \Big]$

(ii) $\exists x \in M \Big[L(x) \wedge \exists v \in V \, [K(v) \wedge \neg I(v) \wedge H(x,v)] \wedge \exists w \in V [I(w) \wedge H(w,x)] \Big]$

2.2 De taal van de predikaatlogica

We definiëren precies wat de taal van de predikaatlogica is.

Definitie 2.1 De taal van de predikaatlogica bevat de volgende ingrediënten.

1. *Variabelen*, meestal $x, y, z, x_0, x_1, \ldots$, en soms ook $v, w, \ldots$
2. *Individu constanten*, of ook wel 'namen', meestal $a, b, c, \ldots$,
3. *Domeinen*, zoals $M, V, E, \ldots$,
4. *Relatiesymbolen*, met een vaste "ariteit", die we er vaak expliciet bij schrijven, zoals P^1, R^2, T^3, ... Dit betekent dat P^1 één argument heeft, R^2 twee argumenten enz.
5. De *atomen* (of 'atomaire formules') zijn $P^1(t)$, $R^2(t_1, t_2)$, $T^3(t_1, t_2, t_3)$ enz., waarbij t, t_1, t_2 en t_3 variabelen of individu constanten zijn.
6. De *formules* worden als volgt gevormd.
   - Elke atomaire formule is een formule.
   - Als f en g formules zijn dan zijn $(f \wedge g), (f \vee g), (f \to g), (f \leftrightarrow g)$ en $\neg f$ ook formules.
   - Als f een formule is, x een variabele en D een domein, dan zijn $(\forall x \in D \; f)$ en $(\exists x \in D \; f)$ ook formules.

Conventie 2.2 Net als bij de propositielogica laten we meestal de buitenste haakjes weg. Om ook wat op de binnenste haakjes te bezuinigen breiden we de haakjesconventie van de propositielogica (conventie 1.2) uit met:

- $\forall$ en $\exists$ binden sterker dan alle andere voegtekens.

Daarentegen schrijven we voor de leesbaarheid vaak juist extra, rechte haakjes. Daar waar de definitie eigenlijk zegt $(\forall x \in D \; f)$, schrijven we meestal $\forall x \in D \; [f]$.

Dit betekent dat we onze eerste formule uit de predikaatlogica (1) dus kunnen schrijven als

$$\forall v \in V \left[\exists x{\in}(M \cup V)\ H(v,x) \rightarrow G(v)\right] \wedge \forall m \in M \left[\exists x{\in}(M \cup V)\ H(x,m) \rightarrow G(m)\right]$$

of zelfs als

$$\forall v \in V \left[\exists x{\in}(M \cup V)\,[H(v,x)] \rightarrow G(v)\right] \wedge \forall m \in M \left[\exists x{\in}(M \cup V)\,[H(x,m)] \rightarrow G(m)\right].$$

Opmerking 2.3 Grammatica van de predikaatlogica.

Net als bij de propositielogica kunnen we de taal van de predikaatlogica definiëren als een formele taal. Dat gaat als volgt. (Bekijk deze opmerking nog eens na blok 3, "Talen en automaten".)

Individu	:=	Variabele | Naam
Variabele	:=	$x, y, z, x_1, y_1, z_1, \ldots$
Naam	:=	$a, b, c, d, e, \ldots$
Domein	:=	$D, E, \ldots$
Atoom	:=	P^1(Individu) | R^2(Individu,Individu)
		| T^3(Individu,Individu,Individu) | ...
Formule	:=	Atoom
		| ¬ Formule
		| (Formule → Formule) | (Formule ∧ Formule)
		| (Formule ∨ Formule) | (Formule ↔ Formule)
		| (∀ Variabele ∈ Domein Formule)
		| (∃ Variabele ∈ Domein Formule)

Opgave 2.C Formaliseer de volgende zin.

Sharon is knap; er is een man die lekker in zijn vel zit van wie zij houdt.

Hierbij wordt gedefinieerd dat iemand lekker in zijn vel zit als hij of zij van zich zelf houdt.

Opgave 2.D Formaliseer de volgende zinnen.

(i) *Voor iedere x en y geldt: x houdt van y alleen als x lekker in zijn of haar vel zit.*

(ii) *Voor iedere x en y geldt: x houdt van y als die lekker in zijn of haar vel zit.*

(iii) *Voor iedere x en y geldt: x houdt precies dan van y als y lekker in zijn of haar vel zit.*

(iv) *Er is iemand die van alle mensen houdt.*

2.3 Waarheid en gevolgtrekking

Bekijk de volgende twee formules

$$\begin{aligned} F_1 &= \forall x{\in}D \exists y{\in}D\ K(x,y) \\ F_2 &= \exists x{\in}D \forall y{\in}D\ K(x,y) \end{aligned}$$

(Let op hoe we op haakjes bezuinigen.) Wanneer is een formule *waar*? Waarheid is relatief en hangt af van een *interpretatie* en een model.

Definitie 2.4 Een *model* is het stukje van de 'echte' wereld waarin de formules een betekenis krijgen via de *interpretatie*.

Omdat we nog niet hebben uitgelegd wat een interpretatie is, zegt deze definitie nog niet zoveel. Maar voordat we dat begrip introduceren leggen we eerst uit aan de hand van een paar voorbeelden wat we bedoelen met dat stukje van de echte wereld. Hierbij gaat het om de afbakening van het domein of de domeinen waarover we spreken, samen met eigenschappen en relaties die binnen die domeinen bekend zijn en waarvan kan worden vastgesteld of er aan voldaan wordt.

1. Model M_1

Domein(en)	alle studenten in het lokaal
Eigenschap(pen)	is vrouwelijk
	is ouder dan 20 jaar
Relatie(s)	heeft een lager studentnummer dan
	is niet ouder dan
	zit naast

2. Model M_2

Domein(en)	mensen
	dieren
Eigenschap(pen)	is vrouwelijk
	is mens
	is hond
Relatie(s)	bezit
	is ouder dan

Bij alle studenten in het lokaal is vast te stellen[5] of ze vrouwelijk zijn of niet. En als Sharon en Maud toevallig in dit lokaal zitten, is vast te stellen of zij naast elkaar zitten of niet. Verder zijn er biologen die bij elk dier kunnen vaststellen of het een hond is of niet. En verder is het normaal gesproken ook vast te stellen of een bepaald persoon de eigenaar is van een bepaald dier. Hoe precies dient men eigenlijk nog netjes af te spreken. Moet zo'n eigenaar een aankoopbewijs kunnen laten zien of is het voldoende als hij een etensbak kan laten zien met daarop de naam van het beest?

Kern van de zaak is dat in al deze stukken van de echte wereld duidelijkheid bestaat over de vraag of een bepaalde bewering over eigenschappen van en relaties tussen subjecten uit de domeinen waar is of niet.

Nu kunnen we het begrip interpretatie definiëren.

Definitie 2.5 Een *interpretatie* wordt gegeven door het woordenboek, waarin staat

1. Welke verzamelingen er horen bij de domeinsymbolen,
2. Welke subjecten er horen bij de namen (en in welke domeinverzamelingen deze zitten),
3. Welke predikaten en relaties er horen bij de predikaat- en relatiesymbolen.

[5] Dit nemen we tenminste aan...

In het bijzonder legt de interpretatie dus een verband tussen de symbolen in de formule en het model dat we bekijken. In de literatuur wordt dan ook vaak van een interpretatiefunctie gesproken. Zie bijvoorbeeld [3].

Definitie 2.6 *Een formule f heet waar in een model onder een interpretatie als de door de interpretatie bepaalde vertaling een uitspraak oplevert die in het model waar is.*

Voorbeeld 2.7 In het geval van Sharon, Koos, Joris en Maud is de interpretatie al gegeven in het woordenboek: we weten nu wat we met $H(s,k)$ bedoelen en wat met $\exists x \in M\ H(x,s)$. Het model is de feitelijke situatie omtrent deze mensen, waar we kunnen nagaan of $H(s,k)$ waar is (i.e. of Sharon van Koos houdt) en of $\exists x \in M\ H(x,s)$ waar is (i.e. of er een man is die van Sharon houdt).

Of de formules F_1 en F_2 waar zijn binnen model M_1, kunnen we dus alleen maar bepalen als we ook de interpretatie van de symbolen vastleggen. Hier geven we drie verschillende interpretaties:

1. Interpretatie I_1

D	alle studenten in het lokaal
$K(x,y)$	x heeft een lager studentnummer dan y

2. Interpretatie I_2

D	alle studenten in het lokaal
$K(x,y)$	x is niet ouder dan y

3. Interpretatie I_3

D	alle studenten in het lokaal
$K(x,y)$	x zit naast y

We kunnen de waarheid van F_1, respectievelijk F_2, in deze modellen onder de gegeven interpretaties vaststellen door te kijken of de eigenschap geldt, die door de formule tot uitdrukking wordt gebracht.

Opgave 2.E (i) Ga na dat F_2 in model M_1 onder interpretatie I_1 niet geldt. Geldt F_1 wel?
(ii) Ga na dat F_1 in model M_1 onder interpretatie I_2 geldt. Geldt F_2 ook?
(iii) Kijk om je heen en ga na of F_1 in het model M_1 onder interpretatie I_3 geldt. Ga ook na of F_2 geldt.

Omdat in model M_1 slechts over één domein wordt gesproken, lijkt het alsof bij de interpretatie het domein altijd hetzelfde moet zijn. Model M_2 laat echter zien dat dat niet hoeft:

1. Interpretatie I_4

D	mensen
$K(x,y)$	x is niet ouder dan y

2. Interpretatie I_5

D	mensen verenigd met dieren
$K(x,y)$	x bezit y

Merk hierbij op dat als x een dier is en y een mens, de bewering $K(x,y)$ automatisch onwaar is omdat geen enkel dier een mens bezit.

3. Interpretatie I_6

D	dieren
$K(x,y)$	y is ouder dan x

We bekijken de formules G_1 en G_2 en we bekijken twee modellen waarin we ze kunnen interpreteren. (Merk het verschil op met de hiervoor gedefinieerde F_1 en F_2.)

$$\begin{aligned} G_1 &= \forall x{\in}D \exists y{\in}D\ K(x,y) \\ G_2 &= \forall x{\in}D \exists y{\in}D\ K(y,x) \end{aligned}$$

Voorbeeld 2.8 Bekijk model M_3

Domein(en)	Natuurlijke getallen, $\mathbb{N} = \{0,1,2,3,\ldots\}$
Relatie(s)	kleiner dan $(<)$

en model M_4

Domein(en)	Rationale getallen, $\mathbb{Q}$, (alle breuken, bijvoorbeeld $-\frac{1}{2}, 3(=\frac{3}{1}), 0, \ldots$)
Relatie(s)	kleiner dan $(<)$

Hierbij definiëren we ook twee voor de hand liggende interpretaties. Interpretatie I_7:

D	$\mathbb{N}$
$K(x,y)$	$x < y$

en interpretatie I_8:

D	$\mathbb{Q}$
$K(x,y)$	$x < y$

In model M_3 onder interpretatie I_7 is de formule G_1 waar. Immers, voor ieder getal $x \in \mathbb{N}$ is er een getal $y \in \mathbb{N}$ (bijvoorbeeld $y = x + 1$) zodat $x < y$. In model M_4 onder interpretatie I_8 is de formule G_1 ook waar. Immers, voor ieder getal $x{\in}\mathbb{Q}$ is er een getal $y{\in}\mathbb{Q}$ (bijvoorbeeld wederom $y = x + 1$) zodat $x < y$. Als we zoals in deze twee gevallen voor elke x zo'n y expliciet kunnen beschrijven, noemen we dat vaak een *algoritme* of *recept*.

Conventie 2.9 Als een formule f waar is in een model M onder interpretatie I, schrijven we

$$(M, I) \models f$$

Dus we hebben in het voorbeeld de volgende zaken ingezien:

$$\begin{aligned} (M_3, I_7) &\models G_1 \\ (M_4, I_8) &\models G_1 \end{aligned}$$

Als we de modellen M_3 en M_4 geen naam hadden gegeven, hadden we het met wat extra haakjes alsvolgt kunnen opschrijven:

$$\begin{aligned}((\mathbb{N},<),I_7) &\models G_1\\ ((\mathbb{Q},<),I_8) &\models G_1\end{aligned}$$

Juist omdat het (belangrijke deel van het) model al vastligt door het geven van de interpretatie, komt het vrij vaak voor dat het model niet expliciet wordt opgeschreven.

Opgave 2.F Ga na dat G_2 waar is in model M_4 onder interpretatie I_8, maar niet in model M_3 onder interpretatie I_7. Ofwel: $((\mathbb{Q},<),I_8) \models G_2$, en $((\mathbb{N},<),I_7) \not\models G_2$.

Opgave 2.G Definieer interpretatie I_9 als volgt:

D	$\mathbb{N}$
$K(x,y)$	$x = 2 \cdot y$

Welke van de formules G_1 en G_2 zijn waar?

Opgave 2.H Definieer interpretatie I_{10} als volgt:

D	$\mathbb{Q}$
$K(x,y)$	$x = 2 \cdot y$

Welke van de formules G_1 en G_2 zijn waar?

Opgave 2.I We bekijken als model de landen van Europa met de volgende interpretatie I_{11}.

L	verzameling van landen van Europa
n	Nederland
d	Duitsland
i	Ierland
$G(x,y)$	x grenst aan y
$D(x,y,z)$	x, y en z hebben een drielandenpunt met elkaar

(i) Formaliseer de zin "Nederland en Duitsland delen een drielandenpunt"

(ii) Welke van de volgende formules is waar in dit model onder deze interpretatie:

(1) $G_3 := \forall x \in L \exists y \in L[G(x,y)]$

(2) $G_4 := \forall x,y \in L[(\exists z \in L\ D(x,y,z)) \rightarrow G(x,y)]$

(3) $G_5 := \forall x \in L[G(i,x) \rightarrow \exists y \in L[D(i,x,y)]]$.

Opgave 2.J Bedenk zelf een model M_5 en een interpretatie I_{12} zodat

$$(M_5, I_{12}) \models \forall x \in D \exists y \in E[R(x,y) \wedge \neg R(y,x) \wedge \neg R(y,y)]$$

In conventie 2.9 hebben we al gezien dat we $(M,I) \models f$ schrijven als een formule f van de predikaatlogica (met of zonder gelijkheid) waar is in een model M onder een interpretatie (woordenboek) I als na vertaling de formule klopt. Dit begrip breiden we nog iets verder uit.

Definitie 2.10 Een formule in de predikaatlogica f heet *waar zonder meer*, notatie $\models f$, als voor *ieder* model en interpretatie de vertaling klopt.

Meestal laten we overigens het *zonder meer* weg. De bewering f is *waar* is dan ook gelijk aan de bewering f is *waar zonder meer*. Verder wordt voor dit begrip ook vaak de term *logisch waar* gebruikt vanwege de analogie met dat begrip in de propositielogica.

Voorbeeld 2.11 Bekijk de volgende uitspraken:
(i) $\models \forall x{\in}D\ (P(x) \to P(x))$.
(ii) $\models [\exists x{\in}D \forall y{\in}D\ P(x,y)] \to [\forall y{\in}D \exists x{\in}D\ P(x,y)]$.
(iii) $\not\models [\forall y{\in}D \exists x{\in}D\ P(x,y)] \to [\exists x{\in}D \forall y{\in}D\ P(x,y)]$.

Dat de formule in uitspraak (iii) niet waar is zien we door de volgende interpretatie te nemen. $D := \mathbb{N}$ en $P(x,y) := x{>}y$. Onder deze interpretatie geldt $\forall y{\in}D \exists x{\in}D\ P(x,y)$, want voor ieder getal $y{\in}\mathbb{N}$ is er een groter getal $x{\in}\mathbb{N}$ te vinden, maar niet $\exists x{\in}D \forall y{\in}D\ P(x,y)$, want dan zou er in $\mathbb{N}$ een grootste getal moeten bestaan en dat is niet zo.

Definitie 2.12 Laten f en g formules uit de predikaatlogica zijn. Dan zeggen we dat g uit f volgt, notatie $f \models g$, als $\models f \to g$. Dat houdt in dat in iedere situatie waarin f geldt, ook g geldt.

Uit (ii) in het vorige voorbeeld blijkt dat $\forall y{\in}D \exists x{\in}D\ P(x,y)$ uit $\exists x{\in}D \forall y{\in}D\ P(x,y)$ volgt.

2.4 De taal van de predikaatlogica met gelijkheid

Soms wordt een formalisering van het volgende gevraagd.

Sharon is knap; er is een man die alleen aan haar aandacht geeft.

Om dit te formaliseren hebben we een *gelijkheidsrelatie* nodig. De formalisering wordt dan als volgt.

$$K(s) \wedge \exists x{\in}M\ (A(x,s) \wedge \forall y{\in}V (A(x,y) \to y = s)) \tag{2}$$

Om dit als een formule van de predikaatlogica te kunnen zien moeten we de taal van de predikaatlogica uitbreiden.

Definitie 2.13 De *predikaatlogica met gelijkheid* wordt gedefinieerd door aan de predikaatlogica standaard een binaire relatie "=" toe te voegen. De interpretatie van deze relatie in een model is altijd "gelijk aan".

Dit zorgt ervoor dat het begrip van officiële formules uit definitie 2.1 als volgt wordt uitgebreid.

- Als x en y variabelen en a en b constanten zijn, dan zijn $(x = y)$, $(x = a)$, $(a = x)$ en $(a = b)$ formules.

Opmerking 2.14 In formule (2) hebben we schaamteloos een andere A gebruikt dan in ons woordenboek op pagina 12. Daar had A ariteit 1 en betekende $A(x)$ dat x aardig is. Hier heeft A ariteit 2 en betekent $A(x,y)$ dat x aandacht geeft aan y. Eigenlijk hadden we hier dus een nieuw woordenboek moeten geven, maar omdat de voorbeelden in dit dictaat niet heel erg ingewikkeld zijn, gaan wij er stilzwijgend vanuit dat je zelf doorhebt welke betekenis dubbel gebruikte letters in een bepaalde context hebben.

Opmerking 2.15 De predikaatlogica met gelijkheid is uitermate geschikt om beweringen met eigenschappen als precies één, precies twee, minimaal twee, maximaal drie, verschillend en alleen weer te geven.

Opgave 2.K Beschouw de interpretatie I_{13}:

M	domein van de mensen;
$V(x)$	x is vrouw;
$O(x,y)$	x is ouder van y;
$G(x,y)$	x is getrouwd met y.

Formaliseer de volgende zinnen in de predikaatlogica met gelijkheid.

(i) *Iedereen heeft precies één moeder.*

(ii) *Iedereen heeft precies twee oma's.*

(iii) *Iedere getrouwde man heeft precies één echtgenote.*

Opgave 2.L Gebruik interpretatie I_{13} van opgave 2.K. Formaliseer de volgende eigenschappen.

(i) $S(x,y)$: x en y hebben samen een kind gemaakt.
(ii) $B(x,y)$: x is broer van y (pas op: zie ook volgende item).
(iii) $H(x,y)$: x is halfzus van y.

Vertaal terug naar de omgangstaal.

(iv) $\exists x{\in}M \forall y{\in}M\ O(x,y)$. Geldt dit?
(v) $\forall z_1{\in}M \forall z_2{\in}M[(\exists x{\in}M \exists y_1{\in}M \exists y_2{\in}M\ O(x,y_1) \wedge O(y_1,z_1) \wedge O(x,y_2) \wedge O(y_2,z_2)) \rightarrow \neg(\exists w{\in}M(O(z_1,w) \wedge O(z_2,w)))]$. Geldt dit?

Opgave 2.M Gegeven de interpretatie I_{14}

D	$\mathbb{N}$;
$P(x,y,z)$	$x+y=z$;
$M(x,y,z)$	$x \cdot y = z$.

Formaliseer

(i) $x < y$.
(ii) $x \mid y$ (x is deler van y).
(iii) x is priemgetal.

2.5 Belangrijke begrippen

3 Talen en automaten

Het genereren en beschrijven van talen is een belangrijke toepassing van computers. Een computer kan alleen goed overweg met talen die een precieze "formele" definitie hebben, zoals bijvoorbeeld een programmeertaal. Maar ook kan men proberen een natuurlijke taal aan de computer te leren door de regels van de natuurlijke taal precies (formeel) te definiëren. Voor nu houden we ons alleen bezig met formeel gedefinieerde talen. Een *taal* vatten we op als een verzameling *woorden*. Een *woord* is een string symbolen uit een van tevoren vastgelegd *alfabet*.

Met deze informele definities kunnen we al een heel aantal interessante vragen over talen precies maken en beantwoorden, zoals

---

- Zit het woord w in de taal L?
- Zijn de talen L en L' gelijk?

