TECHNISCHE HOGESCHOOL DELFT
Onderafdeling der Wiskunde & Informatica
Vakgroep Informatica
Julianalaan 132
2628 BL DELFT

College dictaat

a 137

THEORETISCHE INFORMATICA I

Grammatica's, automaten en formele talen

P.R.J. Asveld
L.C. van der Gaag
R. Sommerhalder

Delft, april 1984
Tweede verbeterde druk: september 1984

INHOUDSOPGAVE

0. INLEIDING

Bij de eerste kennismaking met de informatica wordt men geconfronteerd met een werktuig (computer), waarmee allerlei taken uitgevoerd kunnen worden, mits men in staat is om op ondubbelzinnige wijze deze taken te omschrijven (de computer te programmeren). Dan komen onmiddellijk twee vragen naar voren:

1. Welke taken kunnen met dit type werktuigen worden verricht en anderszijds, welke problemen kunnen niet met de computer opgelost worden?
2. Hoe moeten taken worden gespecificeerd opdat zij ook daadwerkelijk door de computer uitgevoerd kunnen worden?

De taken, die met behulp van een computer kunnen worden verricht, hebben betrekking op abstracte, niet-materiële zaken. De objecten, waar computers mee werken, zijn symboolrijen. De gebruiker kan deze symboolrijen uitleggen als namen van personen of dingen of van abstracties zoals getallen. De vraag "Welke taken zijn uitvoerbaar?" is dan ook een wiskundige vraag, namelijk: welke transformaties van symboolrijen naar symboolrijen zijn daadwerkelijk uitvoerbaar, of, op iets hoger abstractieniveau: welke functies van N naar N zijn berekenbaar? In hoofdstuk 2 komen we terug op de beantwoording van deze vraag en bespreken we een aantal problemen, die van practisch belang zijn en waarvan kan worden aangetoond, dat ze niet met behulp van computers kunnen worden opgelost. In het college al74 Theoretische Informatica II wordt dit onderwerp verder uitgediept.

De taken, die een computer dient uit te voeren, worden gespecificeerd met behulp van kunstmatige talen zoals ALGOL-60, PASCAL, ADA, ALGOL-68. Een kunstmatige taal wordt gedefinieerd door een stelsel van zinsbouwregels, waarmee alle zinnen van de taal geconstrueerd of afgeleid kunnen worden. Natuurlijke talen (Nederlands, Engels, ...) daarentegen zijn als zodanig een gegeven; de regels van de zinsbouw vormen slechts een model, voor (een deel van) de bestaande natuurlijke taal.

Formele talen, dat zijn eindige of oneindige verzamelingen symboolrijen, zijn wiskundige modellen van zowel natuurlijke talen als van kunstmatige talen. Bestaande natuurlijke talen en bruikbare kunstmatige talen zijn

oneindig: er is geen bovengrens aan het aantal zinnen, dat geformuleerd kan worden in de Nederlandse taal; evenmin is er een bovengrens aan het aantal programma's, dat geschreven kan worden in PASCAL.

Een belangrijk deel van dit dictaat handelt over formele talen. In het bijzonder zal er veel aandacht besteed worden aan methoden voor het specificeren van formele talen. Een specificatie moet een eindig object zijn, immers ook de specificatie van een kunstmatige taal als PASCAL is eindig. In dit dictaat ligt de nadruk vooral op de volgende twee manieren van specificatie:

(1) Grammatica's. Dit zijn eindige wiskundige objecten met behulp waarvan zinsbouwregels worden vastgelegd en tevens de wijze waarop die regels gebruikt moeten worden. Een belangrijke vraag daarbij is, wat de relatie is tussen de vorm van de regels van een grammatica en het uitdrukkingsvermogen van de door de grammatica voortgebrachte taal.

(2) Automaten. Dit zijn eindige wiskundige objecten, die model staan voor programma's, die bepalen of een gegeven symboolrij tot een bepaalde taal (bijvoorbeeld PASCAL) behoort. Hierbij is een belangrijke vraag wat de relatie is tussen grammatica's en automaten.

Volgens de bovenstaande zeer globale schets van dit college, beperken we ons tot een wiskundige theorie, die handelt over één aspect van natuurlijke en kunstmatige talen, namelijk de zinsbouw (syntaxis). Andere aspecten, zoals de betekenis (semantiek) van diverse taalconstructies komen slechts heel vluchtig of in het geheel niet aan de orde. Ook het passend gebruik van de taal (pragmatiek) - bij het programmeren van het grootste belang - komt in het geheel niet ter sprake.

Elementaire begrippen en notaties

In dit dictaat gebruiken we de volgende notationele conventie met betrekking tot de verzamelingentheorie:

- elementen worden aangeduid met kleine (eventueel griekse) letters: $x_1, x_2, \ldots, y, \alpha, \beta, \ldots$

- verzamelingen worden aangegeven met (eventueel griekse) hoofdletters: $V_1, V_2, \ldots, W, X, N, L, \Sigma, \Gamma, \ldots$
- klassen (of families) van verzamelingen (Dit zijn verzamelingen waarvan de elementen weer verzamelingen zijn). Deze worden aangeduid met cursieve hoofdletters: $\mathcal{X}_1, \mathcal{X}_2, \ldots$

Toegepast op de bekende definities levert deze conventie:

$V_1 \cup V_2 = \{x \mid x \in V_1 \text{ of } x \in V_2\}$ — vereniging

$V_1 \cap V_2 = \{x \mid x \in V_1 \text{ en } x \in V_2\}$ — doorsnede

$V_1 \setminus V_2 = \{x \mid x \in V_1 \text{ en } x \notin V_2\}$ — verschil

$V_1 \subseteq V_2$ d.e.s.d. als: als $x \in V_1$, dan $x \in V_2$ — inclusie

$V_1 = V_2$ d.e.s.d. als: $V_1 \subseteq V_2$ en $V_2 \subseteq V_1$ — gelijkheid

$V_1 \subset V_2$ d.e.s.d. als: $V_1 \subseteq V_2$ en $V_1 \neq V_2$ — echte inclusie

$\emptyset$ is de lege verzameling

V_1 en V_2 zijn disjunct d.e.s.d. als: $V_1 \cap V_2 = \emptyset$

V_1 en V_2 zijn onvergelijkbaar d.e.s.d. als: $\exists x_1 x_2 [x_1 \in V_1 \setminus V_2$ en $x_2 \in V_2 \setminus V_1]$

Al deze definities laten zich ook formuleren voor klassen:

$\mathcal{X}_1 \cup \mathcal{X}_2 = \{V \mid V \in \mathcal{X}_1 \text{ of } V \in \mathcal{X}_2\}$ — vereniging

$\mathcal{X}_1 \cap \mathcal{X}_2 = \{V \mid V \in \mathcal{X}_1 \text{ en } V \in \mathcal{X}_2\}$ — doorsnede, etc.

In het licht van deze notationele conventie zijn de volgende symboolrijen syntactisch correct:

$\emptyset \subseteq V_1$; $\quad \emptyset \in \mathcal{X}_1$; $\quad \mathcal{X}_1 \subseteq \mathcal{X}_2$; $\quad x_1, x_2 \in V_1$;

$\{x_1, x_2\} \subseteq V_1$; $\quad \{x_1, x_2\} \in \mathcal{X}_1$; $\quad \{x_1, x_2\} \cap V_2 = \emptyset$;

(Of deze symboolrijen als bewering ook semantisch juist zijn hangt natuurlijk van de waarden van $x_1, x_2, V_1, V_2, \mathcal{X}_1$ en $\mathcal{X}_2$ af).
Onder de aannamen, dat $V_i \subseteq U$, $\mathcal{X}_i \subseteq \mathcal{X}$, en $x_i \in U$ $(i = 1,2)$ voor zekere U en $\mathcal{X}$, zijn de volgende symboolrijen syntactisch incorrect (en dus ook semantisch onjuist, d.w.z. geen betekenis hebben, slechts onzin voorstellen):

$V_1 \in V_2$; $\mathcal{L}_1 \in \mathcal{L}_2$; $V_1 \subseteq \mathcal{L}_2$; $x_1 \subseteq V_1$;
$\{x_1,x_2\} \in V_1$; $\{x_1,x_2\} \cap \mathcal{L}_1 = \emptyset$; $\{x_1,x_2\} \cap V_2 \subseteq \mathcal{L}_2$; etc.

Daarnaast gebruiken we nog:
$P(V) = \{X \mid X \subseteq V\}$, $P(V)$ is de machtsverzameling van V. (Merk op dat $P(\emptyset) = \{\emptyset\}$ en $\{\emptyset\} \neq \emptyset$), en
$V_1 \times V_2 = \{(x_1,x_2) \mid x_1 \in V_1 \text{ en } x_2 \in V_2\}$, $V_1 \times V_2$ is het cartesisch of direct produkt van V_1 en V_2. In het algemeen is het cartesisch produkt niet commutatief; d.w.z. er bestaan verzamelingen V_1 en V_2 zodanig dat $V_1 \times V_2 \neq V_2 \times V_1$.
Een (binaire of dyadische) relatie R tussen de verzamelingen V_1 en V_2 is een deelverzameling van $V_1 \times V_2$: $R \subseteq V_1 \times V_2$. Meestal schrijft men xRy in plaats van $(x,y) \in R$. Het domein D(R) en het codomein C(R) van een relatie R worden respectievelijk gedefinieerd door

$$D(R) = \{x \mid x \in V_1 \text{ en } \exists y \in V_2[xRy]\}$$
$$C(R) = \{y \mid y \in V_2 \text{ en } \exists x \in V_1[xRy]\}$$

Indien $V_1 = V_2 = V$ zegt men, dat $R \subseteq V \times V$ een relatie over V is. Een relatie R over V is:

reflexief als voor alle x in V geldt: xRx,
symmetrisch als voor alle x en y in V geldt: als xRy, dan yRx,
anti-symmetrisch als voor alle x en y in V geldt: als xRy en yRx, dan $x = y$,
transitief als voor alle x,y en z in V geldt: als xRy en yRz, dan xRz.

Een equivalentierelatie is een relatie, die reflexief, symmetrisch en transitief is. Een partiële ordening is een relatie, die reflexief, anti-symmetrisch en transitief is. Een totale ordening R over V is een partiële ordening, waarvoor geldt dat voor alle x en y in V: xRy of yRx.

0.1. Voorbeelden

(1) $x = y$ is voor elke verzameling V een equivalentierelatie.
(2) $x \equiv y$ mod p is voor elke vaste $p \in N \setminus \{0\}$ een equivalentierelatie over N (N is de verzameling der natuurlijke getallen).

(3) $x|y$ (d.w.z. "x deelt y") is een partiële ordening over N.
(4) $x \leq y$ is een totale ordening over N. •

Voor een relatie $R \subseteq V_1 \times V_2$ definieert men de inverse relatie R^{-1} van R als volgt:

$$R^{-1} \subseteq V_2 \times V_1 \text{ met } R^{-1} = \{(x,y) \mid yRx\}.$$

Een functie $f: V_1 \to V_2$ is een relatie $f \subseteq V_1 \times V_2$ zodanig, dat voor alle $x \in V_1$ en $y_1, y_2 \in V_2$ geldt:

$$\text{als } xfy_1 \text{ en } xfy_2, \text{ dan } y_1 = y_2.$$

Voor functies schrijft men meestal $f(x) = y$ in plaats van xfy.

Een functie $f: V_1 \to V_2$ heet <u>totaal</u> als $D(f) = V_1$. Als f niet noodzakelijkerwijs totaal is, dan zegt men wel dat $f: V_1 \to V_2$ een <u>partiële functie</u> is. $D(f) \subseteq V_1$, en voor alle $x \in V_1 \backslash D(f)$ is $f(x)$ ongedefinieerd, notatie $f(x)\uparrow$ ($f(x)\downarrow$ betekent dat f voor x gedefinieerd is, dus dat $x \in D(f)$).

Een functie $f: V_1 \to V_2$ is <u>injectief</u> als voor alle x_1 en x_2 in $D(f)$ geldt: als $x_1 \neq x_2$, dan $f(x_1) \neq f(x_2)$ (of, hetgeen hiermee gelijkwaardig is: als $f(x_1) = f(x_2)$, dan $x_1 = x_2$). Een functie $f: V_1 \to V_2$ heet <u>surjectief</u> indien $C(f) = V_2$, en <u>bijectief</u> als f zowel injectief als surjectief is.

Als f een functie is, is f^{-1} niet altijd een functie (maar natuurlijk wel een relatie). Indien f injectief is, dan is $f^{-1}: V_2 \to V_1$ een partiële functie. De inverse van een bijectieve functie, is weer een bijectieve functie (Ga dit na).

Het cartesisch product van verzamelingen kan uitgebreid worden tot een willekeurig aantal factoren; in het bijzonder geldt voor alle $n \geq 1$:

$$A^n = \underbrace{A \times A \times \ldots \times A}_{n \text{ maal}} = \{(a_1, a_2, \ldots, a_n) \mid a_i \in A,\ 1 \leq i \leq n\}$$

terwijl $A^0 = \{\emptyset\}$.

Een (eindig) type τ is een element van N^n voor een of andere $n \geq 0$; dus $\tau = (m_1,\ldots,m_n)$. Een algebra van type $\tau = (m_1,\ldots,m_n)$ is een tweetal (A,F) met

- A is een verzameling, en
- $F = (f_1,\ldots,f_n)$ is een geordend n-tal van functies, de operaties van de algebra, zodanig dat voor alle i $(1 \leq i \leq n)$ $f_i: A^{m_i} \to A$ een totale functie is (m_i wordt de rang van de operatie f_i genoemd).

Merk op dat als $m_j = 0$, dan geldt $f_j: \{\emptyset\} \to A$, d.w.z. f_j is een constante functie, waarvan de waarde $f_j(\emptyset)$ ook wordt aangegeven met f_j; dus $f_j \in A$ als $m_j = 0$).

In dit college komen twee soorten algebra's veelvuldig naar voren: namelijk de monoïde en de halfring.
Een monoïde $(M,(\bullet,1))$ is een algebra van het type (2,0), die voldoet aan

$x \bullet (y \bullet z) = (x \bullet y) \bullet z$	$\bullet$ is associatief
$x \bullet 1 = 1 \bullet x = x$	1 is het neutrale element

voor alle x, y en z in M (Men gebruikt de infix-notatie $x \bullet y$ in plaats van de prefix-notatie $\bullet(x,y)$). M heet commutatief als voor alle x en y in M geldt: $x \bullet y = y \bullet x$.

0.2. Voorbeelden

(1) $(N,(+,0))$, $(N,(\times,1))$, en voor elke verzameling V zijn $(P(V),(\cup,\emptyset))$ en $(P(V),(\cap,V))$ commutatieve monoïden.