Allerlei informaticaproblemen die op het eerste gezicht misschien niets met talen te maken hebben kunnen als een taalprobleem beschouwd worden. Met behulp van (formele) talen kunnen puzzels en belangrijkere zaken opgelost worden.

Van talen naar automaten vergt slechts een kleine stap. Automaten zijn ook weer formele wiskundige objecten. De hoofdvraag die wij hierbij telkens stellen is welke taal er geaccepteerd wordt door zo'n automaat. Vandaar dat wij de onderwerpen talen en automaten in één blok behandelen.

3.1 Alfabet, woord, taal

Definitie 3.2 1. Een *alfabet* is een eindige verzameling Σ van *symbolen*.[6]

2. Een *woord* over Σ is een eindig rijtje symbolen uit dit alfabet.
3. λ is het woord dat bestaat uit 0 symbolen, ook wel genaamd het *lege woord*.
4. Σ^* is de verzameling van alle woorden over Σ.
5. Een *taal* L over Σ is een deelverzameling van Σ^*.

Voorbeeld 3.3 (i) $\Sigma = \{a, b\}$ is een alfabet.
(ii) *abba* zit in Σ^* (notatie: $abba \in \Sigma^*$).
(iii) $abracadabra \notin \Sigma^*$.
(iv) $abracadabra \in \Sigma_0^*$, met $\Sigma_0 = \{a, b, c, d, r\}$.
(v) λ het *lege woord* zit in Σ^* voor iedere Σ.

We kunnen talen op allerlei manieren beschrijven. Enkele manieren die we in dit college zullen tegenkomen zijn *reguliere expressies* (zie sectie 3.2), *grammatica's* (zie sectie 3.3) en *eindige automaten* (zie sectie 3.5). Maar we kunnen talen ook gewoon als verzamelingen beschrijven.

Definitie 3.4 We introduceren wat handige notatie.

[6] Hoewel het alfabet best symbolen als '*' en ',' mag bevatten, worden de symbolen vaak ook letters genoemd.

1. We schrijven a^n voor $a \dots a$ met n keer een a. Preciezer gezegd: $a^0 = \lambda$ en $a^{n+1} = a^n a$.
2. De *lengte* van een woord w geven we weer als $|w|$.
3. Als w een woord is, dan is w^R het *omgekeerde* ("*reverse*") woord, dus bijvoorbeeld $(abaabb)^R = bbaaba$.

Voorbeeld 3.5 (i) $L_1 := \{w \in \{a,b\}^* \mid w$ bevat een even aantal a's$\}$.
(ii) $L_2 := \{a^n b^n \mid n \in \mathbb{N}\}$.
(iii) $L_3 := \{wcv \in \{a,b,c\}^* \mid w$ bevat geen b, v bevat geen a en $|w| = |v|\}$.
(iv) $L_4 := \{w \in \{a,b,c\}^* \mid w = w^R\}$.
Bedenk bij ieder tweetal talen uit dit voorbeeld een woord w dat wel in de ene, maar niet in de andere taal zit.

Als we twee talen L en L' hebben, kunnen we met de bekende verzamelingsoperaties nieuwe talen definiëren. We brengen wat notatie in herinnering.

Definitie 3.6 Laat L en L' talen over het alfabet Σ zijn.

1. $\overline{L}$ is het *complement* van L: de taal bestaande uit de woorden w die *niet* in L zitten. (Dus de $w \in \Sigma^*$ waarvoor geldt $w \notin L$.)
2. $L \cap L'$ is de *doorsnijding* van L en L': de taal bestaande uit de woorden w die zowel in L als in L' zitten.
3. $L \cup L'$ is de *vereniging* van L en L': de taal bestaande uit de woorden w die in L of in L' zitten (of in allebei).

Opgave 3.A In voorbeeld 3.5 heb je voorbeelden gezien van beschrijvingen van talen. Probeer het nu zelf.
(i) Beschrijf $L_1 \cap L_2$.
(ii) Beschrijf $L_2 \cap L_4$.
(iii) Beschrijf $L_3 \cap L_4$.

Er zijn ook operaties die specifiek voor talen zijn. Die definiëren we nu.

Definitie 3.7 Laat L en L' talen over het alfabet Σ zijn.

1. LL' is de *concatenatie* van L en L': de taal bestaande uit de woorden van de vorm wv met $w \in L$ en $v \in L'$.
2. L^R is de taal van *omgekeerde* woorden van L en bestaat uit de woorden w^R met $w \in L$.
3. L^* is de taal bestaande uit eindige concatenaties van woorden uit L en bestaat uit de woorden van de vorm $w_1 w_2 \dots w_k$ met $k \geq 0$ en $w_1, w_2, \dots, w_k \in L$.

De taal L^* heet ook wel de *Kleene afsluiting* van L. De taal L^* bestaat uit concatenaties van 0 of meer woorden uit L, dus het is altijd zo dat $\lambda \in L^*$. Verder is het zo dat $L \subset L^*$ voor iedere taal L. Ga voor jezelf na wat L^* is als $L = \emptyset$ of als $L = \{\lambda\}$.

Opgave 3.B Gebruik de definities uit voorbeeld 3.5.
(i) Bewijs dat $L_1 = {L_1}^*$
(ii) Geldt $L_2 L_2 = L_2$? Geef een bewijs of een tegenvoorbeeld.
(iii) Geldt $\overline{L_1} = \overline{L_1}^*$? Geef een bewijs of een tegenvoorbeeld.
(iv) Voor welke talen uit voorbeeld 3.5 geldt $L = L^R$? (Alleen antwoord, geen bewijs.)

3.2 Reguliere talen

Een populaire manier om talen te beschrijven is door middel van *reguliere expressies*. Een taal die door een reguliere expressie beschreven kan worden noemen we een *reguliere taal*. In de informatica worden reguliere talen gezien als (relatief) eenvoudig: als L een reguliere taal is, dan is het niet moeilijk om een programmaatje te schrijven dat bepaalt of een gegeven woord w in L zit. Zo'n programma heet een *parser* en het oplossen van de vraag "$w \in L$" heet ook wel *parseren*. Parsers voor reguliere talen zijn dus eenvoudig te maken: er zijn zelfs (vele) programma's die parsers voor je genereren (als je er een reguliere expressie in stopt)! De door dit soort "parser generators" gegenereerde parsers zijn zelfs bijzonder *efficiënt*: ze beslissen heel snel of w al dan niet in L zit. We gaan het in dit college verder niet hebben over parseren, maar het zal duidelijk zijn dat reguliere talen een belangrijke klasse van talen vormen. Dus die gaan we nader bekijken.

Definitie 3.8 Laat Σ een alfabet zijn. De *reguliere expressies* over Σ zijn als volgt gedefinieerd.

1. $\emptyset$ is een reguliere expressie,
2. λ is een reguliere expressie,
3. x is een reguliere expressie voor iedere $x \in \Sigma$,
4. als r_1 en r_2 reguliere expressies zijn, dan zijn $(r_1 \cup r_2)$ en $(r_1\, r_2)$ ook reguliere expressies,
5. als r een reguliere expressie is, dan is r^* ook een reguliere expressie.

Voorbeeld 3.9 Dus aba, $ab^*(a \cup \lambda)$, $(a \cup b)^*$, $(a \cup \emptyset)b^*$ en $(ab^* \cup b^*a)^*$ zijn reguliere expressies.

Het verband tussen talen en reguliere expressies kunnen we nu formaliseren.

Definitie 3.10 Bij iedere reguliere expressie r definiëren we de *taal van* r, $\mathcal{L}(r)$ als volgt:

1. $\mathcal{L}(\emptyset) := \emptyset$,
2. $\mathcal{L}(\lambda) := \{\lambda\}$,
3. $\mathcal{L}(x) := \{x\}$ voor iedere $x \in \Sigma$,
4. $\mathcal{L}(r_1 \cup r_2) := \mathcal{L}(r_1) \cup \mathcal{L}(r_2)$,
5. $\mathcal{L}(r_1\, r_2) := \mathcal{L}(r_1)\mathcal{L}(r_2)$,
6. $\mathcal{L}(r^*) := \mathcal{L}(r)^*$.

Als we nagaan wat dat betekent voor de reguliere expressies die we boven als voorbeeld gaven, dan zien we:

- $\mathcal{L}(aba) = \{aba\}$,
- $\mathcal{L}(ab^*(a \cup \lambda)) = \{a\}\{b\}^*\{a, \lambda\}$ en bestaat uit de woorden die beginnen met een a, dan een willekeurig aantal b's en dan niets of weer een a,
- $\mathcal{L}((a \cup b)^*) = \{a, b\}^*$, dus dat zijn alle woorden over het alfabet $\{a, b\}$,

- $\mathcal{L}((a \cup \emptyset)b^*) = \{a\}\{b\}^*$ en dat is dus dezelfde taal als van ab^*,
- $\mathcal{L}((ab^* \cup b^*a)^*) = (\{a\}\{b\}^* \cup \{b\}^*\{a\})^*$. Deze taal is wat moeilijker in woorden te beschrijven.

In reguliere expressies wordt soms ook de operator + gebruikt: de reguliere expressie a^+ staat dan voor "1 of meer keer een a". We hebben de + echter niet nodig, want in plaats van a^+ kunnen we gewoon aa^* schrijven, want dat betekent precies hetzelfde (eerst een a en dan nog 0 of meer keer een a).

Opgave 3.C (i) Laat zien dat we de operator ?, waarbij $a^?$ staat voor 0 of 1 keer a niet hoeven toe te voegen aan de reguliere expressies, omdat we hem al kunnen maken.
(ii) Wat is $\mathcal{L}(\emptyset ab^*)$?
(iii) Geef een reguliere expressie die de taal $L_5 := \{w \in \{a,b\}^* \mid w \text{ bevat minstens één } a\}$ beschrijft.
(iv) Geef een reguliere expressie die de taal L_1 uit voorbeeld 3.5 beschrijft.
(v) Laat zien dat $\mathcal{L}(ab(ab)^*) = \mathcal{L}(a(ba)^*b)$.

We zien dat de reguliere expressie $\emptyset$ niet zo nuttig is: in plaats van $r \cup \emptyset$ kunnen we net zo goed r schrijven en $\emptyset r$ beschrijft $\emptyset$. De reguliere expressie $\emptyset$ wordt alleen gebruikt om de lege taal $\mathcal{L}(\emptyset)$ te kunnen beschrijven en we zullen hem verder niet tegenkomen.[7]

Definitie 3.11 Laat Σ een alfabet zijn. We noemen een taal L over Σ *regulier* als er een reguliere expressie is die hem beschrijft. Oftewel: een taal L over Σ is regulier dan en slechts dan als er een reguliere expressie r is zodat $L = \mathcal{L}(r)$.

De taal L_1 uit voorbeeld 3.5 is regulier, zoals we in opgave 3.C gezien hebben. De talen L_2, L_3 en L_4 uit voorbeeld 3.5 zijn niet regulier. Het bewijs van de niet-regulariteit van deze talen voert te ver voor dit college.

Opgave 3.D Laat zien dat de volgende talen regulier zijn.
(i) $L_6 := \{w \in \{a,b\}^* \mid \text{iedere } a \text{ in } w \text{ wordt direct gevolgd door een } b\}$,
(ii) $L_7 :=$ de taal van alle goedgevormde integer-expressies. Deze bestaan uit de symbolen $0,1,2,3,4,5,6,7,8,9$, maar ze beginnen nooit met een 0 (behalve het getal 0 zelf natuurlijk) en mogen eventueel worden voorafgegaan door een + of een $-$.
(iii) $L_8 :=$ de taal van alle goedgevormde rekenkundige expressies zonder haakjes. Deze bestaan uit natuurlijke getallen met daartussen de operatoren $+$, $-$ of $\times$, bijvoorbeeld $7 + 3 \times 29 - 78$. (Hint: maak eerst een reguliere expressie voor de natuurlijke getallen; zo'n getal begint (bijna) nooit met een 0.)

Als een taal L regulier is dan is de taal L^R ook regulier, want als L wordt beschreven door de reguliere expressie e, dus als $L = \mathcal{L}(e)$, wordt L^R beschreven door de reguliere expressie e^R want $L^R = \mathcal{L}(e^R)$. Er zijn nog meer mooie eigenschappen van de klasse van reguliere talen: Als L en L' regulier zijn, dan zijn $L \cup L'$ en $L \cap L'$ ook regulier en $\overline{L}$ is ook regulier. Van $L \cup L'$ is dat eenvoudig in te zien (ga het voor jezelf na!), maar van $L \cap L'$ en $\overline{L}$ niet.

[7] Omdat $\mathcal{L}(\emptyset) = \emptyset$ als verzameling, kunnen we hier toevallig een en het zelfde symbool $\emptyset$ zowel voor de reguliere expressie als de taal zelf schrijven.

Conventie 3.12 Men identificeert soms een reguliere taal met een reguliere expressie die hem beschrijft. Men heeft het dan bijvoorbeeld over "de taal $b^*(aab)^*$", terwijl de taal $\mathcal{L}(b^*(aab)^*)$ bedoeld wordt. We zullen dat in dit college niet doen; we proberen consequent het onderscheid tussen de reguliere expressie en de bijbehorende taal expliciet maken.

Opgave 3.E Welke van de volgende reguliere expressies beschrijven dezelfde taal? Toon het aan of geef een string die wel in de ene en niet in de andere taal zit.

(i) $b^*(aab)^*$.
(ii) $b^*(baa)^*b$.
(iii) $bb^*(aab)^*$.

3.3 Contextvrije grammatica's

Er is ook een andere manier om talen te definiëren. Daarbij proberen we niet de taal in één keer te *beschrijven* maar we geven een voorschrift hoe de woorden uit de taal *gegenereerd* (of *geproduceerd*) kunnen worden. Zo'n voorschrift heet een "grammatica".

Voorbeeld 3.13 $\Sigma = \{a, b\}$. De taal $L_9 \subseteq \Sigma^*$ wordt gedefinieerd door de *producties* vanuit S (start) die met de volgende *productieregels* te maken zijn.

$$\begin{aligned} S &\rightarrow aAb \\ A &\rightarrow aAb \\ A &\rightarrow \lambda \end{aligned}$$

De manier om nu woorden te genereren is als volgt. We beginnen bij S (start). Dit is het *startsymbool*. S en A zijn de *hulpsymbolen*. We volgen één pijl uit S. (In dit voorbeeld is er slechts één mogelijkheid, maar dat hoeft niet.) Indien er nog een hulpsymbool staat volgen we nog een pijl. Totdat er geen hulpsymbool meer staat en dan hebben we een woord geproduceerd. Voorbeelden van producties:

$$\begin{aligned} &S \rightarrow aAb \rightarrow aaAbb \rightarrow aabb; \\ &S \rightarrow aAb \rightarrow aaAbb \rightarrow aaaAbbb \rightarrow aaabbb. \end{aligned}$$

Deze manier van weergeven van een taal noemen we een grammatica. Bovenstaande grammatica wordt ook wel weergegeven als

$$\begin{aligned} S &\rightarrow aAb \\ A &\rightarrow aAb \mid \lambda \end{aligned}$$

We bedoelen dan dat er vanuit A twee productieregels toepasbaar zijn: $A \rightarrow aAb$ en $A \rightarrow \lambda$.

Dit levert de taal L_9 op waarbij $L_9 = \{ab, aabb, aaabbb, a^4b^4, \ldots, a^nb^n, \ldots\}$. Of anders opgeschreven:

$$L_9 = \{a^nb^n \mid n \in \mathbb{N} \text{ en } n > 0\}$$

Definitie 3.14 Een *contextvrije grammatica* G is een drietal $\langle \Sigma, V, R \rangle$ bestaande uit

1. Een alfabet Σ
2. Een verzameling *hulpsymbolen* V, waarvan er één speciaal is, S, het *startsymbool*.

3. Een verzameling *productieregels* R van de vorm

$$X \to w$$

waarbij X een hulpsymbool is en w een woord bestaande uit letters uit het alfabet en hulpsymbolen. (Anders gezegd: $w \in (\Sigma \cup V)^*$.)

Conventie 3.15 We zullen voor de hulpsymbolen altijd hoofdletters gebruiken (S, A, B, etc) en voor de letters van het alfabet kleine letters (a, b, c, etc).

Voorbeeld 3.16 (i) De taal L_9 uit voorbeeld 3.13 wordt geproduceerd door een contextvrije grammatica.

(ii) De taal L_{10} wordt geproduceerd door de contextvrije grammatica $\langle \Sigma, V, R \rangle$ met $\Sigma = \{a\}$, $V = \{S\}$ en $R = \{S \to aaS, S \to a\}$. Deze grammatica produceert alle woorden bestaande uit een oneven aantal a'tjes.

(iii) De taal L_{11} wordt geproduceerd door de contextvrije grammatica $\langle \Sigma, V, R \rangle$ met $\Sigma = \{a, b\}$, $V = \{S, A, B\}$ en $R = \{S \to AB, A \to Aa, A \to \lambda, B \to Bb, B \to \lambda\}$. De taal L_{11} wordt gevormd door alle woorden bestaande uit eerst een rij van nul of meer a'tjes en dan een rij van nul of meer b'tjes.

Definitie 3.17 Talen geproduceerd door contextvrije grammatica's heten *contextvrije talen*. Voor de taal die wordt geproduceerd door de grammatica G schrijven we ook wel $\mathcal{L}(G)$.

Opmerking 3.18 Contextvrije talen worden systematisch behandeld in het college 'Talen en automaten' van de studierichting Informatica.

Voorbeeld 3.19 De taal van goedgevormde rekenkundige expressies met haakjes is contextvrij. (Deze taal is *niet* regulier!) Een grammatica voor deze taal is de volgende.

$$\begin{array}{lcl} S & \to & B\,S\,O\,S\,E \mid G \\ B & \to & (\\ E & \to &) \\ O & \to & + \mid \times \mid - \\ G & \to & P\,C \\ P & \to & 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\ C & \to & 0C \mid 1C \mid 2C \mid 3C \mid 4C \mid 5C \mid 6C \mid 7C \mid 8C \mid 9C \mid \lambda \end{array}$$

Laat eens zien hoe je $(33 + (20 * 5))$ maakt en $((33 + 20) * 5)$.

Opgave 3.F (i) Laat zien dat de taal van *gebalanceerde haakjesexpressies* contextvrij is. Dat zijn expressies over $\{(,)\}$ waarbij ieder openingshaakje altijd precies één sluithaakje heeft, dus bijvoorbeeld $(( )(( )))$ en $(( ))( )$ zijn gebalanceerd, maar $(( )( )))$ niet.

(ii) Laat zien dat L_1 uit voorbeeld 3.5 contextvrij is.

(iii) Laat zien dat L_2 uit voorbeeld 3.5 contextvrij is. (Kijk naar voorbeeld 3.13.)

(iv) Laat zien dat L_3 uit voorbeeld 3.5 contextvrij is.

(v) Laat zien dat L_4 uit voorbeeld 3.5 contextvrij is.

Opgave 3.G Bekijk de grammatica G_1:

$$\begin{aligned} S &\rightarrow AS \mid Sb \mid \lambda \\ A &\rightarrow aA \mid \lambda \end{aligned}$$

(i) Schrijf G_1 als een tripel $\langle \Sigma, V, R \rangle$.
(ii) Geef een productie die laat zien dat $aabb \in \mathcal{L}(G_1)$.
(iii) Kun je binnen drie minuten een productie geven die laat zien dat $bbaa \in \mathcal{L}(G_1)$?

Opgave 3.H Gegeven is de volgende grammatica voor de taal L_{12}.

$$\begin{aligned} S &\rightarrow aSb \mid A \mid \lambda \\ A &\rightarrow aAbb \mid abb \end{aligned}$$

De hulpsymbolen zijn dit keer S en A en $\Sigma = \{a, b\}$.
(i) Geef de productie van abb en van $aabb$.
(ii) Welke woorden zitten in L_{12}?

Met enige ervaring is het wel in te zien welke woorden in de taal L_{12} zitten. Maar het is niet eenvoudig om ook echt te *bewijzen* dat dit ze ook precies allemaal zijn. Dit kan (meestal) wel, afhankelijk van de moeilijkheid van de grammatica, maar dat voert hier te ver. Een iets eenvoudigere vraag is de volgende.

Laat zien dat aab niet in L_{12} zit.

Hoe zou je dat aanpakken? Om te laten zien dat een woord *wel* geproduceerd kan worden door een grammatica moet je gewoon op zoek naar een productie, en als die er is vind je die (meestal) wel. Maar hoe laat je zien dat een woord niet geproduceerd kan worden?

In de informatica is daarvoor het begrip *invariant* geïntroduceerd.

Definitie 3.21 Een invariant van G is een eigenschap P die geldt voor alle woorden die geproduceerd kunnen worden door de grammatica G.

Dat P geldt voor alle $w \in \mathcal{L}(G)$ bewijs je door te laten zien dat
(i) P voor S geldt en
(ii) dat P *invariant* is onder de productieregels, oftewel, als P geldt voor een woord v en v' kan geproduceerd worden uit v, dan geldt P ook voor v'.

Opmerking 3.22 Merk op dat dat laatste voor alle woorden v uit Σ^* moet gelden en dus niet alleen voor woorden v die inderdaad in de taal van de grammatica zitten!

Als een woord w niet aan P voldoet, kun je concluderen dat w niet geproduceerd kan worden. Dus om met invarianten te bewijzen dat een woord w niet in een taal zit, moet je het volgende doen.

- Een "goede" eigenschap P verzinnen.
- Laten zien dat S voldoet aan P.
- Laten zien dat P invariant is onder de productieregels.

- Laten zien dat w niet aan P voldoet.

En nu de praktijk.

Voorbeeld 3.23 We willen aantonen dat $aab \notin L_{12}$. Wat is een goede invariant? We nemen

$$P(w) := \text{ het aantal } b\text{'s in } w \geq \text{ het aantal } a\text{'s in } w$$

Dit is een invariant, want

- $P(S)$ geldt.
- Als $P(v)$ en $v \longrightarrow v'$, dan ook $P(v')$. (Ga dit na voor iedere productieregel: of er komt een a en een b bij, of een a en twee b's, of geen a en geen b; in alle gevallen blijft het aantal b's groter of gelijk aan het aantal a's.)

  Hoewel er in definitie 3.21 alleen maar over 'geproduceerd' wordt gesproken, is het voldoende om naar alle producties van één stap te kijken en dat zijn natuurlijk precies de productieregels. (Overtuig jezelf waarom dit voldoende is.)

Dus: $P(w)$ geldt voor alle $w \in L_{12}$. Maar omdat $P(aab)$ duidelijk niet geldt, kunnen we dus concluderen dat $aab \notin L_{12}$.

Opgave 3.I We bekijken nogmaals de taal behorende bij G_1 uit opgave 3.G. Er is daar al opgemerkt dat $bbaa \notin \mathcal{L}(G_1)$.

(i) Is
$$P(w) := w \text{ bevat geen } ba \text{ als deelwoord}$$
een invariant voor G_1 die bewijst dat $bbaa \notin \mathcal{L}(G_1)$?

(ii) Is
$$P(w) := w \text{ bevat geen } ba \text{ en geen } bA \text{ als deelwoord}$$
een invariant voor G_1 die bewijst dat $bbaa \notin \mathcal{L}(G_1)$?

(iii) Is
$$P(w) := w \text{ bevat geen } ba \text{ en geen } bA \text{ en geen } bS \text{ als deelwoord}$$
een invariant voor G_1 die bewijst dat $bbaa \notin \mathcal{L}(G_1)$?

Opgave 3.J Gebruik invarianten om te bewijzen dat:

(i) $bba \notin L_{12}$.

(ii) $aabbb$ niet geproduceerd kan worden via de grammatica voor L_3 die je in opgave 3.F gemaakt hebt.

(iii) $aabbb$ niet geproduceerd kan worden via de grammatica voor L_4 die je in opgave 3.F gemaakt hebt.

Opmerking 3.24 Contextvrije grammatica's heten *contextvrij* omdat in de productieregels aan de linker kant alleen maar hulpsymbolen mogen staan. De regel $Sa \to Sab$ is bijvoorbeeld niet toegestaan. Bij contextvrije grammatica's hoef je, om een productieregel toe te passen, nooit naar de *context* (wat er om het hulpsymbool heen staat) te kijken.

3.4 Rechtslineaire grammatica's

Een bekende beperking van contextvrije grammatica's zijn de *rechtslineaire grammatica's*.

Definitie 3.25 Een *rechtslineaire grammatica* is een contextvrije grammatica waarbij de productieregels de volgende vorm hebben

$$\begin{aligned} X &\rightarrow wY \\ X &\rightarrow w \end{aligned}$$

waarbij X en Y hulpsymbolen zijn en $w \in \Sigma^*$.

Dus in een rechtslineaire grammatica mogen hulpsymbolen in de rechterkant alleen aan het einde staan.

Voorbeeld 3.26 (i) In voorbeeld 3.16 is alleen de grammatica bij L_{10} rechtslineair.

(ii) Soms kunnen we bij een contextvrije grammatica een equivalente rechtslineaire grammatica maken. De volgende rechtslineaire grammatica (over $\Sigma = \{a, b\}$) produceert de taal L_{11} uit voorbeeld 3.16:

$$\begin{aligned} S &\rightarrow aS \mid B \\ B &\rightarrow bB \mid \lambda \end{aligned}$$

Een diepzinnige stelling zegt dat de klasse van talen die geproduceerd kunnen worden met een rechtslineaire grammatica precies de klasse van reguliere talen is.

Stelling 3.27 *Een taal L is regulier dan en slechts dan als er een rechtslineaire grammatica is die hem beschrijft.*

Gevolg 3.28 *Een reguliere taal is altijd contextvrij.*

De stelling bewijzen we niet. Om haar te bewijzen moet je laten zien hoe je bij een reguliere expressie een rechtslineaire grammatica maakt die dezelfde taal produceert. Omgekeerd moet je bij iedere rechtslineaire grammatica een reguliere expressie opschrijven die dezelfde taal beschrijft. Het eerste illustreren we met een voorbeeld.

Voorbeeld 3.29 Beschouw de reguliere expressie

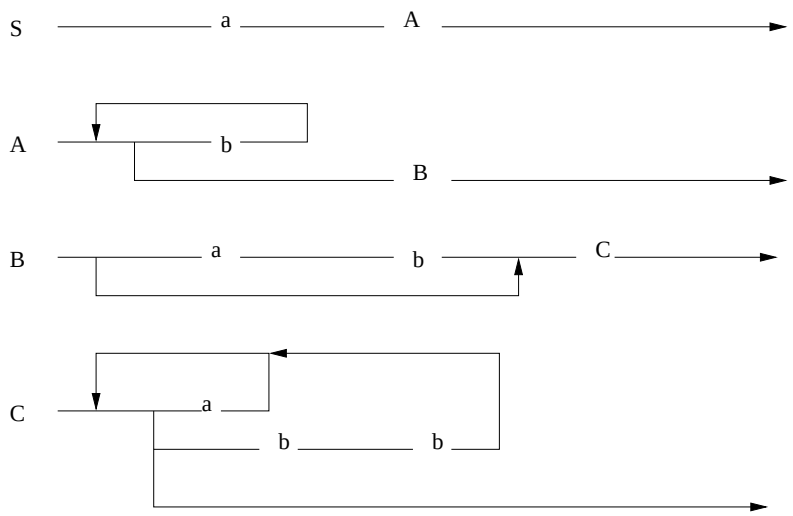
$$ab^*(ab \cup \lambda)(a \cup bb)^*$$

Een rechtslineaire grammatica die dezelfde taal genereert is

$$\begin{aligned} S &\rightarrow aA \\ A &\rightarrow bA \mid B \\ B &\rightarrow abC \mid C \\ C &\rightarrow aC \mid bbC \mid \lambda \end{aligned}$$

Soms worden rechtslineaire grammatica's ook weergegeven met zogenaamde *syntaxdiagrammen*. We laten daarvan hier alleen een voorbeeld zien. In figuur 3.1 staat het diagram dat hoort bij de grammatica uit voorbeeld 3.29.

Een direct gevolg van stelling 3.27 is dat de volgende twee talen (over het alfabet $\{a, b\}$) regulier zijn (zie voorbeeld 3.16): $L_{10} = \{a^n \mid n \text{ is oneven}\}$ en $L_{11} = \{wv \mid w \text{ bevat alleen } a\text{'s}, v \text{ bevat alleen } b\text{'s}\}$. Probeer zelf een reguliere expressie op te schrijven voor L_{10} en ook voor L_{11}.



Figuur 3.1: Syntaxdiagram voor voorbeeld 3.29

Opgave 3.K (i) Bekijk de volgende grammatica over alfabet $\{a, b, c\}$:

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow abS \mid \lambda \\ B &\rightarrow bcS \mid \lambda \end{aligned}$$

Ga na of je de volgende woorden kunt produceren met deze grammatica: *abab*, *bcabbc*, *abba*. Lukt dat, geef dan een productie. Lukt dat niet, geef dan een geschikte invariant waarmee je dit zou kunnen bewijzen.

(ii) Beschrijf de reguliere taal L_{13} die deze grammatica produceert met een reguliere expressie.

(iii) Maak een rechtslineaire grammatica voor de taal L_{14} bestaande uit woorden van de vorm $ab \ldots aba$ (dus a's en b's om en om met vooraan en achteraan een a; zorg dat je ook a krijgt).

Opgave 3.L Geef rechtslineaire grammatica's voor de talen uit opgave 3.D:

(i) $L_6 := \{w \in \{a, b\}^* \mid$ iedere a in w wordt direct gevolgd door een $b\}$,

(ii) $L_7 :=$ de taal van alle goedgevormde integer-expressies. Deze bestaan uit de symbolen $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$, maar ze beginnen nooit met een 0 (behalve het getal 0 zelf natuurlijk) en mogen eventueel worden voorafgegaan door een $+$ of een $-$.

(iii) $L_8 :=$ de taal van alle goedgevormde rekenkundige expressies zonder haakjes. Deze bestaan uit natuurlijke getallen met daartussen de operatoren $+$, $-$ of $\times$, bijvoorbeeld $7 + 3 \times 29 - 78$.

Opgave 3.M Hieronder tonen we een grammatica voor een klein gedeelte van de Engelse taal.

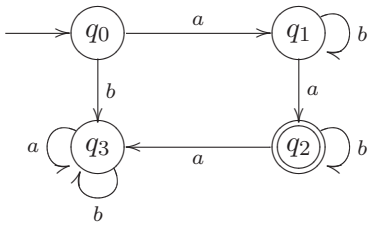
$$
\begin{array}{rcl}
S = \langle \text{zin} \rangle & \to & \langle \text{onderwerpvorm} \rangle \langle \text{werkwoordvorm} \rangle . \\
\langle \text{zin} \rangle & \to & \langle \text{onderwerpvorm} \rangle \langle \text{werkwoordvorm} \rangle \langle \text{lijdendvoorwerpvorm} \rangle . \\
\langle \text{onderwerpvorm} \rangle & \to & \langle \text{eigennaam} \rangle \mid \langle \text{lidwoord} \rangle \langle \text{zelfstandignaamwoord} \rangle \\
\langle \text{eigennaam} \rangle & \to & \text{John} \mid \text{Jill} \\
\langle \text{zelfstandignaamwoord} \rangle & \to & \text{bicycle} \mid \text{mango} \\
\langle \text{lidwoord} \rangle & \to & \text{a} \mid \text{the} \\
\langle \text{werkwoordvorm} \rangle & \to & \langle \text{werkwoord} \rangle \mid \langle \text{bijwoord} \rangle \langle \text{werkwoord} \rangle \\
\langle \text{werkwoord} \rangle & \to & \text{eats} \mid \text{rides} \\
\langle \text{bijwoord} \rangle & \to & \text{slowly} \mid \text{frequently} \\
\langle \text{bijvoeglijknaamwoordlijst} \rangle & \to & \langle \text{bijvoeglijknaamwoord} \rangle \langle \text{bijvoeglijknaamwoordlijst} \rangle \mid \lambda \\
\langle \text{bijvoeglijknaamwoord} \rangle & \to & \text{big} \mid \text{juicy} \mid \text{yellow} \\
\langle \text{lijdendvoorwerpvorm} \rangle & \to & \langle \text{bijvoeglijknaamwoordlijst} \rangle \langle \text{eigennaam} \rangle \\
\langle \text{lijdendvoorwerpvorm} \rangle & \to & \langle \text{lidwoord} \rangle \langle \text{bijvoeglijknaamwoordlijst} \rangle \langle \text{zelfstandignaamwoord} \rangle
\end{array}
$$

(i) Is deze grammatica rechtslineair?
(ii) Laat zien hoe je de volgende zin produceert: *Jill frequently eats a big juicy yellow mango.*
(iii) Maak zelf andere zinnetjes.

3.5 Automaten

In het begin van dit blok hebben we al aangegeven dat talen ook via automaten kunnen worden geformaliseerd. Om dat verband te leggen, gaan we nu eerst het begrip automaat introduceren.

In de informatica worden ook machines bestudeerd. Dat gebeurt door naar de architectuur van computers zelf te kijken, maar ook door geïdealiseerde, abstracte machines te bestuderen. Het voordeel van het bestuderen van zo'n geïdealiseerd, abstract *machinemodel* is dat het eenvoudiger is om aan zo'n model allerlei eigenschappen te bestuderen. Bekende abstracte machines zijn de *eindige automaten*. Deze eindige automaten hebben veel meer toepassingen dan alleen als model voor simpele berekeningen. Zo worden ook processen met eindige automaten (of kleine uitbreidingen daarvan) gemodelleerd. Laten we eerst kijken wat een eindige automaat is en hoe je ermee 'rekent'. Hier is een voorbeeld van een eindige automaat.

M_0 : 

(3)

Een automaat is een zogenaamde gerichte graph, waarbij de lijnen *pijlen* zijn en een *label* hebben. (Zie sectie 4.1 voor een formele beschrijving van graphen.) De punten in deze graph noemen we de *toestanden* van de automaat: q_0, q_1, q_2 en q_3. Het is gebruikelijk om deze toestanden als een rondje te schrijven met de naam van de toestand erin. Er zijn twee toestanden van een speciale soort:

1. De *begintoestand*. Deze wordt aangegeven door er een pijl uit het niets naar toe te tekenen. Een automaat heeft altijd *precies één* begintoestand, hierboven dus q_0.

2. De *eindtoestanden*. Deze worden aangegeven door een dubbele rand. Een automaat heeft altijd *minimaal één* eindtoestand, hierboven is er (toevallig) precies één, q_2. (De begintoestand kan rustig ook een eindtoestand zijn!)

We kunnen een automaat zien als een (heel simpel) computertje dat voor ons kan rekenen. Dat rekenen gaat heel simpel: we kunnen er een woord in stoppen (in bovenstaande automaat woorden over het alfabet $\{a, b\}$) en na een aantal stappen *eindigt* de automaat. Dan zijn we óf in een eindtoestand óf in een niet-eindtoestand. In het eerste geval zeggen we dat het woord *geaccepteerd* wordt, in het tweede geval wordt het woord niet geaccepteerd of ook wel *verworpen*.

Voorbeeld 3.30 In de bovenstaande automaat wordt het woord aa geaccepteerd: begin in q_0 en lees het eerste teken a. Dan komen we in toestand q_1 en we hebben nog a over. Lees ook deze a en we komen in q_2. Er is nu geen input meer: de berekening stopt in een eindtoestand, dus aa is geaccepteerd.
Ga zelf na dat $abbba$ en $ababb$ geaccepteerd worden en dat $ababab$ en bab niet geaccepteerd worden.

Voor we verder gaan geven we een precieze definitie van het begrip eindige automaat.

Definitie 3.31 Een *eindige automaat* bestaat uit de volgende vijf componenten

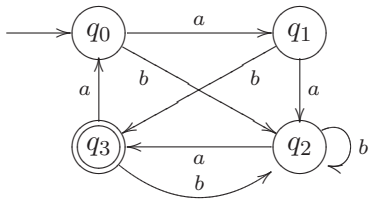
1. Een eindige verzameling Σ, het input alfabet of de verzameling van *atomaire acties*,
2. Een eindige verzameling Q, de *toestanden*,
3. Een speciale toestand $q_0 \in Q$, de *begintoestand*,
4. Een niet-lege verzameling speciale toestanden $F \subseteq Q$, de *eindtoestanden*,
5. Een *overgangsfunctie* δ, die bij iedere toestand q en actie a aangeeft wat de nieuwe toestand q' is. (Dit zijn de gelabelde pijlen in de automaat in (3).

Een eindige automaat noemt men ook wel een *DFA*, naar het Engels': "Deterministic Finite Automaton". Een eindige automaat wordt vaak als het vijftal $M := \langle \Sigma, Q, q_0, F, \delta \rangle$ geschreven.

We zullen, als we een automaat moeten definiëren, meestal gewoon een diagram tekenen, net als we in het voorbeeld boven gedaan hebben.

Opgave 3.N Bekijk de volgende automaat M_1.

M_1 :



In deze automaat is $Q = \{q_0, q_1, q_2, q_3\}$, $F = \{q_3\}$ en $\Sigma = \{a, b\}$.

(i) Ga voor de volgende woorden na of ze geaccepteerd worden: $abaab$, $aaaba$, bab, λ en $aabbab$.

(ii) Gelden de volgende uitspraken? Geef steeds een tegenvoorbeeld of een bewijs.
   (1) Als w geaccepteerd wordt, dan wordt $wabba$ ook geaccepteerd.
   (2) Als w geaccepteerd wordt, dan wordt wab niet geaccepteerd.
   (3) Als w niet geaccepteerd wordt, dan wordt waa ook niet geaccepteerd.
   (4) Als w niet geaccepteerd wordt, dan wordt wbb ook niet geaccepteerd.

3.6 Automaten en talen

We kunnen de Σ uit de eindige automaat opvatten als de verzameling van "atomaire acties" (die ons van één toestand overvoeren in de volgende), maar ook als de symbolen van een alfabet, zoals we boven al gedaan hebben. Als we aan Σ denken als een alfabet, kunnen we een automaat zien als een *taalherkenner*. Op deze manier hoort er bij iedere eindige automaat een taal, namelijk de taal bestaande uit precies die woorden die de automaat accepteert.

Definitie 3.32 Bij een eindige automaat $M := \langle \Sigma, Q, q_0, F, \delta \rangle$ definiëren we de *taal van* M, $L(M)$ als volgt

$$L(M) := \{w \in \Sigma^* \mid w \text{ wordt geaccepteerd door } M\}.$$

Dus: $w \in L(M)$ dan en slechts dan als de automaat M stopt in een eindtoestand op invoer w.

Laten we eens kijken welke taal de automaat M_0 uit het voorbeeld accepteert. De volgende woorden worden geaccepteerd:

- aa wordt geaccepteerd,
- awa wordt geaccepteerd waarbij w een willekeurig lange rij b-tjes is,
- $awav$ wordt geaccepteerd waarbij w en v beide willekeurig lange rijen b-tjes zijn.

Als we toestand q_2 "voorbij zijn" komen we nooit meer in een eindtoestand, dus wat hierboven staat is alles. We kunnen dit samenvatten als

$$L(M_0) = \{ab^n ab^m \mid n, m \geq 0\}$$

Op basis van onze kennis van talen, opgedaan in sectie 3.1, zien we meteen de reguliere expressie die bij deze taal hoort (en we kunnen dus concluderen dat $L(M_0)$ regulier is):

$$L(M_0) = \mathcal{L}(ab^*ab^*).$$

We kunnen hetzelfde met de taal van M_1 proberen, maar dat is lastiger:

- ab wordt geaccepteerd,
- ba wordt geaccepteerd,
- aaa wordt geaccepteerd,
- aab^ka wordt geaccepteerd waarbij $k \geq 0$,
- bb^ka wordt geaccepteerd waarbij $k \geq 0$,
- $aab^ka(ba)^l$ wordt geaccepteerd waarbij $k \geq 0$, $l \geq 0$,
- $bb^ka(ba)^l$ wordt geaccepteerd waarbij $k \geq 0$, $l \geq 0$,

maar we hebben nog steeds niet alles, want als we q_2 "voorbij zijn" kunnen we via een lus weer in q_2 terechtkomen. Kunnen we toch meer grip krijgen op de taal van deze automaat? Ja, dat kunnen we door bij deze automaat een *grammatica* te maken die dezelfde taal *genereert*. Dit gaat als volgt

1. Maak bij iedere toestand q_i een hulpsymbool X_i, zodat bij q_0 het startsymbool S hoort.
2. Als $q_i \xrightarrow{a} q_j$ in de automaat, voeg dan aan de grammatica de regel $X_i \to aX_j$ toe.
3. Als $q_i \in F$, voeg dan aan de grammatica de regel $X_i \to \lambda$ toe.

Voor de automaat M_1 krijgen we dan de volgende grammatica G_2, waarbij $\Sigma = \{a, b\}$.

$$\begin{aligned} S &\to bB \mid aA \\ A &\to aB \mid bC \\ B &\to bB \mid aC \\ C &\to bB \mid aS \mid \lambda \end{aligned}$$

Merk op dat deze grammatica *rechtslineair* is en dat ook de taal $L(M_1)$ dus *regulier* is.