(2) Zij voor elke verzameling V, per definitie $V^V = \{f \mid f: V \to V\}$. Dan is $(V^V,(\bullet,1))$, waarbij

$$f \bullet g = h \text{ met } h(x) = g(f(x)) \text{ voor alle } x \text{ in } V,$$
$$\text{en } 1(x) = x \text{ voor alle } x \text{ in } V,$$

een niet-commutatieve monoïde. •

Een halfring $(H,(\oplus,\odot,0,1))$ is een algebra van het type (2,2,0,0), zo dat voor alle $x,y,z \in H$ geldt:

$x \oplus (y \oplus z) = (x \oplus y) \oplus z$	$\oplus$ is associatief
$x \oplus y = y \oplus x$	$\oplus$ is commutatief
$x \oplus 0 = 0 \oplus x = x$	0 is het neutrale element m.b.t. $\oplus$
$x \odot (y \odot z) = (x \odot y) \odot z$	$\odot$ is associatief
$x \odot 1 = 1 \odot x = x$	1 is het neutrale element m.b.t. $\odot$
$x \odot 0 = 0 \odot x = 0$	0 is het nulelement m.b.t. $\odot$
$x \odot (y \oplus z) = (x \odot y) \oplus (x \odot z)$	$\odot$ is linksdistributief over $\oplus$
$(x \oplus y) \odot z = (x \odot z) \oplus (y \odot z)$	$\odot$ is rechtsdistributief over $\oplus$

(We gebruiken weer de infix-notatie in plaats van de prefix-notatie). H heet <u>commutatief</u> als voor alle x en y in H geldt: $x \odot y = y \odot x$.

0.3. <u>Voorbeelden</u>

$(N,(+,\times,0,1))$ en $(P(V),(\cup,\cap,\emptyset,V))$, V een willekeurige verzameling, zijn commutatieve halfringen. Beschouw

$\mathcal{M} = \{f \mid f: N \to N,\ f(0) = 0 \text{ en als } m \leq n, \text{ dan } f(m) \leq f(n)\}$

$\max(f,g) = h$	met $h(n) = \max\{f(n),g(n)\}$	voor alle $n \in N$,
$f \circ g = h$	met $h(n) = g(f(n))$	voor alle $n \in N$,
$\underline{0} \in \mathcal{M}$	met $\underline{0}(n) = 0$	voor alle $n \in N$,
$\underline{1} \in \mathcal{M}$	met $\underline{1}(n) = n$	voor alle $n \in N$,

dan is $(\mathcal{M},(\max,\circ,\underline{0},\underline{1}))$ een niet-commutatieve halfring (Ga dit na). •

Zij $\tau = (m_1,\ldots,m_n)$ een type en $(A,(f_1,\ldots,f_n))$ en $(B,(g_1,\ldots,g_n))$ algebra's van type τ. Een functie $h: A \to B$, die voldoet aan $h(f_i(a_1,a_2,\ldots,a_{m_i})) = g_i(h(a_1),h(a_2),\ldots,h(a_{m_i}))$ voor alle $a_j \in A$ $(1 \leq j \leq m_i)$ en alle i $(1 \leq i \leq n)$, is een <u>homomorfisme</u> (of homomorfie; Eng. "homomorphism") van A in B. In het bijzonder is een functie $h: M_1 \to M_2$ waarin $(M_1,(o_1,1_1))$ en $(M_2,(o_2,1_2))$ monoïden zijn, een (monoïde-)homomorfisme van M_1 in M_2, indien voor alle x en y in M geldt, dat $h(x\ {}^{o_1}\ y) = h(x)\ o_2\ h(y)$ en $h(1_1) = 1_2$. Evenzo is een functie $h: H_1 \to H_2$, waarin $(H_1,(\oplus_1,\odot_1,0_1,1_1))$ en $(H_2,(\oplus_2,\odot_2,0_2,1_2))$ halfringen zijn een (halfring-)homomorfisme van H_1 in H_2, indien voor alle x en y in H_1 geldt, dat $h(x \oplus_1 y) = h(x) \oplus_2 h(y)$, $h(x \odot_1 y) = h(x) \odot_2 h(y)$, $h(0_1) = 0_2$ en $h(1_1) = 1_2$.

0.4. Voorbeelden

(1) Zij $h: N \to N$ met $h(n) = 2^n$ voor alle $n \in N$. Dan is h een (monoïde-) homomorfisme van $(N,(+,0))$ in $(N,(\times,1))$ want $h(m+n) = 2^{m+n} = 2^m \times 2^n = h(m) \times h(n)$ voor alle $m,n \in N$ en $h(0) = 2^0 = 1$.

(2) Zij de operaties max, $\circ$, $\underline{0}$ en $\underline{1}$ als in voorbeeld 0.3. en $\mathcal{Y} = \{f \mid f: N \to N,\ f(0) = 0,$ en $f(n) \in \{0,n\}$ voor alle $n \in N\}$. Dan is $(\mathcal{Y},(\max,\circ,\underline{0},\underline{1}))$ een commutatieve halfring (Toon dit aan), en de functie $h: \mathcal{Y} \to P(N)$ met $h(f) = \{x \in N \mid f(x) \neq 0\}$ een (halfring-) homomorfisme van $(\mathcal{Y},(\max,\circ,\underline{0},\underline{1}))$ in $(P(N),(\cup,\cap,\emptyset,N\setminus\{0\}))$, daar voor alle f en g in $\mathcal{Y}$ geldt:

$$
\begin{aligned}
h(\max(f,g)) &= \{x \mid (\max(f,g))(x) \neq 0\} = \\
&= \{x \mid f(x) \neq 0 \text{ of } g(x) \neq 0\} = \\
&= \{x \mid f(x) \neq 0\} \cup \{x \mid g(x) \neq 0\} = h(f) \cup h(g), \\
h(f \circ g) &= \{x \mid (f \circ g)(x) \neq 0\} = \{x \mid g(f(x)) \neq 0\} = \\
&= \{x \mid f(x) \neq 0 \text{ en } g(x) \neq 0\} = \\
&= \{x \mid f(x) \neq 0\} \cap \{x \mid g(x) \neq 0\} = h(f) \cap h(g), \\
h(\underline{0}) &= \{x \mid \underline{0}(x) \neq 0\} = \emptyset \\
h(\underline{1}) &= \{x \mid \underline{1}(x) \neq 0\} = N\setminus\{0\}.
\end{aligned}
$$

•

Een isomorfisme is een bijectief homomorfisme. Twee algebra's heten isomorf als er een isomorfisme van de ene in de andere algebra bestaat. Isomorfe algebra's zijn, op de naamgeving van elementen en operaties na, gelijk.

1. BASISBEGRIPPEN: Talen - Herschrijfsystemen - Grammatica's

1.1. Definitie: Een alfabet V is een eindige, niet-lege verzameling elementen. De elementen van het alfabet V worden letters of symbolen genoemd.

1.2. Voorbeeld

De verzamelingen $V = \{0\}$ en $W = \{0,1,2,3\}$ zijn voorbeelden van alfabetten. Beide alfabetten bevatten de letter 0. ●

1.3. Definitie: Een woord over een alfabet V is een eindige geordende rij van nul of meer letters uit V, waarbij eenzelfde letter meer malen mag voorkomen. Het woord dat uit nul letters bestaat, wordt het lege woord genoemd en wordt aangegeven met het symbool λ. Een woord $\alpha \neq \lambda$ over V kan geschreven worden als $\alpha = a_1 a_2 \ldots a_n$, $a_i \in V$, $1 \leq i \leq n$. Een hieruit gekozen rij opeenvolgende letters $\alpha' = a_j a_{j+1} \ldots a_k$, $a_i \in V$, $1 \leq j \leq i \leq k \leq n$, heet een deelwoord van α.

1.4. Voorbeeld

Laat het alfabet V gegeven zijn door $V = \{a,b,c\}$ dan is α = aabcaa een woord over V; α' = abc en α'' = aab zijn deelwoorden van α, maar β = aba is geen deelwoord van α. ●

1.5. Definitie: De verzameling van alle woorden over een alfabet V wordt aangegeven met V^*; V^+ is de verzameling van alle niet-lege woorden over V. Merk op dat $V^+ = V^* \setminus \{\lambda\}$.

1.6. Definitie: Zij V een alfabet en $\alpha \in V^+$, $\beta \in V^*$. Met de notatie $\#(\alpha,\beta)$ wordt het aantal malen aangegeven dat het woord α als deelwoord in het woord β voorkomt.

1.7. Voorbeeld

Laat het alfabet V gegeven zijn door $V = \{a,b,c\}$, dan is $\#(a,abbca) = 2$, $\#(a,\lambda) = 0$, $\#(c,ac^n) = n$ en $\#(aba,ababa) = 2$. •

1.8. Definitie: Zij V een alfabet. De lengte van een woord $\alpha \in V^*$, aangeduid met $|\alpha|$, is gelijk aan $\sum_{x \in V} \#(x,\alpha)$.

1.9. Voorbeeld

De woorden aabcaa, a en λ zijn woorden over $V = \{a,b,c,d\}$. Er geldt $|aabcaa| = 6$, $|a| = 1$ en $|\lambda| = 0$. •

1.10. Definitie: Als α en β woorden over een alfabet V zijn, dan wordt een nieuw woord $\alpha\beta \in V^*$ gevormd door α en β achter elkaar te schrijven. Dit achter elkaar schrijven wordt concateneren genoemd en $\alpha\beta$ heet de concatenatie van de woorden α en β.

1.11. Oefening

Druk de lengte van het woord $\alpha\beta$ uit in de lengten van de woorden α en β.

Merk op dat concatenatie een associatieve operatie is. Algebraïsch gezien betekent dit dat V^* een monoïde is met concatenatie als binaire operatie en λ als neutraal element.

1.12. Definitie: Een taal L over een alfabet V is een verzameling woorden over V, zodat $L \subseteq V^*$.

1.13. Voorbeeld

De volgende verzamelingen zijn talen over het alfabet $\{0,1\}$.

$L_1 = \emptyset$
$L_2 = \{\lambda\}$
$L_3 = \{0, 1, 11, 000\}$
$L_4 = \{0^n \mid n \text{ is priem}\}$
$L_5 = \{0,1\}^*$

Merk op dat $L_1 \neq L_2$, immers L_1 bevat geen elementen en L_2 bevat precies één element. L_3 is een eindige taal, en L_4 en L_5 zijn oneindige talen. Beschouw

$$D = \{\alpha \mid \exists n \in N[0,\alpha \text{ is de binaire schrijfwijze van } \tfrac{1}{n}]\}$$

D is geen taal over $V = \{0,1\}$, aangezien D ook niet-eindige rijtjes nullen en enen bevat. •

Zoals uit voorbeeld 1.13. blijkt, bestaan er verschillende manieren om een taal L over een gegeven alfabet V te specificeren. Als L een eindige taal is, kunnen alle woorden van L opgesomd worden, zoals in voorbeeld 1.13. bij de taal L_3 het geval is. Een tweede methode voor de specificatie van een taal is de formulering van een voor de woorden uit L kenmerkende eigenschap, zoals bij L_4. We kunnen een taal L ook specificeren door een methode te geven met behulp waarvan de woorden uit L voortgebracht kunnen worden. Daarvoor wordt een voortbrengende (of generatieve) grammatica gebruikt. Het centrale begrip achter zo'n voortbrengende grammatica is het begrip herschrijfsysteem.

1.14. Definitie: Een herschrijfsysteem is een paar H = (V,P), waarin V een alfabet is en P een eindige deelverzameling van $V^* \times V^*$. De elementen $(\alpha,\beta) \in P$ heten herschrijf- of productieregels. In plaats van $(\alpha,\beta) \in P$ schrijft men $\alpha \vec{H} \beta$, of $\alpha \rightarrow \beta$, als duidelijk is om welk herschrijfsysteem het handelt.

Met behulp van een herschrijfsysteem kunnen woorden omgevormd worden tot andere woorden door het toepassen van een herschrijfregel. Beschouw het herschrijfsysteem H = ($\{a,b\}$, $\{ab \to aabb\}$). De herschrijfregel $ab \to aabb$ wordt op het woord α = abaa uit $\{a,b\}^*$ toegepast door het deelwoord ab in α te vervangen door het woord aabb. Het resultaat is een nieuw woord β = aabbaa uit $\{a,b\}^*$. Op het woord α = abab kan de produktieregel $ab \to aabb$ op verschillende manieren toegepast worden:

- de produktieregel $ab \to aabb$ wordt tegelijkertijd op alle deelwoorden ab in α toegepast, resulterend in aabbaabb $\in \{a,b\}^*$;
- de produktieregel wordt op het meest linkse (rechtse) deelwoord ab in α toegepast, resulterend in aabbab (abaabb);
- de produktieregel wordt op precies één, willekeurig gekozen, deelwoord ab in α toegepast. Uit α kan dus zowel het woord aabbab als het woord abaabb gevormd worden.

Afgesproken wordt dat in dit dictaat onder het toepassen van een produktieregel altijd de laatstgenoemde mogelijkheid wordt verstaan.

1.15. Definitie: Zij H = (V,P) een herschrijfsysteem en zij $\alpha,\beta \in V^*$. α leidt direkt β af, notatie $\alpha \underset{H}{\Longrightarrow} \beta$, als er woorden $\alpha_1,\alpha_2,\gamma,\delta \in V^*$ zijn, zodanig dat $\alpha = \alpha_1\gamma\alpha_2$ en $\beta = \alpha_1\delta\alpha_2$ en $\gamma \to \delta \in P$.

1.16. Voorbeeld

Beschouw nogmaals het herschrijfsysteem, H = (V,P) = ($\{a,b\}$,$\{ab \to aabb\}$) Als uitgegaan wordt van het woord ab, is het mogelijk door herhaaldelijk de produktieregel $ab \to aabb$ toe te passen, de woorden aabb, aaabbb, ..., a^nb^n, ... ($n \geq 1$) af te leiden. ●

Let op het verschil tussen $\to$ en $\Longrightarrow$: hiermee worden twee totaal verschillende relaties op V^* aangeduid.

- $\to$ is een eindige binaire relatie op V^*: er zijn eindig veel paren α en β, zodanig dat $\alpha \to \beta$ geldt. Merk op dat $\to$ een eindige relatie moet

zijn opdat een herschrijfsysteem een eindig object is. Elke regel van de vorm $\alpha \to \beta$ geeft als het ware een handelingsvoorschrift aan.

- $\Rightarrow$ is een oneindige binaire relatie op V^*: er zijn oneindig veel paren α en β, zodanig dat $\alpha \Rightarrow \beta$ geldt. De relatie $\Rightarrow$ geeft voor alle mogelijke situaties aan hoe het eindige voorschrift $\to$ in die betreffende situatie wordt uitgevoerd.

Het is onjuist te menen dat bij een gegeven relatie $\to$ op zinvolle wijze slechts één relatie $\Rightarrow$ kan worden gedefinieerd (zie definitie 1.15.). Ook als $\Rightarrow$ gegeven is, ligt daarmee $\to$ in het algemeen niet eenduidig vast.