Deze schrijfwijze van de taal $L(M_1)$ met behulp van een grammatica is interessant, al was het alleen al omdat het een andere beschrijving van dezelfde taal oplevert. Maar het kan nog meer opleveren, namelijk als we in de grammatica symbolen gaan substitueren: vul eerst voor A in het rechterlid alle mogelijkheden voor A in: aB en bC. Doe dan hetzelfde met C: vul in de rechterleden voor C overal in bB, aS en λ. Dit levert de volgende grammatica G_3 op, die dezelfde taal als boven genereert. Dit is weer een rechtslineaire grammatica (zie de definitie).

$$\begin{aligned} S &\to bB \mid aaB \mid abbB \mid abaS \mid ab \\ B &\to bB \mid abB \mid aaS \mid a \end{aligned}$$

Opgave 3.O Concludeer uit de grammatica G_3 dat
(i) $(aba)^k ab \in L(M_1)$ voor $k \geq 0$,
(ii) $aab^k a \in L(M_1)$ voor $k \geq 0$,
(iii) als $w \in L(M_1)$, dan ook $abaw \in L(M_1)$,
(iv) als $w \in L(M_1)$, dan ook $aaaaw \in L(M_1)$,

Dat we van een eindige automaat een rechtslineaire grammatica kunnen maken, op de manier zoals we boven voor M_1 gedaan hebben is een algemeen feit.

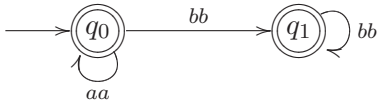
Stelling 3.33 *Bij iedere eindige automaat M kunnen we een rechtslineaire grammatica G maken zodat $\mathcal{L}(G) = L(M)$.*
(De taal die G genereert is dezelfde als de taal die M accepteert.) Bij gevolg is $L(M)$ regulier voor iedere eindige automaat M.

Opgave 3.P Maak een rechtslineaire grammatica bij de eindige automaat M_0, zoals boven gedaan voor M_1. Verklein deze grammatica door overbodige productieregels en hulpsymbolen weg te laten.

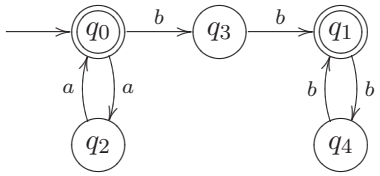
We kunnen ook de andere kant op: bij een rechtslineaire grammatica een eindige automaat maken. Bekijk de volgende rechtslineaire grammatica G_4.

$$\begin{aligned} S &\to aaS \mid bbB \mid \lambda \\ B &\to bbB \mid \lambda \end{aligned}$$

We creëren nu eerst voor ieder hulpsymbool een toestand en maken transities (pijlen) gelabeld met woorden (in plaats van alleen letters). De hulpsymbolen die naar λ kunnen gaan worden eindtoestand en S wordt uiteraard begintoestand.

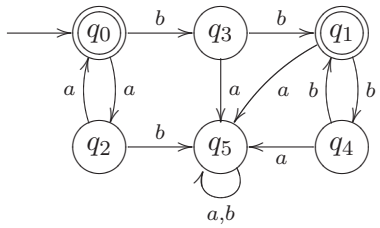


Vervolgens voegen we extra toestanden toe waarmee we de woorden bij de pijlen "ophakken" in letters. We krijgen dan



Dit is nog (net) geen eindige automaat, want bij een eindige automaat eisen we dat er *vanuit iedere toestand voor iedere letter* een stap mogelijk is en dat is nu niet zo. Om er een eindige automaat van te maken moeten we een soort "put" toevoegen waar alle nog niet opgegeven transities naar toe gaan en waar je niet meer uitkomt. Dat wordt q_5 en het levert uiteindelijk de volgende eindige automaat op.

M_2 :

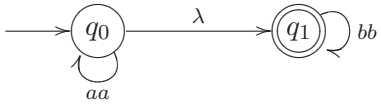


Opmerking 3.34 In de bovenstaande automaat hebben we één pijl (van q_5 naar zichzelf twee labels gegeven. Dit is hetzelfde als twee gelabelde pijlen, maar de pijlen-spaghetti wordt zo iets minder. We zouden ook toe kunnen staan dat we niet alle pijlen hoeven toe te voegen en dan zouden we de "put" dus weg kunnen laten. Maar dat doen we toch niet, want bijvoorbeeld een woord *bba* moet *niet* door M_2 geaccepteerd worden: dat is duidelijk in de tweede versie van M_2, terwijl in de eerste versie de automaat stopt in toestand q_1, een eindtoestand ...

Deze procedure om een automaat uit een rechtslineaire grammatica te maken werkt heel algemeen, behalve wanneer we een productieregel van de vorm $S \to B$ hebben, dus waar het woord dat vòòr het hulpsymbool staat λ is. Bekijk de volgende grammatica G_5.

$$\begin{aligned} S &\to aaS \mid B \\ B &\to bbB \mid \lambda \end{aligned}$$

Als we hier de eerste stap zetten om een automaat te maken (op de manier die we boven gevolgd hebben) krijgen we het volgende.



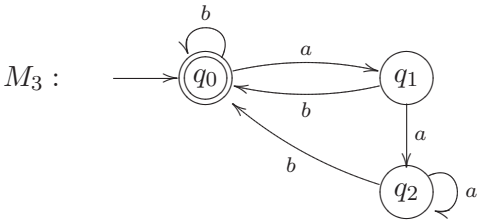
We hebben dus λ, het lege woord, bij de pijl en dit woord kunnen we uiteraard niet "ophakken", dus hier werkt onze techniek niet. De oplossing is om eerst een *equivalente* rechtslineaire grammatica te verzinnen waar producties van de vorm $S \to B$ niet voorkomen. Dat kan (altijd), maar we laten hier niet zien hoe je het in zijn algemeen doet. Voor G_5 zou je uitkomen op de grammatica G_4, dus de automaat M_2 accepteert dezelfde taal als grammatica G_5 genereert. (En G_4 en G_5 genereren dezelfde taal: ga dat voor jezelf na!)

Stelling 3.35 *Bij iedere rechtslineaire grammatica G kunnen we een eindige automaat M maken zodat $L(M) = \mathcal{L}(G)$.*

Opgave 3.Q Maak een eindige automaat die de taal van de volgende grammatica accepteert.

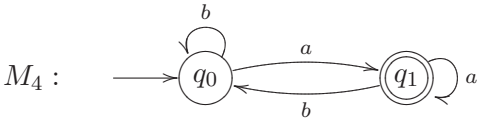
$$\begin{aligned} S &\to abS \mid aA \mid bB \\ A &\to aA \mid \lambda \\ B &\to bB \mid \lambda \end{aligned}$$

Opgave 3.R Bekijk de eindige automaat M_3:



Maak een rechtslineaire grammatica die $L(M_3)$ genereert.

Opgave 3.S Bekijk de eindige automaat M_4:



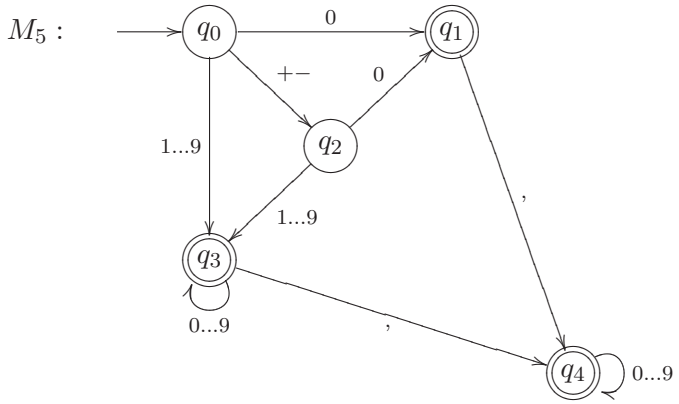
(i) Maak een rechtslineaire grammatica die $L(M_4)$ genereert.
(ii) Geef een (eenvoudige) beschrijving van $L(M_4)$.
(iii) Als we alle toestanden eindtoestand maken, welke taal accepteert M_4 dan?
(iv) Als we de eindtoestanden en de niet-eindtoestanden omwisselen, welke taal accepteert M_4 dan?

Opgave 3.T Maak een eindige automaat die de taal $L_{15} := \{(ab)^k(aba)^l \mid k, l \geq 0\}$ over $\Sigma = \{a, b\}$ accepteert.

Opgave 3.U Maak een eindige automaat die de taal $L_{16} := \{(ab)^k x (ab)^l \mid x \in \{a, b\}, k, l \geq 0\}$ over $\Sigma = \{a, b\}$ accepteert.

Opgave 3.V Bekijk de automaat M_5 die (input) getallen in 'normale' decimale notatie accepteert. Zo'n getal mag eventueel beginnen met een $+$ of een $-$, maar niet met een 0, behalve

natuurlijk als het van de vorm $0,5$ is, want dan is het weer wel goed. In het diagram staat $0 \ldots 9$ voor één van de cijfers uit 0 t/m 9 enz.



(i) Vind je de automaat M_5 goed? Worden alle 'juiste' getallen geaccepteerd en alle 'foute' verworpen?

(ii) Pas M_5 aan als je het er niet mee eens bent.

(iii) Geef een rechtslineaire grammatica die dezelfde taal als M_5 genereert.

3.7 Belangrijke begrippen

4 Discrete wiskunde

Dit blok bestaat uit een serie kleine onderwerpen die wij gevangen hebben onder de rubriek *discrete wiskunde*. Kenmerk van discrete wiskunde is dat alles te tellen valt met gehele getallen: het aantal punten in een graaf, het aantal stappen in een recursieve berekening, het aantal manieren om over een rooster van een bepaald punt naar een ander punt te lopen. Je hebt nooit te maken met continuïteitsproblemen waarbij er tussen twee punten altijd oneindig veel andere punten zitten.

Zie voor meer informatie bijvoorbeeld [1] en [4].

4.1 Grafen

Dit hoofdstuk is een korte inleiding in de graaftheorie. Grafen kom je op allerlei plaatsen tegen bijvoorbeeld bij talen, netwerken, datastructuren, elektrische circuits, transportproblemen en stroomschema's.

Intuïtief bestaat een graaf uit een verzameling P van punten en een verzameling L van lijnen tussen punten.

Voorbeeld 4.1 Twee voorbeelden:

Natuurlijk zijn er nu allerlei knagende onzekerheden, zoals: 'Wanneer zijn twee grafen gelijk?', 'Moeten die punten in een plat vlak liggen?', 'Mogen de lijnen elkaar snijden?'. Al je twijfels verdwijnen als sneeuw voor de zon, dankzij de volgende formele definitie:

Definitie 4.2 Een *graaf* is een paar $\langle P, L\rangle$ waarbij P een verzameling is en L een verzameling van uit twee elementen bestaande deelverzamelingen van P. De elementen van P noemen we *punten*, de elementen van L noemen we *lijnen*; een lijn noteren we niet als $\{a, b\}$ maar als (a, b). De lijn (a, b) is dus gelijk aan de lijn (b, a).

Voorbeeld 4.3 De graaf G_1 uit voorbeeld 4.1 is dus $\langle P, L\rangle$ met $P = \{1, 2, 3, 4\}$ en $L = \{(1, 4), (2, 3), (2, 4)\}$. En $G_2 = \langle P, L\rangle$ met $P = \{a, b, c, d\}$ en $L = \{(a, c), (a, d), (c, d)\}$.

Opmerking 4.4 Opvallend aan definitie 4.2 is dat P best de lege verzameling mag zijn. Merk verder op dat de lijnen in onze grafen geen *richting* hebben: het zijn geen *pijlen*. Dit type graaf heet dan ook *ongericht*. (De automaten uit blok 3 waren gerichte grafen.) Tevens volgt uit definitie 4.2 dat het niet mogelijk is dat een punt via een lijn met zichzelf is verbonden, zoals dat bij automaten (zie sectie 3.5) bijvoorbeeld wel mag. Lijnen zijn immers verzamelingen met twee elementen er in. Maar een lijn van p naar p zou dan dus $(p, p) = \{p, p\} = \{p\}$ zijn en die verzameling heeft maar één element. Verder is het niet mogelijk dat er tussen twee punten p en q meerdere lijnen zijn. Je zou de definitie van 'graaf' kunnen veranderen en dit wel toestaan, maar het blijkt dat we met definitie 4.2 al genoeg stof tot nadenken hebben en al veel interessante eigenschappen kunnen beschrijven en bestuderen.

Definitie 4.5 Laat $G = \langle P, L \rangle$ een graaf en laat $p, q \in P$.

- Een *buur* van p is een punt x met $(p, x) \in L$.
- De *graad* (of: *valentie*) van p is het aantal buren van p.
- Een *pad* van p naar q is een rijtje van verschillende lijnen $(x_0, x_1), (x_1, x_2), \ldots, (x_{n-1}, x_n)$ met $n > 0$, $x_0 = p$ en $x_n = q$. Een kortere notatie voor dit pad: $x_0 \to x_1 \to x_2 \to \cdots \to x_n$.
- Met 'G is *samenhangend*' bedoelen we: tussen ieder tweetal punten bestaat een pad.
- Een *component* van G is een zo groot mogelijk samenhangend deel van een graaf.
- Een *cykel* is een pad van een punt naar zichzelf. Een *circuit* is hetzelfde als een cykel.
- Met 'G is *planair*' bedoelen we: je kunt G in het platte vlak zó tekenen dat de (gebogen) lijnen elkaar niet snijden.
- Met 'G is een *boom*' bedoelen we: G is samenhangend en bevat geen cykels.

De vraag of een graaf planair is, is bijvoorbeeld belangrijk als de graaf een elektrisch circuit voorstelt dat we op een chip willen branden.

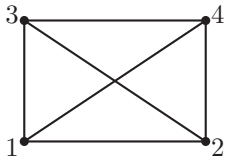
Opgave 4.A Bewijs dat er in een boom van een punt p naar een ander punt q precies één pad bestaat.

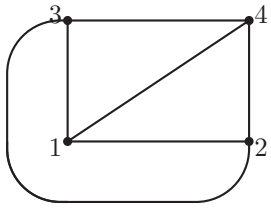
Opgave 4.B Een *brug* in een graaf G is een lijn l waarvoor geldt: als je l uit G weglaat, wordt het aantal componenten verhoogd. Bewijs dat in een boom iedere lijn een brug is.

Voorbeeld 4.6 Grafen komen op verschillende plaatsen 'in het wild' voor. Hier enkele voorbeelden.

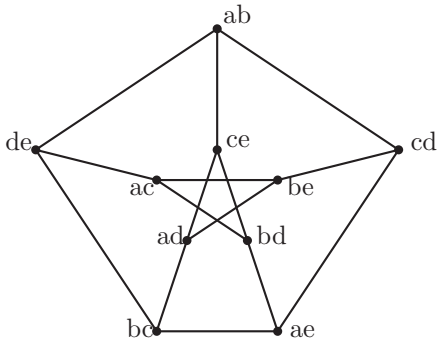
(i) De '*landgraaf*' is $\langle P, L \rangle$ waarbij P de verzameling van alle landen van de wereld is en L de relatie 'grenst aan'. De buren van Nederland zijn dan Duitsland en België. De graad van Nederland is 2. Een pad van Nederland naar Spanje is bijvoorbeeld Nederland $\to$ Duitsland $\to$ Frankrijk $\to$ Spanje. De landgraaf is niet samenhangend. $\{\text{Engeland}, \text{Schotland}\}$ is een component. Nederland $\to$ Duitsland $\to$ België $\to$ Nederland is een cykel. De landgraaf is planair.

(ii) $K_4 = \langle \{1, 2, 3, 4\} \ , \ \{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)\} \rangle$ (de *volledige graaf* met vier punten). Algemeen: K_n is de graaf $\langle \{1, \ldots, n\} \, , \, \{(p, q) \mid 1 \leq p < q \leq n\} \rangle$. De graaf K_4 is planair. Als je dat wilt inzien, moet je niet naar de linker maar naar de rechter tekening van K_4 kijken:





(iii) De *Petersen-graaf* is $\langle P, L\rangle$ met $\begin{cases} P = \{\text{ab, ac, ad, ae, bc, bd, be, cd, ce, de}\} \\ L = \{(p,q) \mid p \text{ en } q \text{ hebben geen letter gemeen}\} \end{cases}$



Je kunt laten zien dat de Petersen-graaf niet planair is.

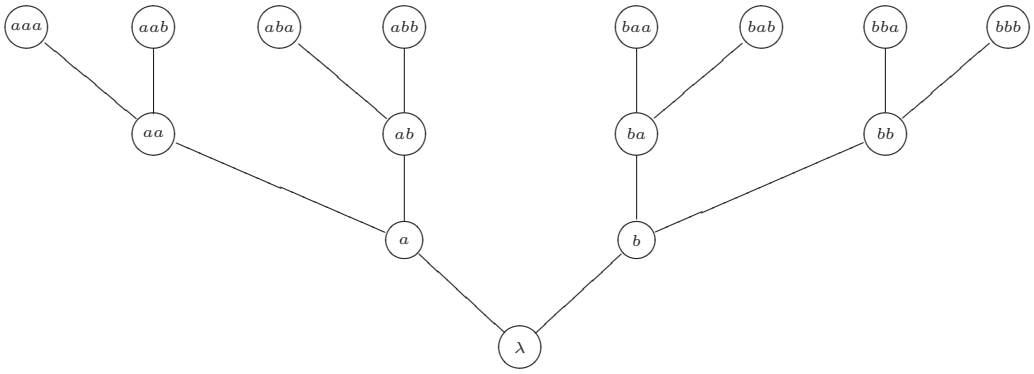
(iv) Bij een taal die gegeven is door middel van een inductieve taaldefinitie, kunnen we een graaf maken. Bekijk bijvoorbeeld eens de taal die als volgt geproduceerd wordt:

axioma	λ
regel	$x \rightarrow xa$ $y \rightarrow yb$

Bij deze taal kunnen we een graaf maken door

- eerst de woorden op te schrijven die er in zitten op grond van het axioma (dat zijn de eerste punten van de graaf),
- dan stap voor stap woorden (en dus punten) toe te voegen die er op grond van de toepassing van een regel inzitten, en een lijn te maken naar het woord van waaruit we dit nieuwe woord hebben kunnen maken.

We krijgen dan de volgende (onvolledige) graaf:

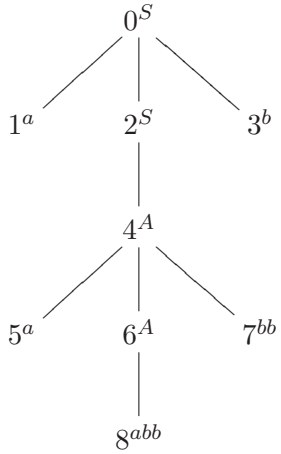


Deze graaf is een boom.

(v) Bij een contextvrije grammatica en een woord dat door deze grammatica geproduceerd wordt, kun je een *parseerboom* maken. Een parseerboom geeft grafisch weer hoe het woord geproduceerd wordt. Bekijk de volgende contextvrije grammatica uit opgave 3.H.

$$\begin{aligned} S &\rightarrow aSb \mid A \mid \lambda \\ A &\rightarrow aAbb \mid abb \end{aligned}$$

Het woord *aaabbbbb* wordt geproduceerd door deze grammatica, op de volgende manier: $S \to aSb \to aAb \to aaAbbb \to aaabbbbb$. Hieronder staat de parseerboom van deze productie.



Dus als we $S \to aSb$ doen, geven we het punt met label S drie onderburen met labels a, S en b en als we $A \to aAbb$ doen, geven we het punt met label A drie onderburen met labels a, A en bb. Als je de (labels van de) bladeren van de boom van links naar rechts achter elkaar zet krijg je het woord dat bij deze parseerboom hoort.

Opmerking 4.7 Informatici tekenen een boom bijna altijd 'op de kop', dus met de 'wortel' boven en de takken naar beneden gericht. Wiskundigen tekenen een boom altijd met de wortel onder (zie vorige voorbeeld). Bij een parseerboom worden doorgaans alleen de labels opgeschreven bij de punten (en laat men de punten dus ongenummerd).

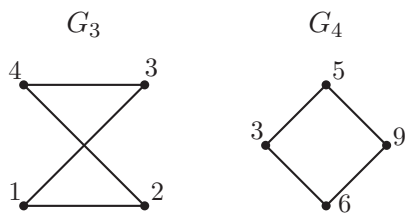
Opgave 4.C Bewijs dat voor alle $n \geq 1$ geldt dat de K_n precies $\frac{1}{2}n(n-1)$ lijnen heeft.