De relatie $\Rightarrow$ beschrijft het eenmalig toepassen van een van de gegeven produktieregels. Wat betreft het herhaald toepassen van de gegeven produktieregels geldt de volgende

1.17. Definitie:
- het nul maal toepassen van een produktieregel wordt genoteerd als $\overset{0}{\Longrightarrow}$:
  $\alpha \overset{0}{\Rightarrow} \beta$ dan en slechts dan als $\beta = \alpha$.
- het $k > 1$ maal toepassen van een of meer produktieregels wordt genoteerd als $\overset{k}{\Longrightarrow}$:
  $\alpha \overset{k}{\Rightarrow} \beta$ dan en slechts dan als er woorden $\alpha=\gamma_0, \gamma_1, \cdots, \gamma_k=\beta$ zijn zodanig dat voor alle $0 \leq i < k$ geldt $\gamma_i \Rightarrow \gamma_{i+1}$. De rij $\alpha = \gamma_0 \Rightarrow \gamma_1 \Rightarrow \ldots \Rightarrow \gamma_k = \beta$ wordt een afleiding van β uit α genoemd.
- het willekeurig vaak doch tenminste eenmaal toepassen van een of meer produktieregels wordt genoteerd als $\overset{+}{\Longrightarrow}$:
  $\alpha \overset{+}{\Rightarrow} \beta$ dan en slechts dan als er een $k > 1$ is, zodanig dat $\alpha \overset{k}{\Rightarrow} \beta$.
- het willekeurig vaak toepassen van een of meer produktieregels wordt genoteerd als $\overset{*}{\Longrightarrow}$:
  $\alpha \overset{*}{\Rightarrow} \beta$ dan en slechts dan als er een $k > 0$ is, zodanig dat $\alpha \overset{k}{\Rightarrow} \beta$.

1.18. Oefening

Beschouw het herschrijfsysteem $H = (\{a,b\}, \{ab \to \lambda\})$. Bepaal voor welke woorden $\alpha \in \{a,b\}^*$ geldt dat $\alpha \overset{*}{\underset{H}{\Longrightarrow}} \lambda$.

Het herhaald toepassen van de gegeven produktieregels, aangegeven met $\overset{+}{\Rightarrow}$ of $\overset{*}{\Rightarrow}$, sluit als volgt aan bij de wiskundige begrippen transitieve afsluiting en reflexieve afsluiting:

1.19. Definitie: Laat $\sim$ een binaire relatie op een verzameling W zijn, dat wil zeggen $\sim \subseteq W \times W$.

- De reflexieve afsluiting van $\sim$ is de relatie $\underset{r}{\sim} = \sim \cup \{(x,x) \mid x \in W\}$.
- De transitieve afsluiting van $\sim$, notatie $\overset{+}{\sim}$, wordt gegeven door
  (i) $\forall x,y \in W$ als $x \sim y$, dan $x \overset{+}{\sim} y$.
  (ii) $\forall x,y,z \in W$ als $x \overset{+}{\sim} y$ en $y \overset{+}{\sim} z$, dan $x \overset{+}{\sim} z$.
  (iii) geen andere paren (x,y) behoren tot $\overset{+}{\sim}$.
- De reflexieve en transitieve afsluiting van $\sim$ is de relatie $\overset{*}{\sim} = \underset{r}{\overset{+}{\sim}} = \overset{+}{\sim} \cup \{(x,x) \mid x \in W\}$.

De relatie $\sim$ (en in het bijzonder $\rightarrow$, $\Rightarrow$, $\overset{+}{\Rightarrow}$ en $\overset{*}{\Rightarrow}$) is in het algemeen niet symmetrisch (een relatie $\sim \subseteq W \times W$ is symmetrisch als $\forall x,y \in W$: $x \sim y$ dan en slechts dan als $y \sim x$).

1.20. Oefeningen

(1) Op de verzameling $\{1,2,3\}$ is de relatie $\sim$ gedefinieerd door $\sim = \{(1,2),(2,2),(2,3)\}$. Bepaal $\overset{+}{\sim}$ en $\overset{*}{\sim}$.

(2) Beschouw de gerichte graaf met als verzameling knopen $\{1,2,3\}$ en als verzameling takken de verzameling $\sim$. Wat is nu de betekenis van $\overset{+}{\sim}$ en $\overset{*}{\sim}$?

(3) Ga na dat $\overset{+}{\Rightarrow}$ en $\overset{*}{\Rightarrow}$ respectievelijk de transitieve en de reflexieve en transitieve afsluiting van de relatie $\Rightarrow$ aanduiden.

Om van een herschrijfsysteem H = (V,P) tot een grammatica te komen, worden een symbool $S \in V$ en een deelverzameling Σ van V aangegeven en wor-

den in het bijzonder die woorden $w \in \Sigma^*$ beschouwd waarvoor geldt $S \overset{*}{\underset{H}{\Longrightarrow}} w$. Een op zo'n manier aangepast herschrijfsysteem wordt een (voortbrengende) grammatica genoemd.

1.21. Definitie: Een grammatica G is een geordend viertal $G = (V,\Sigma,P,S)$ waarin

- V een eindige niet-lege verzameling is; V wordt het alfabet van G genoemd;
- $\Sigma \subseteq V$ een eindige niet-lege deelverzameling van V is; Σ wordt het eindalfabet van G genoemd en de letters uit Σ worden eindsymbolen genoemd (Eng.: "terminal symbols". In de Engelse literatuur wordt veelal de term "terminals" gehanteerd). De letters uit $V \setminus \Sigma$ worden hulpsymbolen genoemd (Eng.: "nonterminal symbols". In de Engelse literatuur wordt veelal de term "nonterminals" gehanteerd);
- $S \in V \setminus \Sigma$ het begin- of startsymbool van G is;
- P een verzameling produktieregels van G is.

1.22. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica. Een woord $\alpha \in V^*$ heet een zinsvorm van G als $S \overset{*}{\Longrightarrow} \alpha$. Een woord $w \in \Sigma^*$ heet een zin van G als $S \overset{*}{\Longrightarrow} w$. Een zin is dus een zinsvorm die alleen uit eindsymbolen bestaat.

In het navolgende wordt de volgende notatieconventie gehanteerd:

- hulpsymbolen worden aangeduid met hoofdletters, meestal uit het begin van het alfabet (uitzondering: het startsymbool S);
- de kleine letters achter in het alfabet geven zinnen aan;
- met de kleine letters voor in het alfabet en soms ook met ¢, $, #, ... worden eindsymbolen aangeduid;
- zinsvormen waarin hulpsymbolen mogen voorkomen, worden met kleine Griekse letters aangegeven.

1.23. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica. De taal die voortgebracht wordt door G, aangeduid met L(G), is de verzameling $L(G) = \{w \in \Sigma^* \mid S \overset{*}{\Longrightarrow} w\}$.

1.24. Definitie: Als G_1 en G_2 grammatica's zijn, dan heten G_1 en G_2 equivalent als $L(G_1) = L(G_2)$.

1.25. Voorbeeld

Beschouw de grammatica $G = (V,\Sigma,P,S)$, waarin

$V = \{S,a,b\}$
$\Sigma = \{a,b\}$
$P = \{S \to SS,\ S \to aSb,\ S \to bSa,\ S \to \lambda\}$

Bewering: $L(G)$ bestaat uit alle woorden over Σ, die evenveel a's als b's bevatten. Om deze bewering te bewijzen, moet worden aangetoond dat

- $L(G) \subseteq L$
- $L \subseteq L(G)$

waarin $L = \{w \mid \#(a,w) = \#(b,w)\}$ •

Voor een dergelijk bewijs wordt gebruik gemaakt van de invariantenmethode:

1.26. Definitie: Een invariant van een herschrijfsysteem $H = (V,P)$ is een uitspraak U waarvoor geldt

$$\forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \overset{*}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)].$$

1.27. Eigenschap

U is een invariant van het herschrijfsysteem $H = (V,P)$, dan en slechts dan als $\forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \underset{H}{\Rightarrow} \psi,\ \underline{\text{dan}}\ U(\psi)]$.

Bewijs

Het bewijs van deze eigenschap verloopt in twee stappen:

(1) Gegeven: U is een invariant van H = (V,P).
Te bewijzen: $\forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \underset{H}{\Rightarrow} \psi,\ \underline{\text{dan}}\ U(\psi)]$.

Bewijs: dit volgt uit de definitie van $\overset{*}{\Rightarrow}$; immers $\Rightarrow \subseteq \overset{*}{\Rightarrow}$.

(2) Gegeven: $\forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \overset{*}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)]$.
Te bewijzen: U is een invariant van H = (V,P).

Bewijs: Met inductie naar k wordt aangetoond dat uit het gegeven mag worden geconcludeerd

$$\forall\, k > 0, \forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \overset{k}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)].$$

Initialisatiestap: k = 0. Als $\phi \overset{0}{\underset{H}{\Rightarrow}} \psi$, dan $\psi = \phi$, dus

$$\forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \overset{0}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)].$$

Inductieveronderstelling: Neem aan dat voor alle $k < m$, $m > 0$, uit het gegeven mag worden geconcludeerd dat:

$$\forall\, \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \overset{k}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)]$$

Inductiestap: Zij $U(\phi)$ en $\phi \overset{m+1}{\underset{H}{\Rightarrow}} \psi$. Dan is er een ξ zodat $\phi \overset{m}{\underset{H}{\Rightarrow}} \xi \underset{H}{\Rightarrow} \psi$.
Uit de inductieveronderstelling volgt dat $U(\xi)$ geldt, en uit het gegeven volgt vervolgens dat $U(\psi)$ geldt, zodat:

$$\forall\, \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \overset{m+1}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)].$$

Hiermee is aangetoond dat uit het gegeven mag worden geconcludeerd

$$\forall\, k > 0, \forall \phi,\ \psi \in V^*[\underline{\text{als}}\ U(\phi)\ \text{en}\ \phi \overset{k}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)].$$

Zij nu $U(\phi)$ en $\phi \overset{*}{\underset{H}{\Rightarrow}} \psi$. Dan is er een $k \geq 0$, zodanig dat $\phi \overset{k}{\underset{H}{\Rightarrow}} \psi$. Uit

$$\forall k \geq 0, \forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi) \text{ en } \phi \overset{k}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)],$$

volgt $U(\psi)$, zodat

$$\forall \phi,\psi \in V^*[\underline{\text{als}}\ U(\phi) \text{ en } \phi \overset{*}{\underset{H}{\Rightarrow}} \psi,\ \underline{\text{dan}}\ U(\psi)].$$

Hieruit volgt dat U een invariant is voor het herschrijfsysteem H.

Uit (1) en (2) volgt hetgeen te bewijzen was.
Einde bewijs.

In het vervolg wordt voor functies en predikaten gebruik gemaakt van de zogenaamde λ-notatie van Church. Een functie $f: D_1 \times D_2 \times \ldots \times D_n \to C_1 \times C_2 \times \ldots \times C_m$ wordt genoteerd als

$$f = \lambda x_1 \ldots x_n[\text{uitdrukking in } x_1 \text{ tot en met } x_n].$$

Als overal in de uitdrukking in x_1 tot en met x_n voor x_i een waarde $d_i \in D_i$, $1 \leq i \leq n$, wordt ingevuld, en de resulterende uitdrukking wordt geëvalueerd, moet een element $(c_1,\ldots,c_m) \in C_1 \times C_2 \times \ldots \times C_m$ worden verkregen. Dit element $(c_1,\ldots,c_m)$ is het beeld van $(d_1,\ldots,d_n) \in D_1 \times D_2 \times \ldots \times D_n$ onder de functie f.

1.28. Voorbeelden

(1) $f = \lambda xy[x^2 + y^2]$ is een functie $f: N^2 \to N$, waarvoor $\forall (x,y) \in N^2$, $f(x,y) = x^2 + y^2$.

(2) $g = \lambda xy[x \geq y]$ is een functie $g: N^2 \to \{\text{waar, onwaar}\}$, waarvoor geldt $g(x,y) =$ waar, dan en slechts dan als $x \geq y$; g kan men ook als predikaat beschouwen.

(3) $h = \lambda xy[(x+y,y)]$ is een functie $h: N^2 \to N^2$. Deze functie beeldt een paar $(x,y) \in N^2$ af op het paar $(x+y,y) \in N^2$. ●

1.29. Voorbeeld

Beschouw nogmaals de grammatica G uit voorbeeld 1.25. Het bewijs van $L(G) \subseteq L$ verloopt als volgt:

De bewering $U = \lambda\phi[\#(a,\phi) = \#(b,\phi)]$ is een invariant van het aan G ten grondslag liggende herschrijfsysteem $H = (V,P)$. Immers als $U(\phi)$ geldt en $\phi \Rightarrow \psi$, dan

(1) $[\#(a,\psi) = \#(a,\phi) = \#(b,\phi) = \#(b,\psi)] = U(\psi)$, als op het woord ϕ een van de produktieregels $S \to SS$ of $S \to \lambda$ wordt toegepast.

(2) $[\#(a,\psi) = \#(a,\phi)+1 = \#(b,\phi)+1 = \#(b,\psi)] = U(\psi)$ als op het woord ϕ een van de overige produktieregels wordt toegepast.

Aangezien U(S) geldt, volgt voor alle $w \in L(G)$ dat U(w) geldt: U(S) en $S \overset{*}{\Rightarrow} w$, dus U(w). Hieruit volgt dat $L(G) \subseteq L$.

Om $L \subseteq L(G)$ te bewijzen, wordt met behulp van inductie naar de woordlengte 2k, $k \geq 1$, aangetoond dat voor elk woord $w \in L$, met $|w| = 2k$, een afleiding $S \overset{*}{\underset{G}{\Rightarrow}} w$ in G bestaat.

Initialisatiestap: $k = 1$. Er zijn in L precies twee woorden ter lengte 2, namelijk de woorden $w_1 = ab$ en $w_2 = ba$. Aangezien $S \to aSb$, $S \to bSa$ en $S \to \lambda$ produktieregels in P van G zijn, geldt $S \overset{*}{\Rightarrow} ab$ en $S \overset{*}{\Rightarrow} ba$ zodat $w_1, w_2 \in L(G)$.

Inductieveronderstelling: Neem aan dat $\forall k < m$, $m \geq 1$ en voor alle woorden $w \in L$, met $|w| = 2k$, geldt $w \in L(G)$.

Inductiestap: Beschouw $w \in L$, met $|w| = 2m+2$. Dan is er een woord $w' \neq \lambda$, zodanig dat

1. $w = aw'a$, of
2. $w = aw'b$, of
3. $w = bw'a$, of
4. $w = bw'b$.

In de gevallen 2. en 3. is $w' \in L$ en $|w'| = 2m$. Uit de inductieveronderstelling volgt $w' \in L(G)$, maar dan is ook $w \in L(G)$, immers $S \Longrightarrow aSb \overset{*}{\Longrightarrow} aw'b$ in geval 2., en $S \Longrightarrow bSa \overset{*}{\Longrightarrow} bw'a$ in geval 3. In de gevallen 1. en 4. is w te schrijven als $w = w_1w_2$, zodanig dat $w_1, w_2 \in L$, en $w_1 \neq \lambda$ en $w_2 \neq \lambda$ (Ga dit na). Aangezien $|w_1|, |w_2| < 2m$ geldt volgens de inductieveronderstelling $w_1, w_2 \in L(G)$. Maar dan is ook $w \in L(G)$, immers $S \Longrightarrow SS \overset{*}{\Longrightarrow} w_1w_2$. •

1.30. Voorbeeld

In dit voorbeeld wordt de tweetallige schrijfwijze van een natuurlijk getal n bepaald, en wel op twee manieren:
- met een Pascal-programma;
- met een herschrijfsysteem.