4.2 Isomorfie van grafen

Definitie 4.8 Een afbeelding f van een verzameling A naar een verzameling B heet een *bijectie* als ieder element in B precies één origineel heeft.

Definitie 4.9 We noemen twee grafen $\langle P, L\rangle$ en $\langle P', L'\rangle$ *isomorf* als er een bijectie $\varphi : P \to P'$ bestaat zodat voor alle x, y in P, $(x, y) \in L$ dan en slechts dan als $(\varphi(x), \varphi(y)) \in L'$. Anders gezegd: twee grafen zijn isomorf als ze, afgezien van de namen van de punten, hetzelfde zijn. Zo'n 'eigenschappen behoudende' bijectie φ wordt dan ook een *isomorfisme* genoemd.

Voorbeeld 4.10 Bekijk de volgende twee grafen.

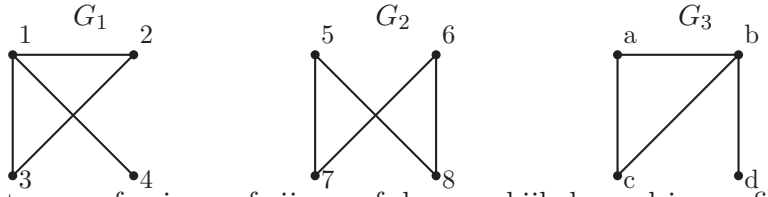


De grafen G_3 en G_4 zijn isomorf, want er bestaat een isomorfisme φ:

$$\begin{aligned} 1 &\mapsto 6 \\ 2 &\mapsto 9 \\ 3 &\mapsto 3 \\ 4 &\mapsto 5 \end{aligned}$$

Isomorfe grafen zijn 'hetzelfde' als we alleen geïnteresseerd zijn in graaf-theoretische eigenschappen. Bijvoorbeeld: Als G en G' isomorf zijn en G is samenhangend, dan is G' dat ook.

Opgave 4.D Ga na welke van de onderstaande grafen isomorf zijn:

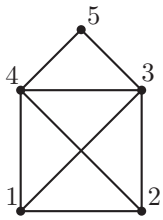


Als twee grafen isomorf zijn, geef dan een bijbehorend isomorfisme. En als twee grafen niet isomorf zijn, leg dan uit waarom zo'n isomorfisme niet kan bestaan.

4.3 Euler en Hamilton

Definitie 4.11 Een *Euler-pad* in een graaf $\langle P, L\rangle$ is een pad waarin iedere lijn uit L precies één keer voorkomt. Een *Euler-circuit* of *Euler-cykel* is een cykel waarin iedere lijn uit L precies één keer voorkomt.

Voorbeeld 4.12 Bekijk de volgende graaf:



Het pad $1 \to 4 \to 5 \to 3 \to 1 \to 2 \to 4 \to 3 \to 2$ is een Euler-pad. Omdat punt 1 graad 3 heeft, is er geen Euler-circuit. Dat kun je inzien door te kijken naar het aantal keren dat je bij een cykel door het punt 1 loopt: is dit aantal 1, dan mis je één van de drie bij punt 1 samenkomende lijntjes, en is dit aantal 2, dan doorloop je minstens één van deze lijntjes dubbel.

Als je deze redenering iets algemener opschrijft, staat er een bewijs van de volgende eenvoudige stelling:

Stelling 4.13 (Euler) *In een samenhangende graaf met minimaal twee punten geldt:*

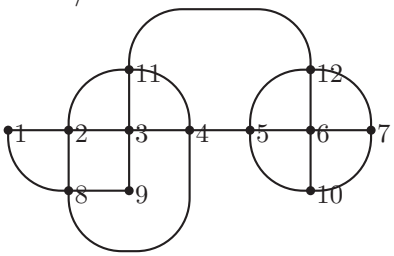
1. *Er bestaat een Euler-circuit dan en slechts dan als ieder punt een even graad heeft.*

2. *Er bestaat een Euler-pad dan en slechts dan als er hoogstens twee punten van oneven graad zijn.*

Euler-circuits zijn bijvoorbeeld van belang voor de krantenbezorger of de wijkagent, die iedere straat van zijn wijk precies één keer wil doorlopen en bij zijn uitgangspunt wil terugkeren. Een heel ander probleem heeft de 'travelling salesman', die iedere klant (of iedere stad) precies één keer wil bezoeken en daarna weer naar huis wil. Hij is gebaat bij een 'Hamilton-circuit':

Definitie 4.14 Een *Hamilton-pad* in een graaf $\langle P, L\rangle$ is een pad waarin je ieder punt van P precies één keer ontmoet. Een *Hamilton-circuit* of *Hamilton-cykel* is een cykel waarin ieder punt precies één keer optreedt (afgezien van het beginpunt dat natuurlijk gelijk is aan het eindpunt).

Opgave 4.E Hier is een plattegrond G van een dorpje, waarbij de straten aangegeven worden door lijntjes. Op ieder hoekpunt bevindt zich een kroeg. De kroegen zijn aangegeven door punten, genummerd van 1 t/m 12:



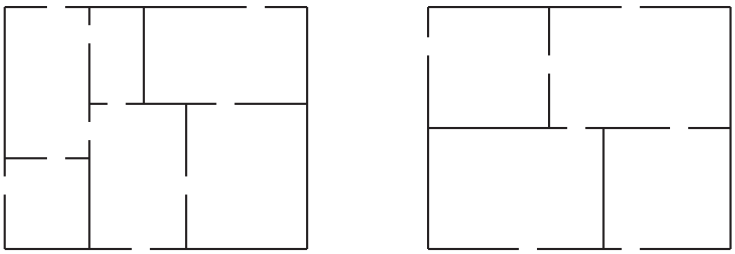
Formuleer de volgende vragen in termen van Hamilton- of Euler-circuits/paden en beantwoord ze:

(i) Is het mogelijk, een wandeling te maken waarbij je iedere straat precies één keer doorloopt? Zo ja, geef dan zo'n wandeling. Zo nee, leg uit waarom niet.

(ii) Is het mogelijk, een wandeling te maken waarbij je iedere straat precies één keer doorloopt en bovendien begint en eindigt bij kroeg 3? Zo ja, geef dan zo'n wandeling. Zo nee, leg uit waarom niet.

(iii) Bestaat er een kroegentocht waarin iedere kroeg precies één keer voorkomt? Zo ja, geef dan zo'n tocht. Zo nee, leg uit waarom niet.

Opgave 4.F Hieronder vind je twee plattegronden van huizen.



(i) Teken bij elke plattegrond de bijbehorende graaf, waarbij je de vertrekken (inclusief het buitengebied) door punten voorstelt en de deuren door lijnen. Bedenk dat punten in een graaf een naam moeten hebben!

(ii) Zoek voor elk huis uit of het mogelijk is een rondwandelingetje te maken waarin je iedere deur precies één keer gebruikt en bij je uitgangspunt terugkeert. Verklaar je antwoord.

(iii) Zoek voor elk huis uit of het mogelijk is een rondwandelingetje te maken waarin je ieder vertrek (en de tuin) precies één keer bezoekt en bij je uitgangspunt terugkeert. Verklaar je antwoord.

Opgave 4.G Welke van de volgende grafen hebben een Hamilton-circuit? Verklaar je antwoord.

Opgave 4.H Zij G de kubus-graaf, met als P de verzameling van de acht hoekpunten, en als L de verzameling van de twaalf ribben.

(i) Is G planair?
(ii) Heeft G een Hamilton-circuit?
(iii) Heeft G een Euler-circuit?

Verklaar telkens je antwoord!

Opgave 4.I Laat zien dat de Petersen-graaf een Hamilton-pad heeft, maar geen Hamilton-cykel.

4.4 Kleuringen

Definitie 4.15 Een graaf $\langle P, L\rangle$ heet *bipartite* als P te schrijven is als $P_1 \cup P_2$ zó dat iedere lijn van een punt uit P_1 naar een punt uit P_2 loopt. Anders gezegd: je kunt de punten zó blauw en rood kleuren dat iedere lijn een blauw en een rood uiteinde heeft.

Voorbeeld 4.16 Twee bipartite grafen:

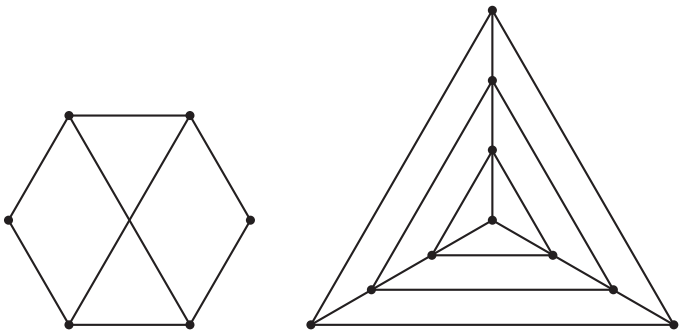
Voorbeeld 4.17 De *volledige bipartite graaf* met m rode en n blauwe punten, waarbij ieder rood punt met ieder blauw punt verbonden is, wordt $K_{m,n}$ genoemd.

$K_{2,3}$ $\qquad$ $K_{3,3}$

Definitie 4.18 Een *puntkleuring* van een graaf $\langle P, L\rangle$ is een functie $f : P \to \{1, \ldots, n\}$ zodat als (p, q) een lijn is, dan $f(p) \neq f(q)$. Dus elk punt krijgt één der n kleuren, maar buren krijgen niet dezelfde kleur.
Het *kleurgetal* (of: *chromatisch getal*) van de graaf is de kleinste n waarvoor dit mogelijk is.

Voorbeeld 4.19 Bipartite grafen zijn dus de grafen met kleurgetal 1 of 2.

Opgave 4.J Bepaal het kleurgetal van de volgende grafen.



Opgave 4.K Laat zien dat als een bipartite graaf een Hamilton-pad toelaat het aantal rode en het aantal blauwe punten hoogstens één verschilt.

Opgave 4.L Op de Volksuniversiteit worden er heel wat vreemde talen aangeboden: Arabisch, Bulgaars, Chinees, Deens, Egyptisch, Fries en Grieks. Bij het maken van het rooster voor de cursussen, moet de roostermaker met twee zaken rekeninghouden:

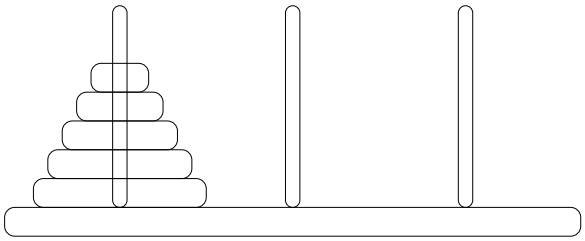
- Het gebouw met tien lokalen kan alleen in zijn geheel gehuurd worden, dus hoe meer cursussen er parallel gegeven kunnen worden, hoe goedkoper het voor de Volksuniversiteit is.
- Sommige studenten hebben zich voor verschillende cursussen ingeschreven en die betreffende talen mogen dus niet tegelijk gegeven worden. In onderstaande tabel betekent een $*$ op een bepaalde positie, dat de taal uit die rij niet mag samen vallen met de taal uit die kolom.

	A	B	C	D	E	F	G
A		*	*	*			*
B	*		*	*	*		*
C	*	*		*		*	
D	*	*	*			*	
E		*					
F			*	*			*
G	*	*				*	

Hoeveel verschillende lesblokken zal de roostermaker gebruiken om aan beide eisen te voldoen?

Een interessante stelling van Appel en Haken, waarvan het in 1975 gevonden bewijs het niveau van dit college helaas ontstijgt:

Stelling 4.20 (Vierkleurenstelling.) *Het kleurgetal van een planaire graaf is hoogstens 4. Anders gezegd: Iedere landkaart kan met hooguit vier kleuren worden ingekleurd, zodanig dat twee aangrenzende landen een verschillende kleur hebben.*



4.5 Torens van Hanoi

De puzzel de 'torens van Hanoi' bestaat uit drie pinnen en een aantal schijven van verschillend formaat. De bedoeling is om alle schijven naar één ander pin te verplaatsen. Iedere zet mogen we een schijf verplaatsen, maar er mag nooit een grote schijf op een kleinere staan.

In het plaatje hebben we vijf schijven. Kun je deze puzzel oplossen?

Na wat puzzelen lukt het je misschien wel, maar dan weet je vaak niet meer hoe je het gedaan hebt. Verder wil je natuurlijk ook weten hoe je het met zeven schijven doet.

Het belangrijke idee bij het oplossen deze puzzel is om het probleem te *generaliseren*. Kunnen we het probleem oplossen bij een willekeurig aantal schijven? Daarnaast moeten we opmerken dat het probleem voor vijf schijven lijkt op het probleem van vier schijven, wat weer lijkt op het probleem voor drie schijven, etc.

Als we het probleem voor vier schijven op kunnen lossen, kunnen we het ook voor vijf schijven. Want laat de grootste schijf maar even liggen. Verplaats de bovenste vier schijven naar pin 2. Dat we dat kunnen volgt uit de aanname dat we het probleem voor vier schijven kunnen oplossen. Pak nu de grootste schijf op en verplaats hem naar pin 3. Verplaats nu de andere schijven van pin 2 naar pin 3. Dat laatste kan weer op grond van onze aanname.

Nu zeg je misschien wat hebben we hieraan? Het probleem voor vier schijven kunnen we ook niet oplossen. Nou ja, dan maken we het nog een beetje makkelijker. Als we het probleem voor drie schijven kunnen oplossen, dan kunnen we het ook voor vier. En als we het probleem voor twee schijven kunnen oplossen, dan kunnen we het ook voor drie. Tenslotte als we het probleem voor één schijf kunnen oplossen, dan kunnen we het ook voor twee.

Opmerking 4.21 Dus *als* we het probleem voor één schijf kunnen oplossen, dan kunnen we het ook voor vijf. Maar het probleem voor één schijf is erg eenvoudig! Dus we kunnen het nu ook voor vijf schijven.

Tenslotte kunnen we nog opmerken dat er niet bijzonders is aan vijf. Het probleem kunnen we nu ook oplossen voor tien schijven of voor ieder ander aantal.

Definitie 4.22 We hebben het vorige probleem *recursief* opgelost. Dat wil zeggen, we hebben het probleem zo ingedeeld dat het uit elkaar valt in een aantal deelproblemen die heel veel op elkaar lijken en zodat moeilijkere problemen altijd zijn terug te brengen tot eenvoudigere problemen. Tenslotte moeten we het allereenvoudigste geval natuurlijk wel op kunnen lossen.

Hoeveel zetten hebben we nu nodig met onze strategie voor de torens van Hanoi? Laat a_n het aantal zetten zijn dat we nodig hebben om de puzzel met n schijven op te lossen. Dus $a_1 = 1$ en $a_2 = 3$. Maar wat is a_5? Het is lastig om dit in één keer in te zien. Maar in ieder geval weten we dat we eerst het probleem met vier schijven moeten oplossen, dan de grote schijf verschuiven en weer het probleem met vier schijven oplossen. Dus $a_5 = a_4 + 1 + a_4 = 2a_4 + 1$.

Omdat er niets bijzonders is aan a_5, kunnen we de algemene *recursieve formule* $a_n = 2a_{n-1}+1$ voor $n \geq 2$ gebruiken. Dus $a_4 = 2a_3 + 1$ en $a_3 = 2a_2 + 1$. Maar a_2 kennen we! En dus $a_3 = 7$, $a_4 = 15$ en $a_5 = 31$. Op dezelfde manier kunnen we iedere a_n uitrekenen. Dat levert de rij van oplossingen $1, 3, 7, 15, 31, 63, 127, \ldots$ op.

Opgave 4.M Definieer met recursie de rij a_n door:

$$\begin{aligned} a_0 &= 3 \\ a_{n+1} &= a_n^2 - 2a_n \text{ voor } n \geq 0 \end{aligned}$$

Geef de waarde van a_5.

Opgave 4.N Definieer met recursie de rij b_n door:

$$\begin{aligned} b_0 &= 4 \\ b_{n+1} &= b_n^2 - 2b_n \text{ voor } n \geq 0 \end{aligned}$$

Geef de waarde van b_5.

Opgave 4.O Bekijk de rij c_n gedefinieerd door:

$$1, 3, 4, 7, 11, 18, 29, 47, 76, 123, 199, 322, 521, 843, 1364, 2207, 3571, 5778, 9349, \ldots$$

Geef een recursieve formule voor c_n.

4.6 Recursief programmeren

Recursie is ook een belangrijke programmeertechniek.

Stel je hebt een programmeertaal waarin je wel kunt optellen, maar nog niet kunt vermenigvuldigen. Hoe definieer je de vermenigvuldiging? Dat kunnen we recursief doen:

$$\begin{aligned} n * 1 &:= n \\ n * (m+1) &:= n * m + n \end{aligned}$$

Nu we kunnen vermenigvuldigen, kunnen we ook machtsverheffen:

$$\begin{aligned} n^1 &:= n \\ n^{(m+1)} &:= n^m * n \end{aligned}$$

In beide gevallen zie je dat de nieuwe operatie telkens kan worden uitgedrukt in iets met de nieuwe operatie op simpelere gevallen, samen met een operatie die we al voor alle gevallen kenden.

Voorbeeld 4.23 In deze paragraaf hebben we het natuurlijk wel over programmeren, maar de code lijkt niet op gebruikelijke imperatieve programmeertalen; het is meer een functionele beschrijving van hoe de vermenigvuldiging en machtsverheffing berekend kunnen worden. Daarom hier een vertaling naar de programmeertaal van `Maple`, een zogenaamd computeralgebrasysteem dat bijzonder handig is om allerlei berekeningen mee te doen. Dit programma `Maple` is vrij beschikbaar op bijvoorbeeld `lilo.science.ru.nl`.

```
mult := proc(n,m)
  if m=1
    then n
    else mult(n,m-1)+n
  end if
end proc;
```

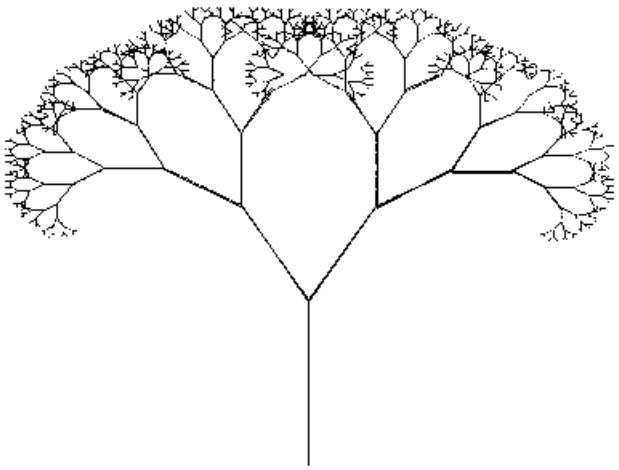
```
pow:=proc(n,m)
  if m=1
    then n
    else mult(pow(n,m-1),n)
  end if
end proc;
```

Opgave 4.P De `Maple`-programma's uit voorbeeld 4.23 zijn niet erg robuust. Indien er voor m een niet natuurlijk getal of 0 wordt gebruikt, zal `Maple` in een oneindige lus terechtkomen. Probeer de programma's zo aan te passen dat er een juiste uitkomst wordt bepaald indien $n, m \in \mathbb{Z}$.

Opmerking 4.24 Op de toetsen en tentamens kan het voorkomen dat we je vragen iets in de vorm van een recursief programma op te schrijven. Hierbij is zogenaamde *pseudocode* genoeg: je hoeft niet aan de specifieke syntax van één of andere taal te voldoen, zolang maar duidelijk is hoe het programma werkt.

4.7 Binaire bomen

Een ander voorbeeld van recursief programmeren komen we tegen als we zogenaamde binaire bomen gaan bestuderen. Bekijk de boom in figuur 4.2.



Figuur 4.2: Een binaire boom

Hoewel deze boom er misschien complex uit ziet, is deze getekend met een simpele recursieve procedure. Het basisinzicht hierbij is dat een boom bestaat uit een stam met daarboven twee andere bomen, links en rechts, die net iets kleiner en iets gedraaid zijn.

Een recursief recept (recursieve functie, recursieve procedure, recursief programma) is eenvoudig te geven:

1. teken een stam
2. teken de linker (sub)boom

3. teken de rechter (sub)boom

Dit recept kunnen we preciezer maken met de volgende functie/procedure:

$$f(n,x,y,\alpha,l) = \begin{cases} \textit{doe niets} & \textit{als } n = 0 \\ 1.\ \textit{teken stam } (\textit{positie } x,y;\ \textit{hoek } \alpha;\ \textit{lengte } l) & \\ 2.\ f(n-1,x_1,y_1,\alpha-30,l/2) & \textit{als } n > 0 \\ 3.\ f(n-1,x_2,y_2,\alpha+30,l/2) & \end{cases}$$

Hierbij is n de hoogte van de boom; x en y de positie van de boom (het startpunt van de stam); α is de hoek van de boom en l is de lengte van de stam. De coördinaten x_1 en y_1 geven het eindpunt aan van de stam en zijn te berekenen vanuit x en y door gebruik te maken van α en l.

De hoogte van de boom, n, is belangrijk in deze recursieve procedure; de andere variabelen laten we even buiten beschouwing.

Het tekenen van een boom met hoogte n kan dus teruggebracht worden tot een simpeler probleem: het tekenen van twee bomen met hoogte $n-1$. Dit recursieve patroon is duidelijk te herkennen wanneer je de stappen volgt van een computer die de procedure uitvoert:

```
[1] teken boom: f(3)
[1.1] teken stam
[1.2] teken linker boom: f(2)
[1.2.1] teken stam
[1.2.2] teken linker boom: f(1)
[1.2.2.1] teken stam
[1.2.2.2] teken linker boom: f(0)
[1.2.2.2.1] doe niets
[1.2.2.3] teken rechter boom: f(0)
[1.2.2.3.1] doe niets
[1.2.3] teken rechter boom: f(1)
[1.2.3.1] teken stam
[1.2.3.2] teken linker boom: f(0)
[1.2.3.2.1] doe niets
[1.2.3.3] teken rechter boom: f(0)
[1.2.3.3.1] doe niets
[1.3] teken rechter boom: f(2)
[1.3.1] teken stam
[1.3.2] teken linker boom: f(1)
[1.3.2.1] teken stam
[1.3.2.2] teken linker boom: f(0)
[1.3.2.2.1] doe niets
[1.3.2.3] teken rechter boom: f(0)
[1.3.2.3.1] doe niets
[1.3.3] teken rechter boom: f(1)
[1.3.3.1] teken stam
[1.3.3.2] teken linker boom: f(0)
```

```
[1.3.3.2.1] doe niets
[1.3.3.3] teken rechter boom: f(0)
[1.3.3.3.1] doe niets
```

Merk op dat een *recursieve definitie* vaak erg eenvoudig is, maar de *uitvoering* van een recursieve functie veel werk met zich mee brengt: er zijn veel stappen en je moet een goede administratie bijhouden om steeds te weten met welk subprobleem je bezig bent. Typisch werk voor een computer dus.

4.8 Inductie

Laten we nog eens kijken naar de rij met oplossingen voor de torens van Hanoi. Met de recursieve formule $a_n = 2a_{n-1} + 1$ konden we voor elke n uitrekenen in hoeveel stappen de puzzel was op te lossen. Maar als we willen weten hoeveel stappen er nodig zijn om de puzzel met 38 schijven op te lossen, moeten we eerst uitrekenen hoeveel stappen er nodig zijn voor de puzzel met 37 schijven. En dat kunnen we alleen door eerst uit te rekenen hoeveel stappen er zijn voor de puzzel met 36 stappen. Dat is natuurlijk een heel gedoe. Het zou veel fijner zijn als we een directe methode hadden om a_{38} uit te rekenen. Met een beetje nadenken lukt dat gelukkig!

Als we nog eens naar de rij van oplossingen 1,3,7,15,31,63,127,... kijken valt je misschien op dat de rij lijkt op de rij 2,4,8,16,32,64,128,... en dat is de rij van de machten van twee. De eerste rij is steeds eentje minder. Is dit toeval of klopt het altijd? Stel even dat $a_{37} = 2^{37} - 1$, dan is

$$a_{38} = 2 \cdot a_{37} + 1 = 2 \cdot \left(2^{37} - 1\right) + 1 = 2^{38} - 2 + 1 = 2^{38} - 1.$$

Dus als het voor het 37ste element klopt, dan ook voor het 38ste. Nu is 37 natuurlijk helemaal niet bijzonder. Dus op dezelfde manier zien we dat als het voor n klopt dan ook voor $n + 1$. Verder wisten we al dat $a_1 = 2^1 - 1$. Dus het klopt voor 2, dus het klopt voor 3, dus het klopt voor 4, dus ..., dus het klopt voor 37, dus het klopt voor 38, dus .... We zien dus dat voor alle getallen n geldt: $a_n = 2^n - 1$. Dit noemen we een *directe formule* voor a_n omdat er rechts van het gelijkteken geen a's meer voorkomen.

De bewijsmethode die we net hebben gebruikt heet *inductie*. Inductie lijkt heel veel op de recursieve methode die we gebruiken om functies te definiëren.

Definitie 4.25 Inductie kunnen we gebruiken als we voor alle natuurlijke getallen (de getallen 0, 1, 2, 3, 4, 5, 6, 7, ...) een bepaalde uitspraak $P(n)$ willen bewijzen. Een *bewijs met inductie* gaat op de volgende manier:

1. Bewijs eerst $P(0)$. (De *basisstap*.)
2. Bewijs dan: Als $P(n)$, dan geldt ook $P(n + 1)$. (De *inductiestap*.)

Dat is genoeg. De aanname $P(n)$ waaruit we $P(n + 1)$ willen bewijzen, noemen we ook wel de *inductie hypothese* (IH).

Als we nu willen laten zien dat bijvoorbeeld $P(37)$ geldt, dan laten we eerst zien dat $P(0)$ waar is, en dus (met 2.) ook $P(1)$ en dus (weer met 2.) ook $P(2)$, dus ook $P(3)$,..., dus ook $P(36)$, dus ook $P(37)$!

In het vorige voorbeeld was $P(n)$ de uitspraak $a_n = 2^n - 1$ voor $n \geq 1$.

Opmerking 4.26 Een terugkerende vraag is of je nu met $P(0)$ of $P(1)$ moet beginnen. Het antwoord daarop is: dat ligt aan de situatie. Wil je voor alle natuurlijke getallen iets bewijzen, dan moet je met $P(0)$ beginnen. Wil je echter iets bewijzen voor alle natuurlijke getallen boven de 7, dan is 8 het kleinste getal uit die serie en begin je dus met $P(8)$. Soms is het zelfs nodig om een paar basisstappen apart te controleren omdat de regelmaat die in de inductiestap nodig is, nog niet meteen aanwezig is, maar bijvoorbeeld pas vanaf $P(5)$.

Opmerking 4.27 Inductie kun je zien als het omgekeerde van recursie. Met inductie ligt de nadruk op steeds moeilijkere gevallen, bij recursie is dat omgekeerd.

Voorbeeld 4.28 Hoeveel is $1 + 2 + 3 + \cdots + 99$? Dat kun je natuurlijk op papier doen. Je kunt je rekenmachine gebruiken. In beide gevallen ben je lang bezig. Tenzij je het slim doet. Je kunt bijvoorbeeld definiëren $s(n) := 1 + 2 + \cdots + n$ voor $n \geq 1$. Dan kun je een recursief programmaatje schrijven dat $s(99)$ voor je uitrekent, want $s(n + 1) = s(n) + n + 1$.

Als je wat waarden uitrekent krijg je het vermoeden dat $s(n) = (n^2+n)/2$. Maar klopt dit nu ook? We gaan het bewijzen met inductie. Neem voor $P(n)$ de uitspraak $s(n) = (n^2+n)/2$.

Basisstap Geldt $P(1)$? Ja, want $s(1) = 1 = (1^2 + 1)/2$.

Inductiestap Stel nu dat $P(n)$ oftewel dat $s(n) = (n^2 + n)/2$. Geldt dan ook $P(n + 1)$?

Laten we eens kijken wat we allemaal weten over $s(n + 1)$:

$$\begin{aligned} s(n+1) &= s(n) + n + 1 \text{ (definitie } s(n+1)) \\ &= \frac{n^2+n}{2} + n + 1 \text{ (IH)} \\ &= \frac{n^2+n+2n+2}{2} \text{ (rekenregels)} \\ &= \frac{(n^2+2n+1)+(n+1)}{2} \text{ (rekenregels)} \\ &= \frac{(n+1)^2+(n+1)}{2} \text{ (rekenregels)} \end{aligned}$$

Dus $P(n + 1)$ geldt inderdaad.

Dus geldt voor alle getallen $n \geq 1$ dat $s(n) = (n^2 + n)/2$.

De wiskundige Gauss kon dit probleem overigens ook eenvoudig oplossen zonder gebruik te maken van inductie. Zie hiervoor opgave 4.U.

Voorbeeld 4.29 Op hoeveel manieren kunnen we de cijfers 1,2,3,4,5,6,7,8,9 rangschikken? Het is lastig alle mogelijkheden op te schrijven. Maar gelukkig hoeft dat niet. Laat a_n het aantal rangschikkingen zijn van een verzameling van n elementen. We willen dus a_9 weten. Gelukkig weten we dat $a_1 = 1$. Verder is $a_{n+1} = (n + 1) \cdot a_n$. Want we kiezen eerst één van de $(n + 1)$ elementen en dat element zetten we vooraan in de rij. Daarna hebben we nog n elementen en die kunnen we op a_n manieren rangschikken.

Hoe rekenen we a_9 nu uit? De definitie die we boven hebben gegeven is precies de definitie van de faculteitfunctie, die wordt genoteerd met $n!$. Dus $a_9 = 9!$ en algemeen $a_n = n!$ Deze functie zit standaard op de meeste rekenmachines.

Voorbeeld 4.30 Wat is de uitkomst van de som

$$1 + a + a^2 + a^3 + \cdots + a^n$$

voor willekeurige a en voor $n \in \mathbb{N}$? Laten we eens wat gevallen uitrekenen:

	$n = 0$	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 10$	$n = 42$
$a = 0$	1	1	1	1	1	1	1
$a = 1$	1	2	3	4	5	11	43
$a = 3$	1	4	13	40	121	88573	164128483697268538813
$a = -1$	1	0	1	0	1	1	1
$a = -2$	1	-1	3	-5	11	683	2932031007403

Een oplettende student herinnert zich van de middelbare school vast de formule voor de som van een meetkundige rij en komt tot de conclusie:

$$1 + a + a^2 + a^3 + \cdots + a^n = \begin{cases} n+1 & \text{als } a = 1 \\ \frac{a^{n+1}-1}{a-1} & \text{als } a \neq 1 \end{cases}$$

We gaan het geval dat $a \neq 1$ bewijzen met inductie. Daartoe maken we eerst expliciet wat ons predikaat P betekent:

$$P(n) := \quad 1 + a + a^2 + a^3 + \cdots + a^n = \frac{a^{n+1} - 1}{a - 1}$$

De inductie hypothese is dan natuurlijk: $P(n)$ is waar.

Basisstap Omdat we het voor alle $n \in \mathbb{N}$ moeten bewijzen, is $P(0)$ de basisstap. Vullen we $n = 0$ in dan moeten we dus bewijzen dat

$$1 = \frac{a^1 - 1}{a - 1}$$

Maar omdat $a^1 - 1$ natuurlijk gelijk is aan $a - 1$ en $a - 1 \neq 0$, is dit waar en is de basisstap bewezen.

Inductiestap Onder aanname van de inductie hypothese, dus dat $P(n)$ waar is, moeten we nu bewijzen dat ook $P(n + 1)$ waar is, dus dat

$$1 + a + a^2 + a^3 + \cdots + a^n + a^{n+1} = \frac{a^{n+2} - 1}{a - 1}$$

geldt. We proberen de linkerkant van deze vergelijking te herschrijven zodat we er een deel van $P(n)$ in herkennen, passen dan de IH toe en proberen vervolgens het resultaat

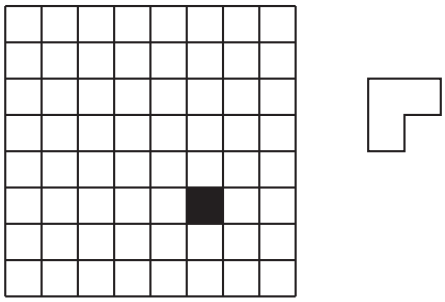
te herschrijven naar de rechterkant van deze vergelijking.

$$\begin{aligned}
&1 + a + a^2 + a^3 + \cdots + a^n + a^{n+1} \\
&\quad = \quad (1 + a + a^2 + a^3 + \cdots + a^n) + a^{n+1} \text{ (maak linkerkant } P(n) \text{ zichtbaar)} \\
&\quad = \quad \frac{a^{n+1} - 1}{a - 1} + a^{n+1} \text{ (IH)} \\
&\quad = \quad \frac{a^{n+1} - 1}{a - 1} + a^{n+1} \cdot \frac{a - 1}{a - 1} \text{ (maak noemers gelijknamig)} \\
&\quad = \quad \frac{a^{n+1} - 1}{a - 1} + \frac{a^{n+2} - a^{n+1}}{a - 1} \text{ (vermenigvuldiging van breuken)} \\
&\quad = \quad \frac{a^{n+1} - 1 + a^{n+2} - a^{n+1}}{a - 1} \text{ (optellen van breuken)} \\
&\quad = \quad \frac{a^{n+2} - 1}{a - 1} \text{ (vereenvoudigen)}
\end{aligned}$$

In het bijzonder zijn we precies uitgekomen op de rechterkant van $P(n + 1)$.

Dus hebben we nu voor alle $n \in \mathbb{N}$ en alle $a \neq 1$ bewezen dat $P(n)$ waar is.

Voorbeeld 4.31 Tot nu toe hebben we vooral gekeken naar stellingen waarbij de bewijzen gebaseerd zijn op de kennis van en vaardigheid met rekenregels. Dit voorbeeld laat zien dat het soms ook om andere dingen gaat. Bekijk het volgende 8×8 'schaakbord'. Eén van de vakjes is verwijderd. Kan de rest van het bord betegeld worden met 21 tegels van de vorm zoals rechts aangegeven?



Na een beetje puzzelen kom je er waarschijnlijk achter dat dit in dit geval inderdaad kan. Echter, was het toeval omdat de verwijderde tegel hier toevallig op een gunstige plek lag? Nee, het blijkt in zijn algemeenheid te kunnen en dat gaan we met inductie bewijzen.

Hiertoe definiëren we eerst weer het predikaat $P(n)$.

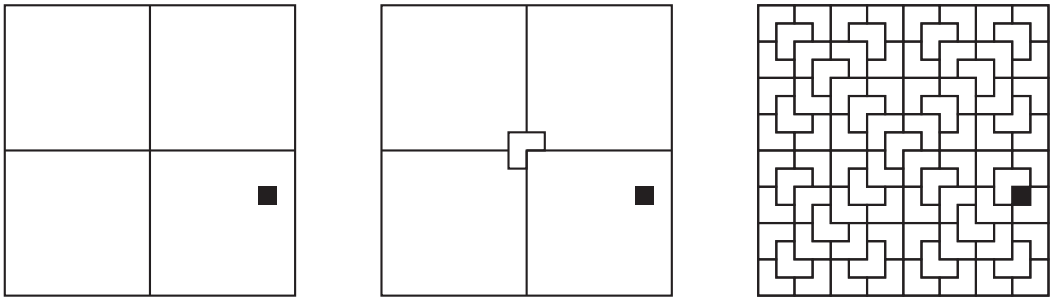
$P(n) :=$ Een $2^n \times 2^n$ 'schaakbord' met $n \in \mathbb{N}$, $n \geq 1$ waarvan één vakje is verwijderd kan altijd precies worden betegeld.

Het bewijs gaat met inductie naar n.

Basisstap De basis is hier $P(1)$, want in de stelling staat expliciet aangegeven dat het om $n \geq 1$ gaat. De vraag is dus of we een 2×2 'schaakbord' kunnen betegelen als er één vakje is verwijderd. Ongeacht welk vakje verwijderd is, door draaien kunnen we er altijd voor zorgen dat dat vakje rechtsonder zit. En voor het al dan niet kunnen betegelen maakt het draaien van het hele bord natuurlijk niet uit.

Het plaatje bewijst dat $P(1)$ dus waar is.

Inductiestap We mogen nu aannemen dat $P(n)$ waar is en moeten daarmee bewijzen dat ook $P(n+1)$ waar is. Neem dus een 'schaakbord' van $2^{n+1} \times 2^{n+1}$ en verwijder één vakje. Merk op dat we zo'n bord kunnen verdelen in vier kleinere borden van elk $2^n \times 2^n$. Na draaiing mogen we aannemen dat het verwijderde vakje in het vierkant rechtsonder zit. We krijgen dan de situatie uit het linker plaatje.



Vervolgens plaatsen we nu één tegel op een strategische plek en krijgen de situatie uit het middelste plaatje. Merk op dat deze tegel precies één vakje bedekt in de drie vierkanten waar het op ligt. Nu kunnen we vier keer de inductie hypothese toepassen. Immers, uit elk van de vier vierkanten van $2^n \times 2^n$ is precies één vakje verwijderd en met $P(n)$ weten we dan dat elk van die vierkanten te betegelen valt met deze tegels. Maar via de constructie is dan ook meteen aangetoond dat het grote $2^{n+1} \times 2^{n+1}$ vierkant minus het ene verwijderde vakje te betegelen is. En dus geldt $P(n+1)$ inderdaad.

In dit geval geeft dit bewijs ook meteen een constructie om de betegeling te maken. In het rechter plaatje is de betegeling afgemaakt.

Opgave 4.Q De uitvinder van het schaakbord mocht van de koning van Perzië een beloning uitzoeken. Hij mocht vragen wat hij maar wilde. Hij koos voor de volgende beloning. Hij wilde op het eerste vakje van het schaakbord 1 rijstkorrel hebben, op de tweede 2, op de derde 4, etc. Dus op ieder vakje twee keer zoveel rijstkorrels. De koning vond hem erg bescheiden. Laten we eens kijken hoeveel hij vroeg: hij wilde dus

$$1 + 2 + 2^2 + 2^3 + 2^4 + \cdots + 2^{63}$$

korrels hebben. Kun je een eenvoudige formule vinden voor de uitkomst van deze som? [Hint: tel eerst eens wat termen uit het begin van de rij $1, 2, 2^2, 2^3, 2^4, \ldots$ op en bekijk het verband, dus bekijk 1 en $1+2$ en $1+2+2^2$ enz. Herken je iets? Wat is je vermoeden van de uitkomst? Kun je dit met inductie bewijzen?]

Opgave 4.R Gegeven is een rij a_n gedefinieerd door

$$\begin{cases} a_0 &= 0 \\ a_{n+1} &= a_n + 2n + 1 \quad \text{voor } n \geq 0 \end{cases}$$

Bewijs dat voor elke $n \in \mathbb{N}$ geldt: $a_n = n^2$.

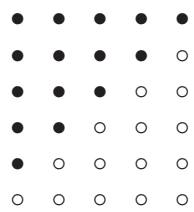
Opgave 4.S In opgave 4.C heb je bewezen dat voor elke $n \geq 1$ geldt dat de graaf K_n precies $\frac{1}{2}n(n-1)$ lijnen heeft. Bewijs dat nu nog eens, maar dan met inductie naar n.

Opgave 4.T Bewijs met inductie dat voor alle $n \in \mathbb{N}$, $2^n \geq n$.

Opgave 4.U We hebben al een mooie formule voor de som van de rij

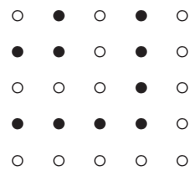
$$1 + 2 + 3 + 4 + 5 + \cdots + n = \frac{(n+1)n}{2}.$$

Wat is het verband met het volgende plaatje?



Opgave 4.V Hoeveel rijtjes kunnen we maken van lengte n met daarin alleen de getallen 1 tot en met 5? Dit kunnen we ook weer recursief oplossen. Noem het aantal rijtjes van lengte n a_n. Een rijtje van lengte 1 kunnen we op vijf manieren maken, dus $a_1 = 5$. Een rijtje van lengte $n+1$ kunnen we maken door eerst een rijtje van lengte n te maken en daar een element achter te zetten. Dus $a_{n+1} := 5 \cdot a_n$. Herinner je nu even de definitie van machtsverheffen. Bewijs met inductie dat voor alle n, $a_n = 5^n$.

Opgave 4.W Bewijs met inductie dat $1+3+5+7+\cdots+(2n-1) = n^2$. Wat is het verband met het volgende plaatje?



4.9 Driehoek van Pascal

Voordat we echt naar de driehoek van Pascal gaan kijken, herhalen we eerst wat combinatoriek aan de hand van een voorbeeld.