Bij het opschrijven van invarianten wordt gebruik gemaakt van de functie $g: \{0,1\}^* \to N$, die als volgt is gedefinieerd:

$$g(\lambda) = 0$$

$$g(b_1 \ldots b_m) = \sum_{i=1}^{m} b_i 2^{m-i} \qquad (b_i \in \{0,1\},\ 1 \leq i \leq m).$$

Het volgende Pascal-programma bepaalt de tweetallige schrijfwijze van een natuurlijk getal n:

invoer : n, een natuurlijk getal
uitvoer: w, een woord over $\{0,1\}^*$, zodanig dat w de tweetallige schrijfwijze van n is.

```
var x;
begin
  w := λ;
  x := n;
  {g(w) + x 2^|w| = n}
  while x > 0 do
  begin
     {g(w) + x 2^|w| = n en x > 0}
     w := (x mod 2)w;
     x := ⌊x/2⌋
     {g(w) + x 2^|w| = n en x ≥ 0}
  end
  {g(w) + x 2^|w| = n en x = 0}
end.
{g(w) = n}
```

De uitspraak $[g(w) + x\ 2^{|w|} = n]$ is voor ieder natuurlijk getal n een invariant voor bovenstaand programma, zoals eenvoudig kan worden aangetoond.

Als de berekening op invoer n termineert, geldt dus inderdaad dat na afloop van de berekening w de tweetallige schrijfwijze van n is. Resteert nog aan te tonen dat voor alle $n > 0$ de door het programma bepaalde berekening op invoer n termineert. De rij van alle waarden die aan de variabele x worden toegekend, begint met n en daalt vervolgens bij elk doorlopen van de while-lus. Aangezien alleen gehele getallen als waarde worden toegekend, wordt na ten hoogste n stappen de waarde 0 bereikt. Dan wordt de while-lus niet langer doorlopen en termineert de berekening.

Het ontwerpen van herschrijfsystemen is vergelijkbaar met het ontwerpen van programma's. Het ontwerpen en analyseren van herschrijfsystemen is lastiger, enerzijds omdat er geen datatypen en datastructurering zijn: alles moet worden opgebouwd met behulp van herschrijfregels; anderzijds omdat er bij herschrijfsystemen vele verschillende afleidingen van een woord w uit een woord v kunnen zijn, terwijl de door een programma bepaalde berekening op invoer v eenduidig is bepaald. Het huidige voorbeeld is voldoende eenvoudig om zowel de overeenkomsten in benadering als de verschillen in uitwerking overzichtelijk te houden.

Er wordt nu een herschrijfsysteem H = (V,P) gegeven "om de tweetallige schrijfwijze van n te bepalen", of beter geformuleerd "waarvoor geldt $\{0,1,a,\vdash,\dashv\} \subseteq V$ en voor elke $n > 0$ en $w \in \{0,1\}^*$, $\vdash a^n \dashv \overset{*}{\Longrightarrow} w$ dan en slechts dan als $g(w) = n$".

Het herschrijfsysteem H is als volgt gedefinieerd:

- $V = \{0,1,a,\vdash,\dashv\}$
- P bestaat uit regels

$$\left.\begin{array}{l} \vdash \to \vdash C \\ Caa \to aC \end{array}\right\} \text{(d)}$$

$$\left.\begin{array}{l} C\dashv \to \dashv 0 \\ Ca\dashv \to \dashv 1 \end{array}\right\} \text{(r)}$$

$$\vdash\dashv \to \lambda$$

De werking van H blijkt uit de volgende afleiding:

$$\vdash a^5\dashv \Rightarrow \vdash Ca^5\dashv \Rightarrow \vdash aCa^3\dashv \Rightarrow \vdash aaCa\dashv \Rightarrow \vdash aa\dashv 1 \Rightarrow$$
$$\Rightarrow \vdash Ca^2\dashv 1 \Rightarrow \vdash aC\dashv 1 \Rightarrow \vdash a\dashv 01 \Rightarrow \vdash Ca\dashv 01 \Rightarrow$$
$$\Rightarrow \vdash\dashv 101 \Rightarrow 101.$$

De deling wordt dus uitgevoerd met behulp van de regels (d), terwijl de regels (r) het bepalen van de rest bij deling door twee alsmede het opbouwen van de tweetallige schrijfwijze verzorgen.

Om de correctheid van het herschrijfsysteem te bewijzen moet worden aangetoond

(1) Als $\vdash a^n\dashv \overset{*}{\Rightarrow} w$ en $w \in \{0,1\}^*$, dan is $g(w) = n$.

Dit komt overeen met "als het programma termineert dan is $g(w) = n$" in het geval van het Pascal-programma.

(2) Voor elke n is er een woord w over $\{0,1\}$ zodanig dat $\vdash a^n\dashv \overset{*}{\Rightarrow} w$ en $g(w) = n$.

Deze bewering is analoog met "voor elke n termineert de berekening" in het geval van het Pascal-programma.

Deel (1) wordt bewezen met behulp van de invariantenmethode. Bij het opschrijven van de invariant wordt gebruik gemaakt van de functie $m: \{a,C\}^* \to N$ die als volgt is gedefinieerd:

$$m(\lambda) = 0$$
$$m(a^{x_1}Ca^{x_2}C\ldots Ca^{x_k}) = \sum_{i=1}^{k} x_i 2^{k-i} \qquad (x_i > 0,\ 1 \leq i \leq k)$$

Voor elke $n \geq 0$ is de hieronder gedefinieerde U_n een invariant van H:

$$U_n(\phi) = [(\phi \in \{0,1\}^* \text{ en } g(\phi) = n) \text{ of}$$

$$\exists \alpha \in \{a,C\}^* \exists w \in \{0,1\}^* [\phi = \vdash \alpha \dashv w \text{ en } g(w) + m(\alpha)2^{|w|} = n]]$$

Ga na dat U_n, $n \geq 0$, inderdaad een invariant van H is.

Stel nu dat voor zekere $n \in N$ en $w \in \{0,1\}^*$ geldt $\vdash a^n \dashv \overset{*}{\Rightarrow} w$. Omdat U_n een invariant is en U_n ($\vdash a^n \dashv$) geldt, is ook $U_n(w)$ geldig, zodat $g(w) = n$.

Voor deel (2) van het bewijs worden eerst een tweetal hulpeigenschappen afgeleid:

- $\forall n > 0$, $Ca^n \overset{*}{\Rightarrow} a^q C a^r$ met q en r quotiënt en de rest van n bij deling door 2.

  Toon dit zelf aan met inductie naar n.

- $\forall w \in \{0,1\}^*$, $\vdash a^{g(w)} \dashv \overset{*}{\Rightarrow} \vdash \dashv w$.

  Toon dit zelf aan met inductie naar de lengte van w en gebruikmakend van de voorgaande hulpeigenschap.

Het eigenlijke bewijs is met deze hulpeigenschappen eenvoudig: zij $n \in N$, dan is er een woord $w \in \{0,1\}^*$ van minimale lengte zodanig dat $g(w) = n$. Dus $\vdash a^n \dashv = \vdash a^{g(w)} \dashv \overset{*}{\Rightarrow} \vdash \dashv w \Rightarrow w$. •

Bij elk gegeven programma is het mogelijk een herschrijfsysteem te definiëren zodanig dat de berekeningen bepaald door het programma overeenkomen met de afleidingen bepaald door het herschrijfsysteem.

1.31. Oefeningen

(1) Toon aan dat voor elke $w \in \{1v \mid v \in \{0,1\}^*\}$ en elke $m \in N$ geldt $\vdash a^{g(w)} \dashv \overset{*}{\Rightarrow} 0^m w$.

(2) Verander het herschrijfsysteem uit voorbeeld 1.30. zodat er geen leidende nullen worden voortgebracht.

(3) Het herschrijfsysteem uit voorbeeld 1.30. kan als het ware met meer delingen tegelijk bezig zijn (er is meer dan één C in $\vdash \alpha \dashv$). Dit kan worden voorkomen door in plaats van de regels (d) en (r) de volgende regels te nemen:

$\vdash \rightarrow \vdash XC$
$Caa \rightarrow aC$ } (d')

$C\dashv \rightarrow Y\dashv 0$
$Ca\dashv \rightarrow Y\dashv 1$ } (r')

$aY \rightarrow Ya$
$XY \rightarrow \lambda$ } signaal "de deling is klaar"

Toon de correctheid van het aldus veranderde herschrijfsysteem aan.

(4) Schrijf een Pascal programma dat van een gegeven $w \in \{a,b,c\}^*$ (dus w van het type file of (a,b,c)) de tweetallige schrijfwijze bepaalt van $\#(a,w)$ zonder dit aantal voorkomens daadwerkelijk te tellen.

In het vervolg worden verschillende eisen aan de vorm van de produktieregels van een grammatica gesteld en worden de hieruit resulterende klassen van grammatica's beschouwd.

1.32. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica dan heet G van het type 0 als elke herschrijfregel in P van de vorm

$$\alpha \rightarrow \beta$$

is, waarin $\alpha \in V^*(V\backslash\Sigma)V^*$ en $\beta \in V^*$.

Merk op dat aan de vorm van een produktieregel in een type 0 grammatica de eis is opgelegd dat het linkerlid van de regel tenminste één hulpsymbool bevat.

1.33. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica, dan heet G verlengend als voor elke produktieregel $\alpha \rightarrow \beta$ in P geldt dat $|\alpha| \leq |\beta|$ met als mogelijke uitzondering de regel $S \rightarrow \lambda$. Als echter de regel $S \rightarrow \lambda$ in P voorkomt, komt het hulpsymbool S niet in het rechterlid van een produktieregel in P voor.

1.34. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica, dan heet G van het type 1 als elke herschrijfregel in P van de vorm

(i) $\alpha A\beta \rightarrow \alpha\delta\beta$, waarin $A \in V\backslash\Sigma$, $\alpha,\beta \in V^*$ en $\delta \in V^+$, of
(ii) $S \rightarrow \lambda$ is. Als deze regel in P voorkomt, komt S niet in het rechterlid van een produktieregel in P voor.

Een grammatica van het type 1 wordt ook wel een contextgevoelige grammatica genoemd.

Met een regel $\alpha A\beta \rightarrow \alpha\delta\beta$, zoals in de voorgaande definitie is gespecificeerd, wordt aangeduid dat het hulpsymbool A alleen dan mag worden herschreven tot δ als in de te herschrijven zinsvorm $\phi = \gamma A\gamma'$, de linkercontext γ van A eindigt met α en de rechtercontext γ' van A begint met β.

1.35. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica, dan heet G van het type 2 als elke herschrijfregel in P van de vorm

$$A \rightarrow \alpha$$

is, waarin $A \in V\backslash\Sigma$ en $\alpha \in V^*$. Een grammatica van het type 2 wordt ook wel een contextvrije grammatica genoemd.

Met de term "contextvrij" wordt aangegeven dat hulpsymbolen kunnen worden herschreven, onafhankelijk van de context waarin ze voorkomen.

1.36. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica dan heet G lineair als elke herschrijfregel in P van de vorm

(i) $A \rightarrow wBw'$
(ii) $A \rightarrow w$

is, waarin $A,B \in V\backslash\Sigma$ en $w,w' \in \Sigma^*$.

1.37. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica dan heet G rechtslineair als elke produktieregel in P van de vorm

(i) $A \to wB$

(ii) $A \to w$

is, waarin $A,B \in V \setminus \Sigma$ en $w \in \Sigma^*$.

1.38. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica, dan heet G linkslineair als elke herschrijfregel in P van de vorm

(i) $A \to Bw$

(ii) $A \to w$

is, waarin $A,B \in V \setminus \Sigma$ en $w \in \Sigma^*$.

1.39. Definitie: Zij $G = (V,\Sigma,P,S)$ een grammatica dan heet G van het type 3 als elke herschrijfregel in P van de vorm

(i) $A \to aB$

(ii) $A \to \lambda$

is, waarin $A,B \in V \setminus \Sigma$, $a \in \Sigma$. Een grammatica van het type 3 wordt ook wel een reguliere grammatica genoemd.

Naast de in de definities 1.32. tot en met 1.39. ingevoerde klassen van grammatica's komen er in de literatuur nog meer klassen voor. De klassen, die in dit dictaat behandeld worden, zijn echter de meest voorkomende.

1.40. Oefening

Bepaal het type van de volgende grammatica's:

(a) $G_1 = (V,\Sigma,P,S) = (\{S,a,b\}, \{a,b\}, P, S)$ met
$P = \{S \to aSb, S \to ab, S \to ba\}$

(b) $G_2 = (V,\Sigma,P,S) = (\{A,B,C,S,a\}, \{a\}, P, S)$ met
$P = \{S \to BAB,\ B \to \lambda,\ A \to a,\ BA \to BC,\ CA \to AAC,\ CB \to AAB\}$

(c) $G_3 = (V,\Sigma,P,S) = (\{S,a\}, \{a\}, P, S)$ met
$P = \{S \to aS,\ S \to \lambda\}$

Welke talen worden door deze grammatica's voortgebracht?

In definitie 1.41. wordt de klassenindeling van grammatica's doorgetrokken naar een klassenindeling van talen.

1.41. Definitie: Een taal L heet X, X $\in$ {"van het type 0", ..., "van het type 3", "rechts(links) lineair", "lineair", "verlengend"} als er een grammatica G, die X is, bestaat, zodanig dat L = L(G).

De volgende notaties worden gebruikt:
- $\mathcal{L}_0$ duidt de klasse van de type 0 talen aan;
- $\mathcal{L}_{verl}$ duidt de klasse van verlengende talen aan;
- $\mathcal{L}_1$ duidt de klasse van de contextgevoelige talen aan;
- $\mathcal{L}_2$ duidt de klasse van de contextvrije talen aan;
- $\mathcal{L}_{lin}$ duidt de klasse van de lineaire talen aan;
- $\mathcal{L}_{rl}$ duidt de klasse van rechtslineaire talen aan;
- $\mathcal{L}_{ll}$ duidt de klasse van linkslineaire talen aan;
- $\mathcal{L}_3$ duidt de klasse van de reguliere talen aan.

1.42. Voorbeeld

Beschouw de grammatica's
$G_1 = (\{S,A,B,a,b\}, \{a,b\}, P_1, S)$ met
$P_1 = \{S \to AB,\ A \to aBA,\ A \to ab,\ B \to baB,\ B \to ba\}$ en
$G_2 = (\{S,A,B,C,D,a,b\}, \{a,b\}, P_2, S)$ met
$P_2 = \{S \to aB,\ B \to bS,\ B \to bA,\ A \to bC,\ C \to aA,\ C \to aD,\ D \to \lambda\}$.

Uit inspectie van de vorm van de herschrijfregels volgt dat G_1 een contextvrije grammatica is en G_2 een reguliere grammatica. Er geldt $L(G_1)$ =

$\{(ab)^n(ba)^m \mid n,m \geq 1\}$ en $L(G_2) = \{(ab)^n(ba)^m \mid n,m \geq 1\}$ (Ga dit na), zodat $L(G_1) = L(G_2)$ en $L(G_2) \in \mathcal{L}_3$. •

In hoofdstuk 3. zal

$$\mathcal{L}_3 = \mathcal{L}_{rl} = \mathcal{L}_{ll} \subset \mathcal{L}_{lin} \subset \mathcal{L}_2 \subset \mathcal{L}_1 = \mathcal{L}_{verl} \subset \mathcal{L}_0$$

worden aangetoond. In hoofdstuk 2. worden eerst enkele nieuwe begrippen geïntroduceerd.

2. HET BEGRIP ALGORITMISCH OPLOSBAAR

Zij V een eindige verzameling. Met de notatie card(V) (Spreek uit: de cardinaliteit van V) wordt het aantal elementen in V aangeduid, bijvoorbeeld $card(\{\lambda,a,aa,ba\}) = 4$ en $card(\emptyset) = 0$. De cardinaliteiten van eindige verzamelingen kunnen met elkaar vergeleken worden:

$$card(\emptyset) < card(\{\lambda\})$$
$$card(\{0,100,1000\}) = card(\{0,1,2\})$$

Voor oneindige verzamelingen kan de cardinaliteit niet met een natuurlijk getal worden aangegeven. Er moeten daartoe nieuwe notaties worden ingevoerd: voor dit dictaat is alleen de notatie voor de cardinaliteit van N van belang. Deze cardinaliteit wordt aangegeven met het symbool $\aleph_0$ (spreek uit: alef-nul): $card(N) = \aleph_0$. De intuïtieve begrippen aantal, meer, minder en evenveel, zijn erg misleidend als ze worden gebruikt voor de omvang van oneindige verzamelingen:

- N is een oneindige verzameling en $card(N) = \aleph_0$.
- $N^+ = N \setminus \{0\}$ is ook een oneindige verzameling waarvoor $card(N^+) = card(N) = \aleph_0$.
- Ook voor de oneindige verzameling $E = \{2n \mid n \in N\}$ geldt $card(E) = card(N) = \aleph_0$.
- Voor $K = \{n^2 \mid n \in N\}$ geldt $card(K) = card(N) = \aleph_0$.

Intuïtief vindt men echter dat N^+, E en K minder elementen dan N bevatten. Daarom is het nodig het vergelijken van cardinaliteiten zo te definiëren dat geen beroep wordt gedaan op de genoemde intuïtieve begrippen.

2.1. <u>Definitie</u>: Zij V en W verzamelingen, dan

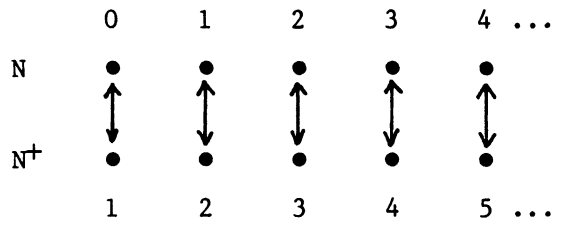
- $card(V) \leq card(W)$ dan en slechts dan als $\exists_\phi$: $V \rightarrowtail W$ (met de notatie $\rightarrowtail$ wordt een injectie aangegeven).
- $card(V) \geq card(W)$ dan en slechts dan als $\exists_\phi$: $V \twoheadrightarrow W$ (met de notatie $\twoheadrightarrow$ wordt een surjectie aangegeven).
- $card(V) = card(W)$ dan en slechts dan als $\exists_\phi$: $V \rightarrowtail\!\!\!\!\twoheadrightarrow W$ (met de notatie $\rightarrowtail\!\!\!\!\twoheadrightarrow$ wordt een bijectie aangegeven).

Twee verzamelingen V en W heten gelijkmachtig als card(V) = card(W). Er kan worden bewezen dat card(V) = card(W) dan en slechts dan als card(V) $\leq$ card(W) en card(W) $\leq$ card(V).

Merk op dat geen definitie van het begrip cardinaliteit is gegeven; een formele behandeling van dit begrip valt buiten het bestek van dit dictaat.

2.2. Voorbeeld

De verzamelingen N en $N^+ = N \setminus \{0\}$ zijn gelijkmachtig: er bestaat een afbeelding ϕ: $N \rightarrowtail\!\!\!\!\!\twoheadrightarrow N^+$,



zodanig dat $\phi = \lambda x[x+1]$, en een inverse afbeelding ϕ^{-1}: $N^+ \rightarrowtail\!\!\!\!\!\twoheadrightarrow N$, zodanig dat $\phi^{-1} = \lambda x[x-1]$. Merk op dat ϕ en ϕ^{-1} bijecties zijn. •

2.3. Definitie: Een verzameling V is eindig dan en slechts dan als er een $n \in N$ bestaat zodanig dat card(V) = card($\{x \in N \mid x < n\}$). V is oneindig als V niet eindig is.

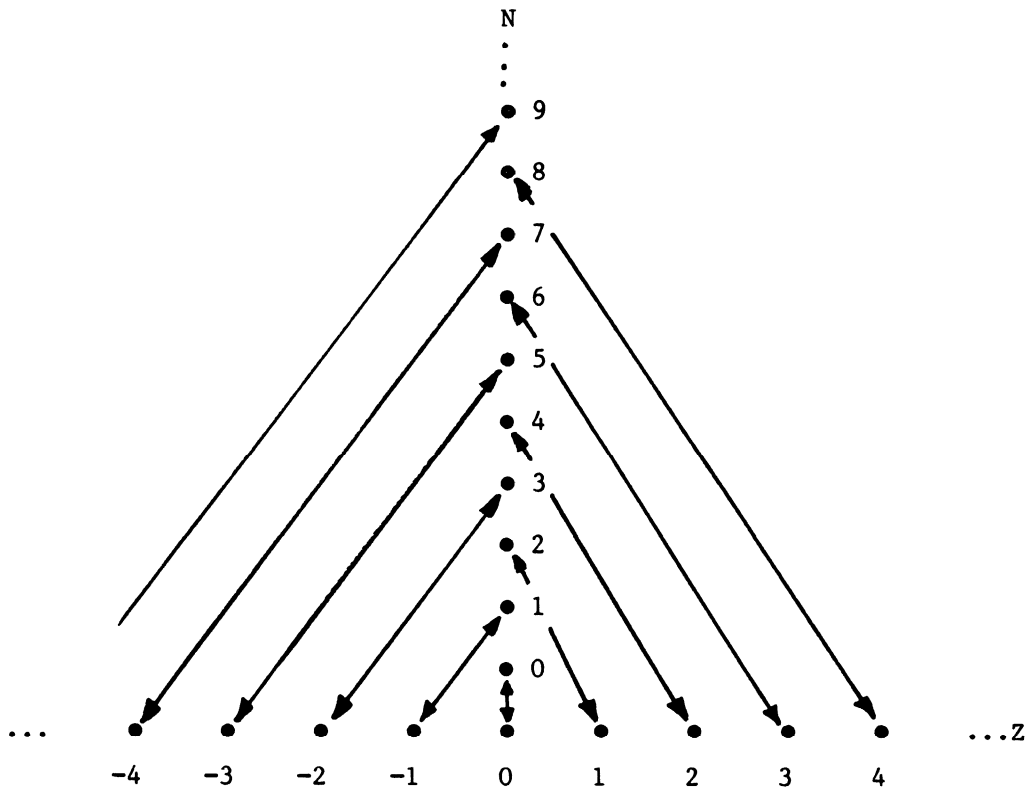
In voorbeeld 2.2. is aangetoond dat N gelijkmachtig is met een echte deelverzameling van zichzelf. Zonder bewijs vermelden we de volgende

2.4. Eigenschap

Een verzameling is oneindig dan en slechts dan als zij gelijkmachtig is met een echte deelverzameling van zichzelf.

2.5. Voorbeeld

Ook Z en N zijn gelijkmachtig. Beschouw de afbeelding ϕ: $Z \rightarrowtail\!\!\!\!\!\rightarrow N$,



zodanig dat

$$\phi = \lambda x[\underline{\text{als}}\ x \geq 0\ \underline{\text{dan}}\ 2x\ \underline{\text{anders}}\ -2x - 1]$$

en de inverse afbeelding ϕ^{-1}: $N \rightarrowtail\!\!\!\!\!\rightarrow Z$, zodanig dat

$$\phi^{-1} = \lambda x[\underline{\text{als}}\ x\ \text{is even}\ \underline{\text{dan}}\ \frac{x}{2}\ \underline{\text{anders}}\ -\frac{x+1}{2}]$$

Uit het feit dat ϕ en ϕ^{-1} bijecties zijn (Ga dit na), volgt dat $\text{card}(N) = \text{card}(Z) = \aleph_0$. ●

2.6. Definitie: Een verzameling V heet aftelbaar oneindig als card(V) = card(N) = $\aleph_0$. Een afbeelding ϕ: N$\twoheadrightarrow$V heet een aftelling of een opsomming van V. Een verzameling V heet aftelbaar (of soms: hoogstens aftelbaar) als V eindig is of als V aftelbaar oneindig is.

2.7. Definitie: Een verzameling V heet overaftelbaar als V niet aftelbaar is. Voor een overaftelbare verzameling V geldt derhalve $\aleph_0$ < card(V).

2.8. Voorbeeld

Zij Σ een willekeurig gekozen alfabet, dan is Σ^* aftelbaar. Laat $\Sigma = \{a_1,\ldots,a_k\}$, $k \geq 1$. Een woord $w = a_{i_1} \ldots a_{i_n} \in \Sigma^*$, $\{i_1,\ldots,i_n\} \subseteq \{1,\ldots,k\}$, wordt met behulp van de volgende afbeelding f: $\Sigma^* \twoheadrightarrow$N als een getal gecodeerd.

$f(\lambda) = 0$
$f(wa) = f(w) \times k + ord(a)$
voor alle $w \in \Sigma^*$ en $a \in \Sigma$,

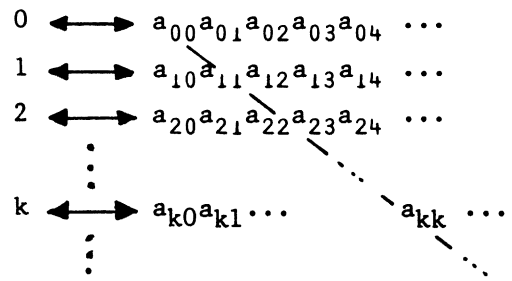
waarin ord(a) het rangnummer van a in de opsomming $a_1,..,a_k$ aanduidt. Ga na dat f een bijectie is. Hieruit volgt dat de afbeelding f^{-1}: N$\twoheadrightarrow\Sigma^*$ een opsomming van Σ^* is, en card(Σ^*) = card(N) = $\aleph_0$. •

2.9. Voorbeeld

Met A^B, waarin A en B verzamelingen zijn, wordt de verzameling van alle functies f: B → A aangeduid. Beschouw de verzameling $\{0,1\}^N$. Elk element uit deze verzameling kan gerepresenteerd worden als een oneindige rij nullen en enen. Aangetoond wordt dat $\{0,1\}^N$ een overaftelbare verzameling is.

(1) duidelijk is dat $\{0,1\}^N$ een oneindige verzameling is.
(2) veronderstel dat er een afbeelding ϕ: N$\twoheadrightarrow\{0,1\}^N$ bestaat, dan

kunnen de representaties van $\phi(0)$, $\phi(1)$, ... als volgt onder elkaar gezet worden.



waarin $\phi(i) = f_i$ met $f_i: N \to \{0,1\}$ en $f_i(j) = a_{ij}$. Definieer een oneindige rij $b_0b_1b_2...$, waarin $b_i \in \{0,1\}$ en b_i is gedefinieerd door

$$b_i = \begin{cases} 0, & a_{ii} = 1 \\ 1, & a_{ii} = 0 \end{cases}$$

Dan is $b = b_0b_1...$ een oneindige rij nullen en enen, en is b de representatie van een element f in $\{0,1\}^N$ met $f(j) = b_j$, dat niet in de opsomming $\phi(0)$, $\phi(1)$, ... voorkomt: b verschilt immers met de representatie $a_{i1}a_{i2}...$ van $\phi(i)$ op tenminste één plaats, (namelijk $f_i(i) \neq a_{ii}$) voor alle $i > 0$. Hieruit volgt dat ϕ geen surjectie en derhalve geen bijectie is.

Uit (1) en (2) volgt dat $\{0,1\}^N$ overaftelbaar is.

De in dit bewijs gehanteerde methode wordt de diagonaalmethode van Cantor genoemd: voor de constructie van een in de opsomming $\phi(0),\phi(1),...$ niet-aanwezige rij nullen en enen worden in de opsomming de elementen op de diagonaalplaatsen beschouwd. ●

2.10. Oefening

Toon aan dat de verzameling van de rationale getallen aftelbaar is, en dat $\{x \mid x \in R \text{ en } 0 \leq x < 1\}$ en R overaftelbaar zijn.

2.11. Definitie: Een algoritme is een eindig handelingsvoorschrift dat werktuiglijk kan worden uitgevoerd.

2.12. Voorbeeld

Voorbeelden van algoritmen in het dagelijks leven zijn recepten en breipatronen. Elk correct Pascal programma is een algoritme. •

In definitie 2.13. wordt gedefinieerd welke functie door een gegeven algoritme wordt berekend.

2.13. Definitie: Zij Q een algoritme met n invoerparameters $x_1,\ldots,x_n$ en m uitvoerparameters $y_1,\ldots,y_m$, $n,m \geq 1$. Zij D_i de verzameling waaruit invoerparameter x_i, $1 \leq i \leq n$, mag worden gekozen en zij C_j de verzameling waaruit uitvoerparameter y_j, $1 \leq j \leq m$, waarden kan aannemen, dan berekent het algoritme Q de partiële functie
$f_Q: D_1 \times D_2 \times \ldots \times D_n \rightarrow C_1 \times C_2 \times \ldots \times C_m$, die als volgt gedefinieerd is: geef invoerparameter x_i de waarde $p_i \in D_i$, $1 \leq i \leq n$. Laat vervolgens algoritme Q op $p_1,\ldots,p_n$ werken. Als de berekening op $p_1,\ldots,p_n$ eindigt, notatie: $Q(p_1,\ldots,p_n)\downarrow$, dan wordt de functiewaarde $f_Q(p_1,\ldots,p_n)$ gegeven door de waarde van $(y_1,\ldots,y_m)$. Eindigt de berekening op invoer $p_1,\ldots,p_n$ niet, notatie: $Q(p_1,\ldots,p_n)\uparrow$, dan is $f_Q(p_1,\ldots,p_n)$ ongedefinieerd. Dit wordt in navolging van de notatie voor algoritmen genoteerd als $f_Q(p_1,\ldots,p_n)\uparrow$. Als $f_Q(p_1,\ldots,p_n)$ wel gedefinieerd is, wordt dit genoteerd als $f_Q(p_1,\ldots,p_n)\downarrow$.

2.14. Definitie: Een functie f die door een algoritme kan worden berekend, heet berekenbaar.

Merk op dat algoritmen in het algemeen partiële functies berekenen; alleen algoritmen die voor alle toegestane invoerwaarden termineren, leveren totale functies op.

2.15. Voorbeeld

Beschouw de functie f: $N \times N^+ \to N \times N$, waarin $f(x,y) = (q,r)$, zodanig dat $x = q * y + r$, $0 \leq r < y$. Deze functie f is berekenbaar, immers $f = f_A$ voor het volgende algoritme A:

invoer : $x \in N$, $y \in N^+$
uitvoer: $q,r \in N$, zodanig dat $x = q * y + r$, $0 \leq r < y$

```
begin
  {x ≥ 0, y > 0}
  q := 0;
  r := x;
  {x = q * y + r, 0 ≤ r}
  while r ≥ y do
  begin
    {x = q * y + r, 0 ≤ y ≤ r}
    r := r - y;
    {x = (q+1) * y + r, 0 ≤ r}
    q := q + 1
    {x = q * y + r, 0 ≤ r}
  end
  {x = q * y + r, 0 ≤ r < y}
end.
```

Merk op dat f_A een totale functie is. Als in de invoerspecificatie $y \in N$ wordt genomen in plaats van $y \in N^+$, dan is f_A niet meer totaal maar partiëel. ●

2.16. Aanname Iedere algoritme kan worden geformuleerd als een Pascal-programma.

In dit dictaat wordt in plaats van Pascal een Pascal-"dialect" gehanteerd. Het verschil tussen beide talen betreft slechts notationele kwesties, zodat aangenomen kan worden dat elk algoritme dat in Pascal kan worden geformuleerd ook in het Pascal-dialect kan worden geformuleerd en omgekeerd.