Voorbeeld 4.32 We gaan kijken hoe de Nederlandse Lotto[8] werkt. Bij dat kansspel is er een 'vaas' met 45 verschillend genummerde ballen. Een notaris trekt hier zonder teruglegging zes ballen uit. De zes getallen die op deze ballen staan worden vergeleken met de zes getallen die elke speler zelf heeft gekozen. Hoe meer getrokken getallen overeenkomen met de zelf gekozen getallen, hoe hoger de uitkering. Hoeveel verschillende uitkomsten van zo'n Lottotrekking zijn er nu eigenlijk? Wel, voor de eerste bal zijn er natuurlijk 45 mogelijkheden. Voor de tweede nog 44, voor de derde nog 43, enzovoorts. Dus voor de zes ballen zijn er

$$45 \cdot 44 \cdot 43 \cdot 42 \cdot 41 \cdot 40 = \frac{45 \cdot 44 \cdot 43 \cdot 42 \cdot 41 \cdot 40 \cdot 39 \cdot 38 \cdot \cdots \cdot 1}{39 \cdot 38 \cdot \cdots \cdot 1} = \frac{45!}{39!} = \frac{45!}{(45-6)!}$$

[8] We laten de Jackpot en Superzaterdag hier buiten beschouwing.

mogelijkheden. Maar omdat de onderlinge volgorde van trekken niet van belang is voor de totale uitkomst, hebben we hier een hele hoop dezelfde uitkomsten dubbel geteld en wel precies 6! keer. Om dat te corrigeren moeten we dus nog door 6! delen en uiteindelijk zijn er dus

$$\frac{45!}{6!\,(45-6)!} = 8145060$$

echt verschillende uitkomsten bij een Lottotrekking.

Definitie 4.33 Zij n en k natuurlijke getallen met $k \leq n$. Dan definiëren we de *binomiaalcoëfficiënt* $\binom{n}{k}$ als het aantal manieren om k objecten uit een verzameling van n elementen te pakken. Dit getal wordt ook wel het *combinatiegetal* genoemd.

We bekijken nu de roosterpunten (n, k) waarvoor $k \leq n$. Deze verzameling van roosterpunten tekenen we zó, dat het punt $(0, 0)$ de top wordt:

$$\begin{array}{ccccccccccccccccccccc}
 & & & & & & & & (0,0) & & & & & & & & & & \\
 & & & & & & & (1,0) & & (1,1) & & & & & & & & & \\
 & & & & & & (2,0) & & (2,1) & & (2,2) & & & & & & & & \\
 & & & & & (3,0) & & (3,1) & & (3,2) & & (3,3) & & & & & & & \\
 & & & & (4,0) & & (4,1) & & (4,2) & & (4,3) & & (4,4) & & & & & & \\
 & & & (5,0) & & (5,1) & & (5,2) & & (5,3) & & (5,4) & & (5,5) & & & & & \\
 & & (6,0) & & (6,1) & & (6,2) & & (6,3) & & (6,4) & & (6,5) & & (6,6) & & & & \\
\ldots & & \ldots & & \ldots & & \ldots & & \ldots & & \ldots & & \ldots & & \ldots & & \ldots & &
\end{array}$$

We gaan nu op verschillende manieren ieder van deze roosterpunten voorzien van een getal.

Definitie 4.34 Zet aan de rand eentjes en vul de rest in door 'schuin optellen'. Anders gezegd: voorzie de roosterpunten $(n, 0)$ en (n, n) van het getal 1 en schrijf bij elk ander roosterpunt (n, k) de som van de getallen, die je bij de roosterpunten $(n-1, k)$ en $(n-1, k-1)$ moet schrijven. De *Eerste Driehoek van Pascal* (1e $\triangle$vP) is dus het volgende schema van getallen[9]

$$\begin{array}{ccccccccccccccccccccc}
 & & & & & & & & 1 & & & & & & & & & & \\
 & & & & & & & 1 & & 1 & & & & & & & & & \\
 & & & & & & 1 & & 2 & & 1 & & & & & & & & \\
 & & & & & 1 & & 3 & & 3 & & 1 & & & & & & & \\
 & & & & 1 & & 4 & & 6 & & 4 & & 1 & & & & & & \\
 & & & 1 & & 5 & & 10 & & 10 & & 5 & & 1 & & & & & \\
 & & 1 & & 6 & & 15 & & 20 & & 15 & & 6 & & 1 & & & & \\
 & 1 & & 7 & & 21 & & 35 & & 35 & & 21 & & 7 & & 1 & & &
\end{array}$$

Definitie 4.35 Zet op plaats (n, k) het getal $\binom{n}{k}$. Dit levert de zogenaamde *Tweede Driehoek van Pascal* (2e $\triangle$vP) op.

Opgave 4.X Bewijs dat er in de 2e $\triangle$vP aan de rand allemaal eentjes staan.

We beweren dat als $k < n$, dan $\binom{n+1}{k+1} = \binom{n}{k+1} + \binom{n}{k}$. Kiezen we namelijk een deelverzameling A van k elementen uit $\{1, 2, 3, 4, \ldots, n+1\}$ dan zijn er twee mogelijkheden: ofwel $n+1 \in A$, ofwel $n+1 \notin A$. In het eerste geval is $A \setminus \{n+1\}$ dus een deelverzameling van

[9] Het gaat natuurlijk naar onderen oneindig ver door; we hebben alleen maar een klein beginstuk opgeschreven.

k elementen uit $\{1, 2, 3, 4, \ldots, n\}$, in het tweede geval is A een deelverzameling van $k+1$ elementen uit $\{1, 2, 3, 4, \ldots, n\}$. Dus ook in de 2e $\triangle$vP geldt het principe van 'schuin optellen': ieder inwendig getal is de som van de schuin erboven staande getallen. Gevolg: de 2e $\triangle$vP is precies gelijk aan de 1e $\triangle$vP.

Definitie 4.36 Zet op plaats (n, k) het aantal wegen via roosterpunten van $(0, 0)$ naar (n, k), waarbij we iedere keer een stap naar linksonder of rechtsonder zetten. Dit levert de zogenaamde *Derde Driehoek van Pascal* (3e $\triangle$vP) op.

Je kunt nu inzien:

- In de 3e $\triangle$vP staan aan de rand allemaal eentjes.
- In de 3e $\triangle$vP geldt ook het principe van 'schuin optellen'.

Gevolg: de 3e $\triangle$vP is precies gelijk aan de 1e $\triangle$vP (en dus ook aan de 2e $\triangle$vP).

Definitie 4.37 Zet op plaats (n, k) de coëfficiënt van x^k in de veelterm $(1 + x)^n$. Dus op plaats $(6, 2)$ staat 15 want $(1+x)^6 = 1+6x+15x^2+20x^3+15x^4+6x^5+x^6$ en de coëfficiënt van x^2 is dus 15. Dit levert de zogenaamde *Vierde Driehoek van Pascal* (4e $\triangle$vP) op.

Opgave 4.Y Laat zien dat ook de 4e $\triangle$vP gelijk is aan de 1e $\triangle$vP.

De vier behandelde Driehoeken van Pascal leiden dus allemaal tot hetzelfde schema van getallen. Ondanks dat we vier varianten hebben geïntroduceerd, mogen we dus best van *de* driehoek van Pascal spreken.

Opgave 4.Z Laat zien hoe je in de driehoek van Pascal kunt vinden hoeveel manieren er zijn om vier objecten uit een collectie van zes objecten te kiezen. Wat is de notatie voor de bijbehorende binomiaalcoëfficiënt?

Definitie 4.38 De volgende gelijkheid staat bekend als het *Binomium van Newton*.

$$(1+x)^n = \binom{n}{0} + \binom{n}{1}x + \binom{n}{2}x^2 + \cdots + \binom{n}{n}x^n$$

Dit Binomium van Newton is eigenlijk niets anders dan de stelling: 2e $\triangle$vP is gelijk aan 4e $\triangle$vP. Het meer algemene geval

$$(y+z)^n = \binom{n}{0}y^n + \binom{n}{1}y^{n-1}z + \binom{n}{2}y^{n-2}z^2 + \cdots + \binom{n}{n}z^n$$

volgt hier uit: $(y+z)^n = y^n(1+(\frac{z}{y})^n)$ als $y \neq 0$. Het geval $y = 0$ is eenvoudig.

4.10 Belangrijke begrippen

5 Modale logica

Propositielogica en predikaatlogica zijn de bekendste, maar er zijn er veel meer 'logica's'. Verschillende logica's verschillen in de mate waarin ze nuances van uitspraken kunnen representeren. Ze verschillen in de operatoren waarmee je de formules kunt opschrijven en ze verschillen soms ook in de interpretatie die aan die operatoren wordt gegeven.

Een paar voorbeelden van logica's: Floyd-Hoare logica (waarin je het gedrag van programma's kunt opschrijven), intuïtionistische logica (die gaat over wat je met een computer kunt berekenen), en paraconsistente logica (waarin je een uitspraak kunt terugnemen wanneer er meer informatie beschikbaar komt).

Een belangrijke groep logica's zijn de *modale logica's*, die gaan over de *mate* waarin een uitspraak waar is. Modale logica is het onderwerp van dit hoofdstuk.

Binnen de modale logica's is er de belangrijke subgroep van de *temporele logica's*. Dit zijn logica's waarin je kunt opschrijven *wanneer* een uitspraak waar is. We zullen aan het eind van dit hoofdstuk een temporele logica in detail bekijken. Zie verder de boeken [2] en [3] voor meer informatie over modale logica's.

5.1 Noodzaak en mogelijkheid

Traditioneel gaat modale logica over *noodzaak*. Neem de volgende zin:

Het regent.

Deze zin kan waar zijn of niet waar zijn, maar hij is niet *noodzakelijk* waar. Het regent zeker niet altijd en zelfs als het regent is dat niet zo omdat het zo *moet* zijn: als het weer anders was geweest had het niet geregend. Anders ligt het bij de zin:

Als het regent dan valt er water uit de lucht.

Deze zin is duidelijk wel noodzakelijk waar: het hoort bij de betekenis van het woord 'regenen' dat er water uit de lucht valt.

Dit verschil in noodzaak komt niet in de propositielogica tot uitdrukking. Als we het volgende woordenboek gebruiken:

R	het regent
W	er valt water uit de lucht

dan worden de vertalingen in propositielogica van deze twee zinnen:

$$R$$

en

$$R \to W$$

en daarin komt het verschil in noodzakelijkheid niet tot uitdrukking. In de modale logica is er daarom een operator $\Box$ ingevoerd die 'noodzakelijk' betekent. De zin:

Het is noodzakelijk dat als het regent dat er dan water uit de lucht valt.

wordt geschreven als:

$$\Box(R \to W)$$

Misschien denk je dat het omgekeerde ook geldt, dat als er water uit de lucht valt dat het dan regent, maar dat is niet zo. Als je een tuinsproeier aanzet, of onder een waterval gaat staan, dan valt er wel water uit de lucht, maar regent het niet. Daarom hebben we:

Het is niet noodzakelijk dat als er water uit de lucht valt dat het dan regent.

wat als formule van de modale logica geschreven wordt als:

$$\neg\Box(W \to R)$$

Evenwel kan de formule

$$W \to R$$

in specifieke situaties best waar zijn. Als het regent is de rechterkant van de implicatie waar, en de waarheidstabel van de implicatie leert ons dat in dat geval de hele formule waar is.

Behalve voor noodzakelijkheid heeft de modale logica ook een notatie voor *mogelijkheid*. Dit wordt geschreven als $\Diamond$. De ware zin:

Het is mogelijk dat het regent.

wordt in de modale logica weergegeven door de formule:

$$\Diamond R$$

Als je er diep over nadenkt zul je inzien dat deze zin hetzelfde betekent als:

Het is niet noodzakelijk dat het niet regent.

Deze zin wordt weergegeven door de formule:

$$\neg\Box\neg R$$

Opgave 5.A De formule $\Diamond R$ betekent dus hetzelfde als $\neg\Box\neg R$. Vind een formule zonder het symbool '$\Box$' die hetzelfde betekent als $\Box R$. Vertaal zowel $\Box R$ als de formule die je hebt gevonden in een Nederlandse zin.

Definitie 5.1 Er zijn voor een uitspraak U drie mogelijkheden:

- U is noodzakelijk
- U is onmogelijk
- U is contingent

Een uitspraak heet *contingent* als hij niet noodzakelijk waar, maar ook niet onmogelijk is. Contingentie zit dus tussen noodzakelijk waar en noodzakelijk onwaar in.

Opgave 5.B Nu wordt 'U is noodzakelijk' symbolisch opgeschreven als $\Box U$, en 'U is onmogelijk' als $\Box\neg U$ of $\neg\Diamond U$. Geef twee verschillende formules van de modale logica die 'U is contingent' uitdrukken.

Opgave 5.C Gebruik het woordenboek:

G	ik heb geld
K	ik koop iets

en vertaal de volgende Nederlandse zinnen naar formules van de modale logica:

(i) *Het is mogelijk dat ik iets koop zonder dat ik geld heb.*
(ii) *Het is noodzakelijk dat als ik iets koop dat ik dan ook geld heb.*
(iii) *Het is mogelijk dat als ik iets koop dat ik dan geen geld heb.*

Welke van *deze* zinnen lijken je waar? Leg uit waarom.

5.2 Syntax

We geven nu de contextvrije grammatica van de formules van de modale propositielogica. (Er bestaat ook modale predikaatlogica, maar daarover zullen we het hier niet hebben.) Deze grammatica is precies hetzelfde als die van de propositielogica (zie opmerking 1.4), behalve dat de twee modale operatoren zijn toegevoegd:

$$S \;\to\; a \mid \neg S \mid (S \wedge S) \mid (S \vee S) \mid (S \to S) \mid (S \leftrightarrow S) \mid \Box S \mid \Diamond S$$

Conventie 5.2 Net als bij de propositielogica mogen we in modale formules haakjes weglaten. Hierbij binden de modale operatoren net zo sterk als de negatie '$\neg$', dus sterker dan de binaire voegtekens.

Dat wil zeggen dat de formule

$$\Diamond a \wedge b \to \Diamond(\Box a \vee \neg c)$$

moet worden gelezen als

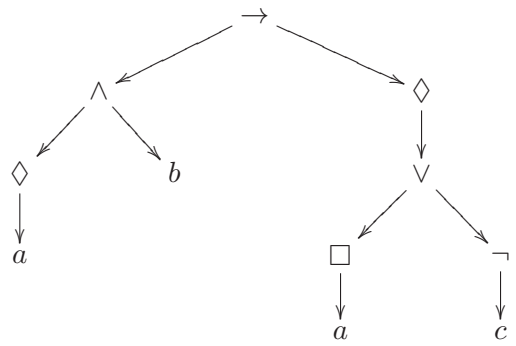
$$(((\Diamond a) \wedge b) \to (\Diamond((\Box a) \vee (\neg c))))$$

Omdat in de grammatica de unaire operatoren geen haakjes hebben, is de formule die voldoet aan de grammatica:

$$((\Diamond a \wedge b) \to \Diamond(\Box a \vee \neg c))$$

Dit is de 'officiële' vorm van de formule. De twee andere formules moeten worden gezien als een (mogelijk duidelijkere) manier om deze string van zeventien symbolen op te schrijven.

We kunnen de structuur van deze formule tekenen als een boom:



In deze boom staan de atomaire proposities aan de bladeren en de logische operatoren in de knopen.

Opgave 5.D Geef bij de volgende formules van de modale logica de vorm die aan de officiële grammatica voldoet, en teken de bijbehorende bomen.

(i) $\Box(\Box a)$
(ii) $\Box a \to a$
(iii) $(\Box a) \to a$
(iv) $\Box(a \to a)$
(v) $\neg a \to \neg\Box a$
(vi) $\Diamond a \to \Box\Diamond a$
(vii) $\Box a \to \Diamond a$
(viii) $\Diamond a \wedge b \to \Diamond\Box a \vee \neg c$

5.3 Modaliteiten

Tot nog toe had de operator $\Box$ de betekenis 'noodzakelijk' en de operator $\Diamond$ de betekenis 'mogelijk'. Dit is de traditionele lezing, maar is slechts één van de interpretaties voor deze symbolen. Er zijn vele andere interpretaties mogelijk. Afhankelijk van hoe we deze symbolen lezen krijgen we dan verschillende modale logica's:

logica	modaliteit	$\Box f$	$\Diamond f$
modale logica	noodzaak	f is noodzakelijk	f is mogelijk
epistemische logica	kennis	ik weet dat f	f is niet strijdig met mijn kennis
doxastische logica	geloof	ik geloof dat f	f is niet strijdig met wat ik geloof
temporele logica	tijd	altijd f	soms f
deontische logica	plicht	je moet f	je mag f
programma-logica		na executie moet f gelden	na executie kan f gelden

Als je er verder niets bij zegt wordt met 'modale logica' meestal de logica van de noodzaak bedoeld. De term wordt evenwel ook gebruikt om de hele familie van modale logica's aan te duiden.

Opgave 5.E Gebruik het volgende woordenboek:

K	ik ben klaar
H	ik ga naar huis

Vertaal voor ieder van de logica's uit de tabel (behalve voor de programma-logica) de formule

$$\neg K \to \neg\Diamond H$$

naar het Nederlands.

De logica's in de tabel hebben ieder zelf ook weer varianten. Zo zijn er bijvoorbeeld, net zoals er verschillende modale logica's zijn, ook verschillende temporele logica's. Met 'altijd' kan dan bijvoorbeeld 'altijd vanaf nu, inclusief dit moment' of 'altijd vanaf nu, maar op dit moment hoeft nog niet' bedoeld worden. Of er kan rekening worden gehouden met het feit dat je nog niet weet wat er gaat gebeuren. (Dit heet in het Engels: 'branching time'.)

5.4 Axioma's

Afhankelijk van hoe je de modale operatoren leest gelden er verschillende eigenschappen. Zo kun je je afvragen of alle formules van de vorm

$$\Box f \to f$$

altijd waar zijn. (Pas op dat je dit moet lezen als '$(\Box f) \to f$' en niet als '$\Box(f \to f)$', want de $\Box$ bindt sterker dan de $\to$.) In de standaardinterpretatie van de modale logica geldt dit altijd. Dan staat hier:

Als f noodzakelijk is, dan f.

en als iets noodzakelijk waar is, is het natuurlijk zeker het geval. Maar het geldt niet in de logica van het geloof, de doxastische logica. Want dan staat er:

Als ik geloof dat f, dan f.

en als ik iets geloof dan hoeft het helemaal niet zo te zijn, want ik kan me best vergissen. In de logica van de kennis, de epistemische logica, geldt het dan weer wel, want dan staat er:

Als ik weet dat f, dan f.

En als ik *weet* dat iets zo is, dan is het ook zo, want anders zou het geen weten zijn. Het hangt dus van de interpretatie van de $\Box$ af of bovenstaand principe in zijn algemeenheid geldig is of niet.

In de modale logica zijn er vele van dergelijke principes, die *axiomaschema's* van de modale logica worden genoemd. De belangrijkste hiervan zijn:

naam	axiomaschema	eigenschap
K	$\Box(f \to g) \to (\Box f \to \Box g)$	distributief
T	$\Box f \to f$	reflexief
B	$f \to \Box\Diamond f$	symmetrisch
4	$\Box f \to \Box\Box f$	transitief
5	$\Diamond f \to \Box\Diamond f$	Euclidisch
D	$\Box f \to \Diamond f$	serieel

In de eerste kolom staat de letter of cijfer die de naam van het axiomaschema is. In de laatste kolom staat de eigenschap waarmee dat schema ook wel wordt aangeduid.

Opgave 5.F Maak een tabel van modale logica's en axiomaschema's, waarin je aangeeft in welke logica's welke schema's gelden. Zet de logica's uit de lijst op pagina 64 langs de linkerkant van de tabel en de axiomaschema's aan de bovenkant. Zet nu een '+' in de tabel wanneer je denkt dat het schema in de logica geldt, en een '−' wanneer je denkt dat dat niet altijd zo hoeft te zijn. Zo maak je dus een tabel met 36 plussen en minnen.

Afhankelijk van welke axiomaschema's je mag gebruiken heb je een ander soort modale logica. Hierbij gaat het dus niet om de *interpretatie* van de $\Box$ en de $\Diamond$, maar op welke manieren je met deze symbolen mag redeneren. De belangrijkste modale logica's in deze technische zin zijn:

systeem	axioma's
K	K
D	K + D
T	K + T
S4	K + T + 4
S5	K + T + B + 4 + 5

Zoals je ziet geldt het K-axioma in ieder van deze logica's. Dit schema geldt als de basis van de modale logica en de modale logica **K** is dan ook de *zwakste* van alle modale logica's.