2.17. Definitie: Een probleem is een open bewering $U(x_1,\ldots,x_n)$ samen met een specificatie van domeinen $D_1,\ldots,D_n$, waaruit de invoerparameters $x_1,\ldots,x_n$ waarden mogen aannemen. Een probleem wordt genoteerd als $[x_1 \in D_1,\ldots,x_n \in D_n\colon\ U(x_1,\ldots,x_n)]$. Door voor elke invoerparameter x_i een waarde $p_i \in D_i$, $1 \leq i \leq n$, te kiezen, wordt een uitkomst $U(p_1,\ldots,p_n) \in$ {waar, onwaar} verkregen. Een probleem samen met een waarde voor de invoerparameters wordt een exemplaar (Eng.: "instance") van dat probleem genoemd.

Een probleem kan beschouwd worden als een predikaat op een bepaald domein. In dit dictaat wordt echter de voorkeur gegeven aan de formulering uit definitie 2.17. om zoveel mogelijk aan te sluiten bij het begrip algoritme (zie definitie 2.19.).

2.18. Voorbeelden

(1) Zij $U = [a \in Z,\ b \in Z,\ c \in Z\colon \exists x \in Z[ax^2 + bx + c = 0]]$. Een exemplaar van U is $U(5,7,1) = [\exists x \in Z[5x^2 + 7x + 1 = 0]]$ = onwaar.

(2) Zij L_1^1 de verzameling van alle syntactisch correcte Pascal-programma's met één invoerparameter en één uitvoerparameter, die beide waarden uit N mogen aannemen. $U = [1 \in L_1^1,\ x \in N\colon\ 1(x)\downarrow]$ is een probleem.

(3) $U = [1_1,1_2 \in L_1^1\colon\ 1_1$ en 1_2 zijn equivalent$]$ is een probleem. •

2.19. Definitie: Een probleem $U = [x_1 \in D_1,\ \ldots,\ x_n \in D_n\colon\ U(x_1,\ldots,x_n)]$ heet algoritmisch oplosbaar (Eng.: "algorithmically solvable") als er een algoritme Q bestaat met n invoerparameters $x_1,\ldots,x_n (x_i \in D_i,\ 1 \leq i \leq n)$ en één uitvoerparameter, die een waarde uit {waar, onwaar} aan kan nemen, zodanig dat $f_Q(a_1,\ldots,a_n) = U(a_1,\ldots,a_n)$ voor alle $(a_1,\ldots,a_n)$ in $D_1 \times \ldots \times D_n$. Dit houdt tevens in dat Q voor alle $(a_1,\ldots,a_n)$ in $D_1 \times \ldots \times D_n$ termineert en dus dat f_Q totaal is.

Een probleem $U = [x_1 \in D_1,\ \ldots,\ x_n \in D_n\colon\ U(x_1,\ldots,x_n)]$ heet algoritmisch partieel oplosbaar (Eng.: "algorithmically

partially solvable") als er een algoritme Q bestaat met n invoerparameters $x_1,\ldots,x_n$ met $x_i \in D_i$, $1 \leq i \leq n$, en één uitvoerparameter die waarden uit {waar, onwaar} kan aannemen, zodanig dat $Q(a_1,\ldots,a_n)\downarrow$ dan en slechts dan als $a_i \in D_i$, $1 \leq i \leq n$, en $U(a_1,\ldots,a_n)$ = waar, en in dat geval geldt $f_Q(a_1,\ldots,a_n) = U(a_1,\ldots,a_n)$ = waar.

Een probleem $U = [x_1 \in D_1, \ldots, x_n \in D_n: U(x_1,\ldots,x_n)]$ heet algoritmisch onoplosbaar (Eng.: "unsolvable") als U niet algoritmisch oplosbaar is.

Er zijn problemen die algoritmisch onoplosbaar zijn, maar wel algoritmisch partieel oplosbaar. Voorbeelden hiervan zullen in het navolgende behandeld worden.

2.20. In het vervolg zal het wenselijk blijken te zijn Pascal-programma's als getallen te coderen. Hiertoe worden een aantal functies ingevoerd. Pascal-programma's worden, gebruikmakend van een eindig aantal k van basissymbolen $a_1,\ldots,a_k$ ($k \geq 1$), opgeschreven. Voor de codering van Pascalteksten als getallen wordt in het vervolg gebruik gemaakt van de in voorbeeld 2.8. beschouwde afbeelding $f: \{a_1,\ldots,a_k\}^* \hookrightarrow \mathbb{N}$.

Verder wordt er aangenomen dat er een berekenbare functie $g: \{a_1,\ldots,a_k\}^* \longrightarrow$ {waar, onwaar} bestaat, die bepaalt of een aangeboden tekst $w \in \{a_1,\ldots,a_k\}^*$ een syntactisch correct Pascal-programma is. Laat nu $L \subseteq \{a_1,\ldots,a_k\}^*$ de verzameling syntactisch correcte Pascal-programma's zijn, dan bestaat er een berekenbare functie $h: \mathbb{N} \to L$, zodanig dat

$$h = \lambda x[\underline{\text{als}}\ g(f^{-1}(x)) = \text{waar}\ \underline{\text{dan}}\ f^{-1}(x)\ \underline{\text{anders}}\ A]$$

waarin A een vast maar van tevoren willekeurig gekozen Pascal-programma is, bijvoorbeeld

```
var x;
begin
  x := 1
end.
```

2.21. Voorbeeld

Het probleem uit voorbeeld 2.18.(1) is algoritmisch oplosbaar. Het probleem uit voorbeeld 2.18.(2) is algoritmisch onoplosbaar. De algoritmische onoplosbaarheid van dit probleem, het stopprobleem genaamd, wordt uit het ongerijmde bewezen (een bewijs uit het ongerijmde toont de onaannemelijkheid aan van het tegengestelde van hetgeen te bewijzen is).

Stel dat er een algoritme P bestaat met twee invoerparameters $x,y \in N$, en één uitvoerparameter die een waarde uit {waar, onwaar} kan aannemen, die de volgende functie berekent:

$$f_P = \lambda xy[\underline{\text{als}}\ f_{h(x)}(y)\downarrow\ \underline{\text{dan}}\ \text{waar}\ \underline{\text{anders}}\ \text{onwaar}]$$

Zij H een algoritme met één invoerparameter $x \in N$, en één uitvoerparameter die de waarde waar kan aannemen, die de volgende functie berekent:

$$f_H = \lambda x[\underline{\text{als}}\ f_P(x,x) = \text{onwaar}\ \underline{\text{dan}}\ \text{waar}\ \underline{\text{anders}}\ \text{ongedefinieerd}]$$

Als algoritme P bestaat, dan bestaat algoritme H ook. Maar dat leidt tot een tegenspraak:

Zij x_0 een index van algoritme H, dat wil zeggen $h(x_0) = H$.

(1) als $f_H(x_0)\downarrow$, dan is $f_P(x_0,x_0)$ = onwaar, maar dan geldt niet $f_H(x_0)\downarrow$, maar $f_H(x_0)\uparrow$;

(2) als $f_H(x_0)\uparrow$, dan is $f_P(x_0,x_0)$ = waar, maar dan geldt niet $f_H(x_0)\uparrow$, maar $f_H(x_0)\downarrow$.

Uit (1) volgt $f_H(x_0)\uparrow$ en uit (2) volgt $f_H(x_0)\downarrow$; (1) en (2) vormen derhalve samen een tegenspraak. Hieruit volgt dat de aanname dat er een algoritme P bestaat met de werking

$$f_P = \lambda xy[\underline{\text{als}}\ f_{h(x)}(y)\downarrow\ \underline{\text{dan}}\ \text{waar}\ \underline{\text{anders}}\ \text{onwaar}]$$

onjuist is. Het stopprobleem is dus algoritmisch onoplosbaar. •

In voorbeeld 2.21. hebben we de algoritmische onoplosbaarheid van een probleem uit het ongerijmde aangetoond. Een tweede manier om de algorit-

mische onoplosbaarheid van een probleem aan te tonen, is dit probleem te reduceren tot een ander probleem waarvan al eerder is aangetoond dat het algoritmisch onoplosbaar is. Een dergelijke reductie vindt plaats door het aangeven van een effectieve methode om een gegeven willekeurig exemplaar van het algoritmisch onoplosbare probleem om te zetten naar een exemplaar van het probleem in kwestie. Als het laatstgenoemde probleem algoritmisch oplosbaar zou zijn, dan zou daarmee ook het algoritmisch onoplosbare probleem algoritmisch oplosbaar zijn. Aangezien dit niet het geval is, wordt vervolgens geconcludeerd dat het probleem in kwestie algoritmisch onoplosbaar is. In de volgende paragrafen zal veelvuldig van deze reductiemethode gebruik gemaakt worden.

In de theorie van de formele talen is het prettig te beschikken over een algoritmisch onoplosbaar probleem, dat betrekking heeft op symboolrijen in plaats van op algoritmen. Een dergelijk probleem is het correspondentieprobleem van Post (Eng.: "Post Correspondence Problem", "PCP").

2.22. Definitie: Het correspondentieprobleem van Post over het alfabet V is het probleem PCP = $[n \in N^+,\ a_1,\ldots,a_n,b_1,\ldots,b_n \in V^+:$ $\exists k \in N,\ k > 1,\ \exists\ i_1,\ldots,i_k,$ met $1 \leq i_j \leq n$ voor $1 \leq j \leq k$ zodanig dat $[a_{i_1}\ldots a_{i_k} = b_{i_1}\ldots b_{i_k}]]$.

2.23. Voorbeelden

(1) Beschouw het exemplaar van het PCP, waarin V = {a,b}, n = 3, (a_1,a_2,a_3) = (a,bba,aab) en (b_1,b_2,b_3) = (ba,aaa,ba).

Dit exemplaar levert de uitspraak onwaar op, aangezien elk rijtje indices dat aan de gestelde eisen voldoet, met een zodanig index i, $1 \leq i \leq 3$, moet beginnen, dat a_i en b_i met eenzelfde niet-leeg deelwoord beginnen. Ga na dat voor dit exemplaar een dergelijke index niet bestaat.

(2) Beschouw het exemplaar van het PCP, waarin V = {a,b}, n = 2, (a_1,a_2) = (bbb,abb) en (b_1,b_2) = (bb,babbb).

Dit exemplaar levert de uitspraak waar op, want voor het rijtje indices 121 geldt immers

bbb.abb.bbb = bb.babbb.bb

Merk op dat ook het rijtje 121121 aan de gestelde eisen voldoet. In het algemeen geldt dat als er voor een exemplaar van het PCP zo'n rijtje indices bestaat, er meer en wel oneindig veel van die rijtjes bestaan. Zij I de verzameling indices van een PCP, en $L \subseteq I^*$ de verzameling oplossingen. Dan geldt: als $x,y \in L$ dan ook $xy \in L$. Met andere woorden $L^+ = L$, waarbij L^+ gedefinieerd is als $L^+ = \{x_1..x_n \mid n \geq 1;\ x_i \in L,\ 1 \leq i \leq n\}$.

(3) Beschouw het exemplaar van het PCP, waarin $V = \{a,b\}$, $n = 3$, $(a_1,a_2,a_3) = (ba,abb,bab)$ en $(b_1,b_2,b_3) = (bab,bb,abb)$.

Uit inspectie van (a_1,a_2,a_3) en (b_1,b_2,b_3) volgt dat elk rijtje indices dat aan de gestelde eisen voldoet, met de index $i_1 = 1$ moet beginnen:

a_1 = ba
b_1 = bab

De index i_2 moet nu zo gekozen worden dat a_{i_2} met de letter b begint: $i_2 = 1$ of $i_2 = 3$. Als $i_2 = 1$ wordt gekozen, dan

a_1a_1 = baba
b_1b_1 = babbab

hetgeen niet tot een rijtje indices leidt dat aan de gestelde eisen voldoet, zodat $i_2 = 3$ moet worden gekozen:

a_1a_3 = babab
b_1b_3 = bababb

Uit eenzelfde redering volgt $i_3 = 3$, $i_4 = 3$, ... Elk eindig rijtje 1333...3 levert echter woorden van ongelijke lengte op, zodat het beschouwde exemplaar van het PCP de uitspraak onwaar oplevert. ●

2.24. Eigenschap

Het correspondentieprobleem van Post is algoritmisch onoplosbaar.

Binnen het kader van dit dictaat zou het te ver voeren het bewijs van de algoritmische onoplosbaarheid van het correspondentieprobleem van Post te behandelen. In de literatuur is een reductie van het PCP naar het stopprobleem te vinden.

Merk op dat het feit dat het voor sommige exemplaren van het PCP mogelijk is te bepalen of deze exemplaren de uitspraak waar dan wel onwaar opleveren, niet in tegenspraak is met eigenschap 2.24.: het begrip algoritmische onoplosbaarheid heeft immers betrekking op het bestaan van een werktuigelijke methode, die voor alle exemplaren het juiste antwoord levert.

2.25. Oefeningen

Toon aan dat het PCP algoritmisch oplosbaar is als voor het alfabet V geldt card(V) = 1.

2.26. Definitie: Zij V een alfabet en $L \subseteq V^*$. De verzameling L heet

1. beslisbaar (Eng.: "decidable") als de karakteristieke functie χ_L van L, gedefinieerd door

$$\chi_L(x) = \begin{cases} 1, & \text{dan en slechts dan als } x \in L \\ 0, & \text{dan en slechts dan als } x \notin L \end{cases}$$

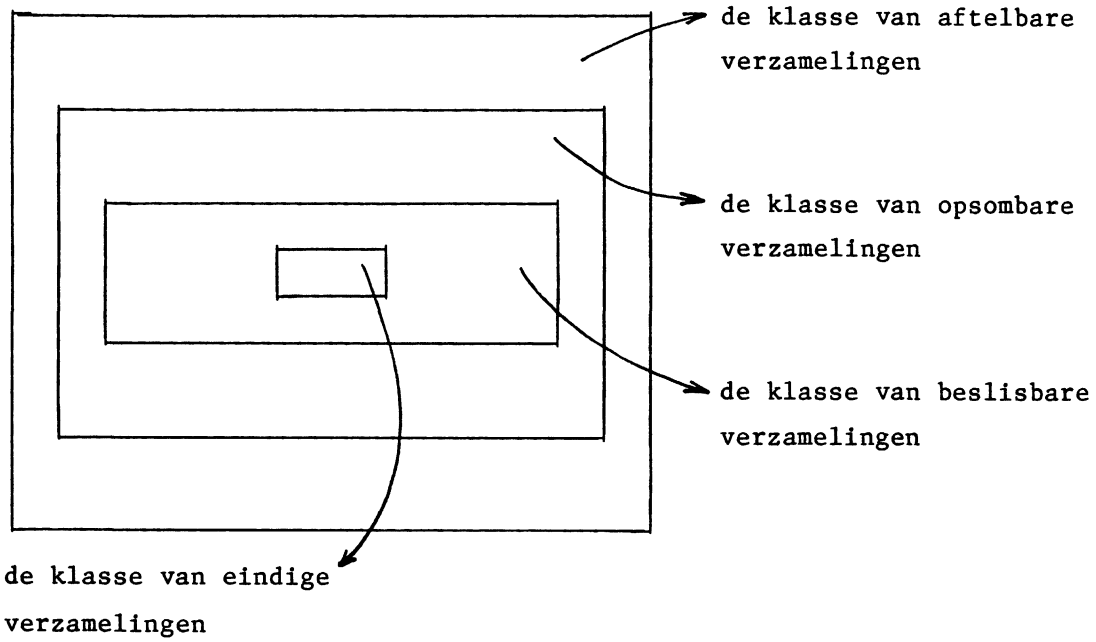
berekenbaar is.

2. opsombaar (Eng.: "recursively enumerable", "semi-decidable") als er een berekenbare (partiële) functie bestaat, waarvan het definitiegebied samenvalt met L.

2.27. Oefening

Zij V een alfabet en zij $L \subseteq V^*$. Toon aan: als L beslisbaar is, dan is $V^* \setminus L$ beslisbaar. Toon vervolgens aan dat L beslisbaar is dan en slechts dan als zowel L als $V^* \setminus L$ opsombaar zijn.

De relaties tussen de behandelde begrippen aftelbaar, opsombaar en beslisbaar komen in het volgende Venn-diagram tot uitdrukking:



2.28. Eigenschap

Elke eindige verzameling is beslisbaar.

Bewijs
De karakteristieke functie van een eindige verzameling kan worden berekend door een algoritme bestaand uit een eindige reeks if-statements.
Einde bewijs

In eigenschap 2.28. is aangetoond dat de klasse van eindige verzamelingen een deelverzameling is van de klasse van beslisbare verzamelingen. In voorbeeld 2.29 wordt aangetoond dat de klasse van eindige verzamelingen een echte deelverzameling is van de klasse van beslisbare verzamelingen.

2.29. Voorbeeld

De verzameling $\{w \mid w \in \{a,b\}^* \text{ en } |w| \text{ is een 13-voud}\}$ is een beslisbare oneindige verzameling. Ook de verzameling van alle syntactische correcte Pascal-programma's is een oneindige beslisbare verzameling. Van elk Pascal-programma kan immers met behulp van een Pascal-compiler vastgesteld worden of het al dan niet syntactisch correct is. •

2.30. Eigenschap

Als een verzameling L beslisbaar is, dan is L ook opsombaar.

Bewijs
De verzameling L is beslisbaar, dat wil zeggen dat de karakteristieke functie χ_L van L berekenbaar is. Het volgende algoritme stopt precies voor die invoerwaarden x waarvoor geldt dat $x \in L$.

```
invoer : x
variabele: i

begin
  i := χL(x);
  while i = 0 do i := 0
end.
```

Einde bewijs

2.31. Eigenschap

Er bestaat een verzameling die wel opsombaar maar niet beslisbaar is.

Bewijs
Zij $U = [x_1 \in D_1, \ldots, x_n \in D_n, n \geq 1: U(x_1,\ldots,x_n)]$ een probleem. Zij $V(U) = \{(a_1,\ldots,a_n) \mid U(a_1,\ldots,a_n) = \text{waar}\}$. Het is duidelijk dat V(U) beslisbaar is dan en slechts dan als U algoritmisch oplosbaar is. Als voor U een algoritmisch onoplosbaar probleem wordt gekozen, is V(U) dus een onbeslisbare verzameling.

Kies als algoritmisch onoplosbaar probleem het stopprobleem zoals dat in voorbeeld 2.21. werd geformuleerd: STOP = $[x,y \in N: f_{h(x)}(y)\downarrow]$. Laat V(STOP) gedefinieerd zijn door V(STOP) = $\{(x,y) \mid U(x,y) = \text{waar}\}$. Uit de voorgaande beschouwing volgt dat V(STOP) een onbeslisbare verzameling is. Beschouw nu het volgende algoritme A:

invoer : $x,y \in N$
uitvoer: $z \in N$

begin
  $z := f_{h(x)}(y)$
end.

Algoritme A berekent de volgende functie

$f_A = \lambda xy[\underline{\text{als}}\ f_{h(x)}(y)\downarrow\ \underline{\text{dan}}\ f_{h(x)}(y)\ \underline{\text{anders}}\ \text{ongedefinieerd}]$

waarvan het definitiegebied samenvalt met V(STOP). Hieruit volgt dat V(STOP) opsombaar is.
Einde bewijs

2.32. Oefeningen

(1) Toon aan dat elke opsombare verzameling aftelbaar is.
(2) Bedenk een aftelbare verzameling die niet opsombaar is.

3. EIGENSCHAPPEN VAN EN RELATIES TUSSEN KLASSEN VAN TALEN

In hoofdstuk 1 is een taal L over een alfabet Σ gedefinieerd als een deelverzameling van Σ^*. Verder is opgemerkt dat grammatica's talen specificeren. In de eigenschappen 3.1. en 3.2. wordt aangetoond dat er niet voor elke deelverzameling L van Σ^* een grammatica G bestaat, zodanig dat L(G) = L.

3.1. Eigenschap

Zij Σ een alfabet. De verzameling van alle deelverzamelingen van Σ^* is overaftelbaar.

Bewijs

In voorbeeld 2.8. is aangetoond dat er een bijectie ϕ: $\Sigma^* \rightarrowtail\!\!\!\!\!\rightarrow N$ bestaat; dus ϕ: $P(\Sigma^*) \rightarrowtail\!\!\!\!\!\rightarrow P(N)$ met $\psi = \lambda x[\{\phi(y) \mid y \in x\}]$ is een bijectie, zodat $card(P(\Sigma^*)) = card(P(N))$. Merk op dat $card(P(N)) = card(\{0,1\}^N)$. In voorbeeld 2.10. is aangetoond dat $card(\{0,1\}^N) > \aleph_0$. Uit $card(P(\Sigma^*)) = card(P(N))$ en $card(P(N)) > \aleph_0$ volgt $card(P(\Sigma^*)) > \aleph_0$.

Einde bewijs

Zij Σ een alfabet en $\mathcal{L}_\Sigma = \{L \mid L \in \mathcal{L}_0 \text{ en } L \subseteq \Sigma^*\}$, dan is $\mathcal{L}_\Sigma$ de verzameling van alle deelverzamelingen van Σ^*, die door een grammatica kunnen worden voortgebracht.

3.2. Eigenschap

De klasse $\mathcal{L}_\Sigma$ is aftelbaar oneindig.

Bewijs

Het bewijs verloopt in twee stappen:

(1) Duidelijk is dat er minstens aftelbaar oneindig veel deelverzamelingen van Σ^* zijn, die door een grammatica kunnen worden voortgebracht. Elke deelverzameling van Σ^* die precies één element bevat, kan immers door een grammatica worden voortgebracht.

(2) Aangetoond wordt dat er een surjectie $N \twoheadrightarrow \mathcal{L}_\Sigma$ bestaat, zodat $card(\mathcal{L}_\Sigma) \leq \aleph_0$. Dit gebeurt via het coderen van een grammatica G met eindalfabet Σ als een woord over het alfabet $W = \Sigma \cup \{\#, @, \$, \rightarrow\}$ waarin $\#, @, \$, \rightarrow \notin \Sigma$. Op willekeurige wijze worden de hulpsymbolen van G genummerd van 1 tot en met k, met dien verstande dat het startsymbool S het nummer 1 krijgt. Een hulpsymbool A dat het nummer i heeft gekregen, $1 \leq i \leq k$, wordt gecodeerd als $\#\$^i\#$. Een produktieregel wordt gecodeerd door er een @ voor en een @ achter te schrijven, de hulpsymbolen door hun respectievelijke code te vervangen en de regel verder ongewijzigd over te nemen. Alle gecodeerde produktieregels worden vervolgens achter elkaar geschreven.

Volgens deze codering horen bij elke grammatica verscheidene codewoorden, en kunnen bij elk woord $w \in W^*$ nul of meer grammatica's worden gevonden, die op herbenoemen van de hulpsymbolen na, gelijk zijn.

Zij $f: W^* \rightarrowtail N$ een bijectie (zie voorbeeld 2.8.) en $h: N \twoheadrightarrow \mathcal{L}_\Sigma$ gedefinieerd aldus:

$h = \lambda x[$<u>als</u> $f^{-1}(x)$ een codewoord van grammatica G is <u>dan</u> L(G) <u>anders</u> $\emptyset]$.

Als $L \in \mathcal{L}_\Sigma$, dan is er een grammatica G met eindalfabet Σ zodanig dat L(G) = L, een codewoord c(G) voor G en een $n \in N$ zodanig dat $f^{-1}(n) = c(G)$ en dus h(n) = L. Dus is h een surjectie.

Uit (1) en (2) samen volgt $\aleph_0 < card(\mathcal{L}_\Sigma) \leq \aleph_0$, zodat $\mathcal{L}_\Sigma$ aftelbaar oneindig is.
<u>Einde bewijs</u>

In hoofdstuk 1 zijn verschillende klassen van talen die voortgebracht kunnen worden door grammatica's, geïntroduceerd. Tussen de klassen $\mathcal{L}_0$, $\mathcal{L}_1$, $\mathcal{L}_2$ en $\mathcal{L}_3$ bestaat het volgende verband, dat de hierarchie van Chomsky wordt genoemd:

3.3. Eigenschap

$\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$.

De rest van dit hoofdstuk zal vrijwel geheel gewijd worden aan het bewijs van deze eigenschap.

3.4. Eigenschap

$\mathcal{L}_3 \subseteq \mathcal{L}_2$

Bewijs
Deze eigenschap volgt direct uit de definities 1.35. en 1.39.
Einde bewijs

Aangetoond wordt dat de klasse $\mathcal{L}_{r1}$ van alle talen die voortgebracht kunnen worden door rechtslineaire grammatica's, gelijk is aan $\mathcal{L}_3$.

3.5. Eigenschap

$\mathcal{L}_3 \subseteq \mathcal{L}_{r1}$

Bewijs
Deze eigenschap volgt direct uit de definities 1.37. en 1.39.
Einde bewijs

3.6. Eigenschap

$\mathcal{L}_{r1} \subseteq \mathcal{L}_3$

Bewijs
Zij G = (V,Σ,P,S) een rechtslineaire grammatica. De constructie van een equivalente type 3 grammatica G" bestaat uit drie deelconstructies.

(1) Construeer uit G een met G equivalente rechtslineaire grammatica G' = (V,Σ,P',S), zodanig dat in P' geen produktieregels van de vorm

$A \to B$, $A,B \in V\backslash\Sigma$, voorkomen. Zij $N(A) = \{B \mid B \in V\backslash\Sigma \text{ en } A \overset{*}{\Rightarrow} B\}$. De verzameling N(A) kan daadwerkelijk worden geconstrueerd (zie 3.7.). De verzameling P' van nieuwe produktieregels wordt uit de verzameling P van produktieregels verkregen door alle produktieregels van de vorm $A \to B$ te verwijderen en voor alle mogelijke afleidingen $A \overset{*}{\Rightarrow} B \Rightarrow \alpha$, $B \in N(A)$, $\alpha \notin V\backslash\Sigma$, een produktieregel $A \to \alpha$ in P' op te nemen:

$$P' = \{A \to \alpha \mid \alpha \notin V\backslash\Sigma \text{ en } \exists B \in N(A)[B \to \alpha \in P]\}.$$

Dan geldt $X \overset{*}{\underset{G'}{\Rightarrow}} Y$ dan en slechts dan als $X \overset{*}{\underset{G}{\Rightarrow}} Y$. Ga dit na.

(2) Voer voor elke produktieregel van de vorm $A \to a_1 \ldots a_n B$, $A,B \in V\backslash\Sigma$ en $a_1,\ldots,a_n \in \Sigma$, $n \geq 2$, in P', n-1 nieuwe hulpsymbolen $A_1,\ldots,A_{n-1}$ in en vervang de regel $A \to a_1 \ldots a_n B$ door het stelsel regels

$$\begin{aligned} A &\to a_1 A_1 \\ A_1 &\to a_2 A_2 \\ &\vdots \\ A_{n-1} &\to a_n B \end{aligned}$$

(3) Voer voor elke produktieregel van de vorm $A \to a_1 a_2 \ldots a_n$, $A \in V\backslash\Sigma$, $a_1,\ldots,a_n \in \Sigma$, $n \geq 1$, in P', n nieuwe hulpsymbolen $A'_1,\ldots,A'_n$ in en vervang de regel $A \to a_1 \ldots a_n$ door het stelsel regels

$$\begin{aligned} A &\to a_1 A'_1 \\ &\vdots \\ A'_{n-1} &\to a_n A'_n \\ A'_n &\to \lambda \end{aligned}$$

Noem de verzameling produktieregels die uit de fasen (2) en (3) resulteert, P". Dan geldt $\alpha \overset{*}{\underset{P'}{\Rightarrow}} w$ dan en slechts dan als $\alpha \overset{*}{\underset{P''}{\Rightarrow}} w$ dan en slechts dan als $\alpha \overset{*}{\underset{P''}{\Rightarrow}} w$. Met behulp van deze eigenschap kan men op eenvoudige wijze aantonen dat L(G") = L(G') = L(G). (Toon dit zelf aan.)

Einde bewijs

3.7. In het bewijs van eigenschap 3.6. is voor de grammatica $G = (V,\Sigma,P,S)$ voor elk hulpsymbool $A \in V\setminus\Sigma$ de verzameling $N(A) = \{B \mid B \in V\setminus\Sigma \text{ en } A \overset{*}{\Longrightarrow} B\}$ gedefinieerd. Het volgende algoritmeschema bepaalt voor elk hulpsymbool A de verzameling N(A); in het algoritme wordt een rij verzamelingen $N_0(A)$, $N_1(A)$, ..., $N_{i_0}(A)$ voor alle hulpsymbolen $A \in V\setminus\Sigma$ aldus bepaald:

$$N_0(A) = \{A\}$$
$$N_{i+1}(A) = N_i(A) \cup \{B \mid B \in V\setminus\Sigma \text{ en } \exists X \in N_i(A)[X \to B \in P]\}$$

Zij H de verzameling hulpsymbolen, dat wil zeggen $H = V\setminus\Sigma$.

invoer : de verzameling productieregels P, een hulpsymbool X en de verzameling hulpsymbolen H.
uitvoer: de verzameling hulpsymbolen N, zodanig dat N = N(X).

```
var Y,Z: char; U,W: set of char;
begin
   U := ∅;
   W := {X};
   while U ≠ W do
   begin
      U := W;
      for all Z ∈ U do
      begin
         for all Y ∈ H do
             if Z → Y ∈ P then W := W ∪ {Y}
      end
   end;
   N := W
end.
```

Uit bovenstaand programma worden enkele kenmerken van het in dit dictaat gebruikte Pascal-dialect duidelijk, zoals bijvoorbeeld de for all x ∈ V constructie die één voor één de elementen x van een verzameling V afloopt. Tevens wordt er vanuit gegaan dat een variabele van het type set of T een willekeurig aantal elementen van het type T kan bevatten (In werkelijkheid treft men altijd een door de implementatie bepaalde bovengrens aan het aantal elementen aan.).