5.5 Mogelijke werelden en Kripke modellen

In sectie 5.1 hebben we intuïtief uitspraken gedaan over de vraag of beweringen waar zijn of niet. Hier gaan we dat netjes formaliseren. We zeggen dat 'het regent' niet noodzakelijk waar is, omdat we ons ook een wereld kunnen voorstellen waarin het niet regent. Zo'n wereld heet een *mogelijke wereld*. Als het in alle mogelijke werelden regent is het duidelijk wél noodzakelijk dat het regent. De manier om via mogelijke werelden aan formules uit de modale logica een betekenis te geven heet mogelijke werelden-semantiek. (In de logica is *syntax* de verzameling regels die vertelt welke formules welgevormd zijn en is *semantiek* de verzameling regels die vertelt wat de formules betekenen. Syntax gaat dus over de vorm en semantiek over de inhoud.)

Je zou kunnen denken dat in de mogelijke werelden-semantiek de formule $\Box f$ betekent dat de formule f in alle werelden waar is en dat de formule $\Diamond f$ betekent dat de formule f in minstens één wereld waar is, maar dat is niet algemeen genoeg. Met deze semantiek zou immers de formule

$$\Box f \to f$$

altijd waar zijn (als het in *alle* mogelijke werelden waar is, is het zeker in de echte wereld waar) en we hebben net in de vorige sectie gezien dat er modale logica's zijn waar dit principe niet in zijn algemeenheid geldt. We moeten dus iets ingewikkelders met de mogelijke werelden doen.

Het extra ingrediënt wat nog nodig is, is dat we moeten vertellen welke werelden een wereld kan 'zien'. We zullen voor iedere wereld een verzameling werelden hebben die *toegankelijk* vanuit deze wereld zijn. En de betekenis van $\Box f$ zal dan zijn dat $\Box f$ waar is in een wereld x, precies als f waar is in alle werelden toegankelijkwereld vanuit x, dat is, in alle werelden die x kan zien. En $\Diamond f$ is dan waar in de wereld x als f waar is in minstens *één* wereld die x kan zien. Dit wordt later formeel gemaakt in definitie 5.6.

Als je aan temporele logica denkt is deze toegankelijkheidsrelatie tussen werelden niet zo gek. In een mogelijke werelden-semantiek voor temporele logica zijn de werelden toegankelijk vanuit een wereld die wereld zelf plus de werelden die na die wereld komen. En dan betekent $\Box f$ dus dat f vanaf nu zal gelden, inclusief op dit moment.

Een verzameling mogelijke werelden samen met een opvolger-relatie heet een *Kripke-model*, naar de Amerikaanse filosoof en logicus Saul Kripke. We zullen nu een nette wiskundige definitie van Kripke-modellen geven.

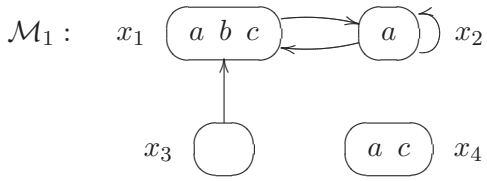
Definitie 5.3 Een *Kripke-model* $\mathcal{M} = \langle W, R, V \rangle$ bestaat uit:

- een niet-lege verzameling W van *werelden*

- voor iedere wereld $x \in W$ een verzameling $R(x) \subseteq W$ van *opvolgers* van x De relatie R heet de *toegankelijkheidsrelatie*.
- voor iedere wereld $x \in W$ een verzameling *atomaire proposities* $V(x)$ die in de wereld x waar zijn

We zullen een Kripke-model tekenen als een gerichte graph. We tekenen de werelden als rondjes, we tekenen pijlen tussen werelden om aan te geven welke werelden vanuit een wereld toegankelijk zijn, en we schrijven de atomen die in de wereld waar zijn in de rondjes.

Voorbeeld 5.4 Hier is een voorbeeld van een Kripke-model, dat we $\mathcal{M}_1$ zullen noemen:



In dit model hebben we dus vier werelden $W = \{x_1, x_2, x_3, x_4\}$. De toegankelijkheidsrelatie van dit Kripke-model is:

$$\begin{array}{lcl} R(x_1) & = & \{x_2\} \\ R(x_2) & = & \{x_1, x_2\} \\ R(x_3) & = & \{x_1\} \\ R(x_4) & = & \emptyset \end{array}$$

en de werelden maken de volgende atomaire proposities waar:

$$\begin{array}{lcl} V(x_1) & = & \{a, b, c\} \\ V(x_2) & = & \{a\} \\ V(x_3) & = & \emptyset \\ V(x_4) & = & \{a, c\} \end{array}$$

We hebben dus dat de proposities a, b en c waar zijn in wereld x_1, maar dat propositie d *niet* waar is in wereld x_1.

We moeten nu vertellen wat het betekent dat een formule van de modale logica f waar is in een wereld x uit een Kripke-model $\mathcal{M}$. Eerst de notatie:

Definitie 5.5 De bewering "een formule f van de modale logica is waar in een wereld x uit een Kripke-model $\mathcal{M}$" noteren we als

$$\mathcal{M}, x \Vdash f$$

Als er geen onduidelijkheid is over het model, wordt dit afgekort tot

$$x \Vdash f$$

En als $x \Vdash f$ niet geldt schrijven we $x \nVdash f$, met een streepje door het $\Vdash$ teken.

Waarheid speelt zich meestal alleen binnen een wereld af en komt dan overeen met de propositielogica. De enige formules waarbij we met meer dan één wereld tegelijk te maken hebben zijn die waarin een modale operator voorkomt. Deze observatie leidt tot de volgende definitie, waarmee voor gegeven formules f kan worden vastgesteld of $x \Vdash f$ al dan niet geldt:

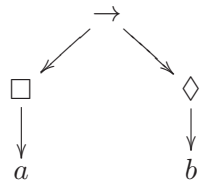
Definitie 5.6 Bekijk het Kripke-model $\mathcal{M} = \langle W, R, V \rangle$. Zij $x \in W$ en zij p een propositieletter en f en g willekeurige formules van de modale logica. Dan geldt:

$$\begin{aligned}
x \Vdash p &\quad \text{desda} \quad p \in V(x) \\
x \Vdash \neg f &\quad \text{desda} \quad x \nVdash f \\
x \Vdash f \wedge g &\quad \text{desda} \quad x \Vdash f \text{ en } x \Vdash g \\
x \Vdash f \vee g &\quad \text{desda} \quad x \Vdash f \text{ of } x \Vdash g \\
x \Vdash f \to g &\quad \text{desda} \quad \text{als } x \Vdash f \text{ dan } x \Vdash g \\
x \Vdash f \leftrightarrow g &\quad \text{desda} \quad (x \Vdash f \text{ desda } x \Vdash g)
\end{aligned}$$

$$\begin{aligned}
x \Vdash \Box f &\quad \text{desda} \quad \text{voor alle } y \in R(x) \text{ geldt } y \Vdash f \\
x \Vdash \Diamond f &\quad \text{desda} \quad \text{er is minstens één } y \in R(x) \text{ waarvoor geldt dat } y \Vdash f
\end{aligned}$$

Opmerking 5.7 De *modale* essentie zit hem natuurlijk in de laatste twee regels. De overige regels komen precies overeen met de waarheidstabellen uit de propositielogica.

We zullen nu gaan kijken of $\Box a \to \Diamond b$ waar is in wereld x_1 van ons Kripke-model $\mathcal{M}_1$. Deze formule heeft als structuur:



We moeten ons dus eerst afvragen wat de waarheidswaarden van $\Box a$ en $\Diamond b$ zijn in wereld x_1. Voor $\Box a$ moeten we naar *alle* opvolgers van x_1 kijken, dat wil zeggen naar x_2. Daar is a inderdaad waar, ofwel $x_2 \Vdash a$, en dus is $\Box a$ waar in wereld x_1, ofwel $x_1 \Vdash \Box a$. Om $\Diamond b$ in wereld x_1 waar te maken moet er een opvolger van x_1 zijn waarin b waar is, maar de enige opvolger van x_1 is x_2 en daarin geldt $x_2 \nVdash b$. Dus hebben we $x_1 \nVdash \Diamond b$.

We gebruiken nu de waarheidstabel van de implicatie. We hebben dat $\Box a$ waar is in x_1 en dat $\Diamond b$ onwaar is in x_1. En dus is $\Box a \to \Diamond b$ ook onwaar in x_1 ofwel:

$$x_1 \nVdash \Box a \to \Diamond b$$

Ga nu zelf na in welke werelden van $\mathcal{M}_1$ de formule $\Box a$ waar is. Als je het goed hebt gedaan, zie je dat deze formule voor alle werelden van $\mathcal{M}_1$ waar is. We schrijven dit als $\mathcal{M}_1 \models \Box a$

Definitie 5.8 Een formule f heet *waar in een Kripke-model* $\mathcal{M}$ als voor alle werelden x in $\mathcal{M}$ geldt dat $\mathcal{M}, x \Vdash f$. De notatie hiervoor is:

$$\mathcal{M} \models f$$

Merk op dat er onderscheid wordt gemaakt tussen $\models$ en $\Vdash$!

Opmerking 5.9 Merk op dat in ons voorbeeld in wereld x_3 de formule a *niet* waar is, oftewel $x_3 \nVdash a$. Evenwel hadden we gezien dat wel geldt dat $\mathcal{M}_1 \Vdash \Box a$. Er geldt in een Kripke-model dus niet dat als $\Box f$ waar is, dat f dan in alle mogelijke werelden van dat model waar moet zijn!

Opgave 5.G Ga van de volgende formules f na in welke werelden x van ons voorbeeld van een Kripke-model ze waar zijn, ofwel, voor welke x er geldt dat $x \Vdash f$. Welke zijn in het hele model waar, ofwel, voor welke geldt $\mathcal{M}_1 \models f$?

(i) a
(ii) $\Box a$
(iii) $\Diamond a$
(iv) $\Box a \to a$
(v) $\Box a \to \Box\Box a$
(vi) $a \to \Box\Diamond a$

Opgave 5.H (i) Bestaat er een Kripke-model $\mathcal{M}_2$ zodat $\mathcal{M}_2 \models a$ maar $\mathcal{M}_2 \not\models \Box a$? Als je antwoord 'ja' is: geef een voorbeeld van zo'n model $\mathcal{M}_2$. Als je antwoord 'nee' is: waarom niet?

(ii) Bestaat er een Kripke-model $\mathcal{M}_3$ zodat $\mathcal{M}_3 \models a$ maar $\mathcal{M}_3 \not\models \Diamond a$? Als je antwoord 'ja' is: geef een voorbeeld van zo'n model $\mathcal{M}_3$. Als je antwoord 'nee' is: waarom niet?

Je kunt beperkingen opleggen aan welke Kripke-modellen je acceptabel vindt. Een aantal van dat soort beperkingen correspondeert met de axiomaschema's uit de vorige sectie. Zo kun je eisen dat er van iedere wereld x een pijl naar zichzelf moet zijn, ofwel dat $x \in R(x)$ voor alle $x \in W$. Een Kripke-model dat aan deze eigenschap voldoet heet *reflexief*. De verzameling van de reflexieve Kripke-modellen correspondeert met het axiomaschema T ofwel met $\Box f \to f$. Of je kunt eisen dat er uit iedere wereld minstens één pijl moet vertrekken. Dit heet een *serieel* Kripke-model. Deze Kripke-modellen corresponderen met het schema D ofwel met $\Box f \to \Diamond f$. Merk op dat ons voorbeeld $\mathcal{M}_1$ reflexief noch serieel is. (Waarom?)

Uit de lijst axiomaschema's is er één die door *alle* Kripke-modellen waar wordt gemaakt. Dit is het axiomaschema K:

$$\Box(f \to g) \to (\Box f \to \Box g)$$

Opgave 5.I (i) Bestaat er een Kripke-model $\mathcal{M}_4$ dat serieel is maar niet reflexief? Als je antwoord 'ja' is: geef een voorbeeld van zo'n model $\mathcal{M}_4$. Als je antwoord 'nee' is: waarom niet?

(ii) Bestaat er een Kripke-model $\mathcal{M}_5$ dat reflexief is maar niet serieel? Als je antwoord 'ja' is: geef een voorbeeld van zo'n model $\mathcal{M}_5$. Als je antwoord 'nee' is: waarom niet?

5.6 Temporele logica

We zullen nu een specifieke temporele logica in meer detail bekijken. Dit is de logica LTL, voor *Linear Time Logic*. Dit is de logica van de SPIN model checker.[10]

Een *model checker* is een computerprogramma om een stuk software te analyseren. Om een model checker te gebruiken moet je eerst een *eindige automaat* maken die het gedrag van de software beschrijft. Vervolgens voer je die automaat in het systeem in, samen met een formule van de temporele logica van het systeem die een eigenschap van de automaat beschrijft waarvan je graag wil weten of hij geldt. De model checker gaat vervolgens de automaat analyseren en vertelt je of de formule waar is of niet.

Een voorbeeld van een formule die je een model checker zou kunnen laten checken is

De automaat komt telkens weer terug in toestand q_{42}.

[10]SPIN staat voor 'Simple Promela INterpreter' en Promela voor 'PRocess MEta LAnguage', een taal waarin eindige automaten kunnen worden gespecificeerd. SPIN is één van de bekendste model checkers en is ontwikkeld door de Nederlander Gerard Holzmann bij Bell Laboratories in Amerika.

(Een dergelijke eigenschap heet 'liveness', levendigheid, want je zegt dat de automaat steeds actief genoeg blijft om weer in de toestand terug te komen.) Een formule van de temporele logica die dit zou kunnen beschrijven is

$$\Box\Diamond q_{42}$$

Je moet dit lezen als

Vanaf nu geldt steeds dat er weer een moment komt waarop we weer in q_{42} zijn.

Je zou kunnen denken dat een model checker niet iets heel ingewikkelds doet. De complicatie is dat zelfs automaten die uit relatief weinig toestanden bestaan, al snel leiden tot een gigantisch aantal situaties dat gecontroleerd moet worden. Dit heet de *toestandsexplosie*.

Definitie 5.10 In LTL worden de modale operatoren niet als $\Box$ en $\Diamond$ geschreven, maar als de hoofdletters $\mathcal{G}$ en $\mathcal{F}$. We hebben:

modaal	LTL	afkorting	betekenis
$\Box f$	$\mathcal{G}\, f$	$\mathcal{G}$lobally	vanaf nu (inclusief nu) geldt f altijd
$\Diamond f$	$\mathcal{F}\, f$	$\mathcal{F}$uture	er komt een moment dat f geldt (of het geldt nu al)

LTL heeft ook nog andere operatoren dan de modale:

LTL	afkorting	betekenis
$\mathcal{X}\, f$	ne$\mathcal{X}$t	na precies één stap geldt f
$f\, \mathcal{U}\, g$	$\mathcal{U}$ntil	f geldt tot g geldt en er komt een moment dat g geldt
$f\, \mathcal{W}\, g$	$\mathcal{W}$eak until	f geldt tot g geldt, of f geldt altijd
$f\, \mathcal{R}\, g$	$\mathcal{R}$elease	g geldt tot f geldt en op dat moment geldt g ook nog, of g geldt altijd

Zoals je ziet zijn er in LTL ook operatoren die twee formules combineren. De operator $\mathcal{X}\, f$ wordt ook wel met een klein rondje als $\circ f$ geschreven.

Opgave 5.J Druk de volgende zinnen uit in LTL. Je mag in je vertalingen alle operatoren van LTL gebruiken.

(i) *Er komt een moment waarna de uitspraak a altijd waar zal zijn.*
(ii) *De uitspraken a en b zijn om de beurt waar.*
(iii) *Telkens als a waar wordt, dan is b na enige tijd ook waar.*

Enkele LTL operatoren kunnen in termen van elkaar kunnen worden gedefinieerd. Zo heb je bijvoorbeeld:

$$\begin{aligned} f\, \mathcal{U}\, g &\quad \text{is equivalent aan} \quad (f\, \mathcal{W}\, g) \wedge \mathcal{F}\, g \\ f\, \mathcal{W}\, g &\quad \text{is equivalent aan} \quad (f\, \mathcal{U}\, g) \vee \mathcal{G}\, f \\ f\, \mathcal{R}\, g &\quad \text{is equivalent aan} \quad \neg(\neg f\, \mathcal{U}\, \neg g) \end{aligned}$$

Opgave 5.K Definieer de operator $\mathcal{U}$ in termen van $\mathcal{R}$.

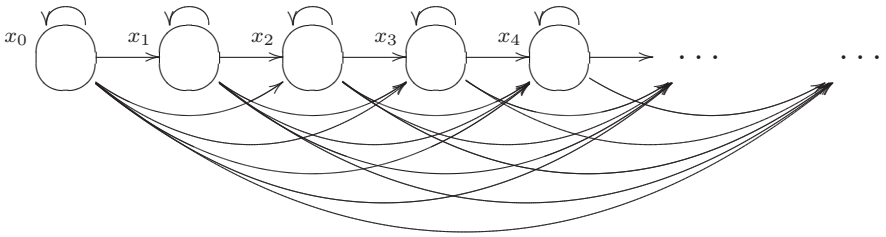
Opgave 5.L Definieer de operator $\mathcal{G}$ en $\mathcal{F}$ in termen van $\mathcal{U}$ en $\mathcal{R}$. Je mag hierbij de proposities $\top$ en $\bot$ gebruiken die respectievelijk altijd waar en altijd onwaar zijn.

Opgave 5.M Laat zien dat je alle LTL operatoren (behalve $\mathcal{X}$) kunt definiëren met de $\mathcal{W}$ operator. Je mag hierbij weer de proposities $\top$ en $\bot$ gebruiken.

De werelden in de Kripke-modellen van LTL corresponderen met de tikken van een klok. Een andere manier om dit te zeggen is dat er een wereld is voor ieder natuurlijk getal. De verzameling werelden is dus:

$$W = \{x_i \mid i \in \mathbb{N}\}$$

Verder zijn vanuit wereld x_i precies de werelden x_j toegankelijk waarvoor $i \leq j$. Het enige waarin de Kripke-modellen voor LTL dus van elkaar verschillen is in welke atomaire proposities waar worden gemaakt in de verschillende werelden. Een plaatje van een LTL Kripke-model is:



Deze formulering in termen van kloktikken en werelden zorgt ervoor dat we de LTL operatoren uit definitie 5.10 op een formelere manier kunnen definiëren.

Definitie 5.11 Gegeven een Kripke-model $\langle W, R, V\rangle$ met W en R als hierboven.

$x_i \Vdash \mathcal{G}f$	voor alle $j \geq i$ geldt $x_j \Vdash f$
$x_i \Vdash \mathcal{F}f$	er is een $j \geq i$ met $x_j \Vdash f$
$x_i \Vdash \mathcal{X}f$	$x_{i+1} \Vdash f$
$x_i \Vdash f\,\mathcal{U}\,g$	er is een $j \geq i$ met $x_j \Vdash g$ en voor alle $k \in \{i, i+1, \ldots, j-1\}$ geldt $x_k \Vdash f$
$x_i \Vdash f\,\mathcal{W}\,g$	ofwel er is een $j \geq i$ met $x_j \Vdash g$ en voor alle $k \in \{i, i+1, \ldots, j-1\}$ geldt $x_k \Vdash f$, ofwel voor alle $l \geq i$ geldt $x_l \Vdash f$
$x_i \Vdash f\,\mathcal{R}\,g$	ofwel er is een $j \geq i$ met $x_j \Vdash f$ en voor alle $k \in \{i, i+1, \ldots, j-1, j\}$ geldt $x_k \Vdash g$, ofwel voor alle $l \geq i$ geldt $x_l \Vdash g$

Met deze formalisatie kunnen we desgewenst de eerder genoemde equivalenties ook bewijzen.

Opgave 5.N Beschouw het LTL Kripke-model $\mathcal{M}_6 = \langle W, R, V\rangle$. Dus met $W = \{x_i \mid i \in \mathbb{N}\}$ en $x_j \in R(x_i)$ als $i \leq j$. Definieer V nu als volgt:

$$a \in V(x_i) \quad \text{desda} \quad i \text{ is een tweevoud}$$
$$b \in V(x_i) \quad \text{desda} \quad i \text{ is een drievoud}$$

Ga na of de volgende eigenschappen gelden:

(i) $x_1 \Vdash \mathcal{F}(a \wedge b)$
(ii) $x_6 \Vdash \mathcal{G}(a \vee b)$
(iii) $\mathcal{M}_6 \Vdash \mathcal{G}\mathcal{F}(a\,\mathcal{U}\,b)$

Opgave 5.O Welke van de axiomaschema's uit de tabel op pagina 65 gelden in LTL?

5.7 Belangrijke begrippen

Index