Dit algoritme termineert voor elke grammatica G; immers $N_0 \subseteq \ldots \subseteq N_i(A) \subseteq \ldots \subseteq H$ is een monotoon niet-dalende rij verzamelingen begrensd door een eindige verzameling, zodat

$$\exists i \in N \, [N_i(A) = N_{i+1}(A)].$$

Hieruit volgt dat

$$\exists i_0 \in N \, \forall A \in H \, [N_{i_0}(A) = N_{i_0+1}(A)].$$

Merk op dat $i_0 \leq \text{card}(H) = \text{card}(V \setminus \Sigma)$. Voor i_0 geldt verder dat

$$N_{i_0}(A) = N(A)$$

Met behulp van inductie naar de (verzamelings-) index i, $i \geq 1$, wordt aangetoond dat $N_i(A) = \{B \mid B \in V \setminus \Sigma \text{ en } \exists m \leq i \, [A \overset{m}{\Rightarrow} B]\}$.

Initialisatiestap: $i := 1$. $N_i(A) = N_0(A) \cup \{B \mid B \in V \setminus \Sigma \text{ en } A \to B \in P\} =$
$= \{B \mid B \in V \setminus \Sigma \text{ en } (A \overset{0}{\Rightarrow} B \text{ of } A \overset{1}{\Rightarrow} B)\} =$
$= \{B \mid B \in V \setminus \Sigma \text{ en } \exists m \leq 1 \, [A \overset{m}{\Rightarrow} B]\}$.

Inductieveronderstelling: Neem aan dat $\forall i \leq k$, $k \geq 1$, geldt
$N_i(A) = \{B \mid B \in V \setminus \Sigma \text{ en } \exists m \leq i \, [A \overset{m}{\Rightarrow} B]\}$,

Inductiestap: $N_{k+1} = N_k(A) \cup \{C \mid C \in V \setminus \Sigma \text{ en } \exists B \in N_k(A) [B \to C \in P]\}$
$= \{B \mid B \in V \setminus \Sigma \text{ en } \exists m \leq k \, [A \overset{m}{\Rightarrow} B]\} \cup$
$\{C \mid C \in V \setminus \Sigma \text{ en } \exists B \in \{B \mid B \in V \setminus \Sigma \text{ en } \exists m \leq k \, [A \overset{m}{\Rightarrow} B]\}$
$[B \to C \in P]\} =$
$= \{C \mid C \in V \setminus \Sigma \text{ en } \exists m \leq k \, [A \overset{m}{\Rightarrow} B \Rightarrow C]\} =$
$= \{B \mid B \in V \setminus \Sigma \text{ en } \exists m \leq k+1 \, [A \overset{m}{\Rightarrow} B]\}$.

Hieruit volgt dat $N_{i_0}(A) = \{B \mid B \in V \setminus \Sigma \text{ en } \exists m \leq i_0 \, [A \overset{m}{\Rightarrow} B]\}$. Maar aangezien voor alle $i > i_0$ geldt $N_i(A) = N_{i_0}(A)$, geldt:

$$N_{i_0}(A) = \{B \mid B \in V \setminus \Sigma \text{ en } A \overset{*}{\Rightarrow} B\}.$$

Merk op dat deze zelfde methode van toepassing is op lineaire en contextvrije grammatica's. In het navolgende zal hierop worden teruggekomen.

Definieer bij de grammatica G = (V,Σ,P,S) een gerichte graaf met als verzameling knopen de verzameling V\Σ en als verzameling takken de verzameling $\{(A,A) \mid A \in V\backslash\Sigma\} \cup \{(A,B) \mid A \to B \in P \text{ en } A,B \in V\backslash\Sigma\}$. Het voorgaande algoritme bepaalt voor elke knoop A uit de aldus geconstrueerde graaf, de verzameling knopen die vanuit A via een gericht pad bereikbaar zijn.

3.8. De complexiteit van algoritme 3.7. wordt voor een grammatica G=(V,Σ,P,S) als volgt bepaald: zij card(V\Σ) = n.

- de kosten van de bepaling van $N_0(A)$ zijn constant en de kosten van de bepaling van $N_1(A)$ zijn O(card(P)). De bepaling van $N_{i+1}(A)$ uit $N_i(A)$ kost ten hoogste $\sum_{B \in N_i(A)} \text{card}(V\backslash\Sigma) \leq n^2$.
  Aangezien deze laatste stap hoogstens n maal wordt uitgevoerd, bedragen de kosten van de bepaling van N(A) ten hoogste $O(n^3)$ + O(card(P)). De bepaling van N(A) voor alle $A \in V\backslash\Sigma$, is derhalve $O(n^4)$ + O(n × card(P));
- de kosten van de constructie van de nieuwe produktieregels in fase (1) zijn O(n × card(P));
- laat m de lengte van het maximale rechterlid van de produktieregels zijn, dan bedragen de kosten van het vervangen van één produktieregel in fase (2) ten hoogste m-1. Aangezien alle produktieregels beschouwd moeten worden, is de complexiteit van deze fase van O(m × card(P));
- bepaal zelf de complexiteit van fase (3).

3.9. Voorbeeld

Beschouw de rechtslineaire grammatica G = ({a,b,A,B,S} , {a,b}, P, S) waarin P de volgende herschrijfregels bevat:

S → A
A → abA
A → bB
B → A
B → a

Om een met G equivalente reguliere grammatica te bepalen wordt de constructie uit het bewijs van eigenschap 3.6. toegepast.

(1) In de eerste stap worden alle produktieregels van de vorm $A \to B$ uit P vervangen; hiertoe wordt voor alle hulpsymbolen X de verzameling $N(X) = \{Y \mid X \overset{*}{\Longrightarrow} Y\}$ bepaald volgens het algoritme uit 3.7.

X	$N_0(X)$	$N_1(X)$	$N_2(X)$
S	{S}	{S,A}	{S,A}
A	{A}	{A}	{A}
B	{B}	{B,A}	{B,A}

Hieruit volgt $N(S) = \{S,A\}$, $N(A) = \{A\}$, $N(B) = \{A,B\}$.

De verzameling nieuwe produktieregels P' waarin geen regels van de vorm $A \to B$, $A,B \in V\setminus\Sigma$, voorkomen, bevat de volgende elementen:

$S \to abA$
$S \to bB$
$A \to abA$
$A \to bB$
$B \to abA$
$B \to bB$
$B \to a$

(2) In de uit fase (1) resulterende verzameling produktieregels zijn er drie regels van de vorm $A \to a_1 \ldots a_n B$, $(n \geq 2)$ $A,B \in V\setminus\Sigma$ en $a_1, \ldots, a_n \in \Sigma$.

- $S \to abA$ wordt nu vervangen door het stelsel regels
  $S \to aH_1$
  $H_1 \to bA$, waarin H_1 een nieuw hulpsymbool is.

- $A \to abA$ wordt vervangen door het stelsel regels
  $A \to aH_2$
  $H_2 \to bA$, waarin H_2 een nieuw hulpsymbool is.

- $B \rightarrow abA$ wordt vervangen door het stelsel regels

  $B \rightarrow aH_3$

  $H_3 \rightarrow bA$, waarin H_3 een nieuw hulpsymbool is.

(3) In fase (3) wordt $B \rightarrow a$ vervangen door het stelsel regels

$B \rightarrow aH_4$

$H_4 \rightarrow \lambda$, waarin H_4 een nieuw hulpsymbool is.

Uit de fase (1), (2) en (3) resulteert de volgende reguliere grammatica $G'' = (\{a,b,A,B,S,H_1,H_2,H_3,H_4\}, \{a,b\}, P'', S\}$, waarin P'' de volgende produktieregels bevat:

$S \rightarrow aH_1$
$S \rightarrow bB$
$A \rightarrow aH_2$
$A \rightarrow bB$
$B \rightarrow aH_3$
$B \rightarrow bB$
$B \rightarrow aH_4$
$H_1 \rightarrow bA$
$H_2 \rightarrow bA$
$H_3 \rightarrow bA$
$H_4 \rightarrow \lambda$ •

3.10. Oefening

Construeer een type 3 grammatica die equivalent is met de rechtslineaire grammatica G, waarin $G = (\{S,A,B,a,b\}, \{a,b\}, P, S)$ en $P = \{S \rightarrow abS, S \rightarrow abA, A \rightarrow B, B \rightarrow baB, B \rightarrow ba\}$.

De volgende eigenschap, die de pompstelling (van Rabin en Scott) voor reguliere talen wordt genoemd, wordt gebruikt om aan te tonen dat bepaalde talen geen type 3 taal zijn. Om te bewijzen dat talen niet tot $\mathcal{L}_{lin}$ respectievelijk $\mathcal{L}_2$ behoren, bestaan overeenkomstige eigenschappen (zie 3.15. respectievelijk 3.33.).

3.11. Eigenschap

Voor elk type 3 taal $L \subseteq \Sigma^*$ bestaat er een natuurlijk getal $p \geq 1$, zodanig dat elk woord $w \in L$ met $|w| > p$ geschreven kan worden als $w = xyz$, zodanig dat $0 < |y| \leq p$ en $\forall i \geq 0\ [xy^iz \in L]$.

Bewijs

Zij $G = (V,\Sigma,P,S)$ een rechtslineaire grammatica, zodanig dat $L(G) = L$ en P geen produktieregels van de vorm $A \rightarrow B$, $A,B \in V\setminus\Sigma$ bevat. Zij m de maximale lengte van de rechterleden van de produktieregels in P. Kies nu $p = m \times \text{card}(V\setminus\Sigma)$. Beschouw vervolgens een woord $w \in L$ met $|w| > p$. Er bestaat een afleiding $S = \alpha_0 \Rightarrow \alpha_1 \Rightarrow .. \Rightarrow \alpha_r = w$. Hierin is $r > \text{card}(V\setminus\Sigma)$, immers $r \times m \geq |w| > p = \text{card}(V\setminus\Sigma) \times m$, zodat er in de afleiding van w uit S een α_i en een α_j, $1 \leq i < j < r$, voorkomen zodanig dat het hulpsymbool in α_i gelijk is aan het hulpsymbool in α_j; laat dit het hulpsymbool X zijn. De afleiding van w uit S is nu te schrijven als $S = \alpha_0 \overset{*}{\Rightarrow} \alpha_i = xX \overset{*}{\Rightarrow} \alpha_j = xyX \overset{*}{\Rightarrow} \alpha_r = xyz = w$. Omdat P geen regels van de vorm $A \rightarrow B$ bevat, geldt voor elke regel van de vorm $A \rightarrow wB$ dat $|w| \geq 1$. Dit houdt in dat $y \neq \lambda$.

Het is verder duidelijk dat i en j zodanig gekozen kunnen worden dat $|y| \leq p$: immers als $|y| > p$ is, dan bevat de deelafleiding $\alpha_i \overset{*}{\Rightarrow} \alpha_j$ een herhaling van een hulpsymbool. Uit $X \overset{*}{\Rightarrow} yX$ volgt dan dat $\forall i \geq 0\ [xy^iz \in L]$.

Einde bewijs

3.12. Voorbeelden

(1) Beschouw de taal $L = \{a^nb^n \mid n \geq 0\}$. Met behulp van eigenschap 3.11. wordt aangetoond dat $L \notin \mathcal{L}_3$.

Stel dat $L \in \mathcal{L}_3$, dan bestaat er een natuurlijk getal p, $p \geq 1$, zodanig dat elk woord $w \in L$ met $|w| > p$, geschreven kan worden als $w = xyz$, zodanig dat $0 < |y| \leq p$ en $\forall i \geq 0\ [xy^iz \in L]$. Beschouw het woord $a^pb^p \in L$. Hiervoor geldt $|a^pb^p| > p$. Nu moet a^pb^p te schrijven zijn als xyz, zodanig dat $\forall i \geq 0\ [xy^iz \in L]$, waarbij $0 < |y| \leq p$. Het deelwoord y kan niet zowel a's als b's

bevatten, aangezien dan in xy^2z de alfabetische volgorde wordt verstoord. Het deelwoord y kan echter ook niet alleen uit a's of uit b's bestaan, omdat dan geldt $\#(a,xz) \neq \#(b,xz)$. Aangezien $|y| > 0$ moet zijn, kan $a^p b^p$ niet volgens eigenschap 3.11. in deelwoorden gesplitst worden. Hieruit volgt dat $L \notin \mathcal{L}_3$.

(2) Beschouw de taal $L = \{w \mid w \in \{a,b\}^* \text{ en } \#(a,w) = \#(b,w)\}$. Alle woorden $w \in L$, $|w| > 2$, kunnen nu geschreven worden als $w = xyz$ met $0 < |y| \leq 2$, terwijl $\forall i \geq 0\ [xy^iz \in L]$. Want als $w \in L$ en $|w| > 2$, dan bevat w het deelwoord ab (respectievelijk het deelwoord ba). Schrijf w als $w = xabz$ (respectievelijk $w = xbaz$), dan geldt $0 < |ab| \leq 2$ (respectievelijk $0 < |ba| \leq 2$) en $\forall i \geq 0\ [x(ab)^iz \in L]$. ●

In voorbeeld 3.12. (2) is een taal gedefinieerd die bestand is tegen "pompen" volgens eigenschap 3.11. Hieruit mag niet geconcludeerd worden dat deze taal een type 3 taal is. Eigenschap 3.11. geeft alleen een noodzakelijke en geen voldoende voorwaarde voor het type 3-zijn van een taal.

3.13. Oefeningen

(1) Laat zien dat elke eindige taal bestand is tegen pompen volgens eigenschap 3.11.

(2) Laat R_0 de klasse van alle talen zijn die bestand zijn tegen pompen volgens eigenschap 3.11. Vervang in eigenschap 3.11. de eis "$\forall i \geq 0\ [xy^iz \in L]$" door de eis "$\forall i > 0\ [xy^iz \in L]$". Laat R_1 de klasse van talen aanduiden die bestand zijn tegen pompen volgens deze versie van eigenschap 3.11. Bedenk een taal uit $R_1 \setminus R_0$ en een taal uit $R_0 \setminus \mathcal{L}_3$. Laat zien dat de door u bedachte talen tot de respectievelijke klassen behoren.

3.14. Eigenschap

$\mathcal{L}_3 \subset \mathcal{L}_2$

Bewijs

In eigenschap 3.4. is aangetoond dat $\mathcal{L}_3$ een deelverzameling is van $\mathcal{L}_2$. Om aan te tonen dat $\mathcal{L}_3$ een echte deelverzameling van $\mathcal{L}_2$ is, is het voldoende een taal L aan te geven, waarvoor geldt $L \notin \mathcal{L}_3$ en $L \in \mathcal{L}_2$, zodat $L \in \mathcal{L}_2 \setminus \mathcal{L}_3$.

Beschouw de taal $L = \{a^n b^n \mid n \geq 0\}$.

(1) In voorbeeld 3.12. (1) is aangetoond dat $L \notin \mathcal{L}_3$.

(2) De grammatica $G = (\{S,a,b\}, \{a,b\}, P, S)$ met $P = \{S \to aSb, S \to \lambda\}$ is een type 2 grammatica, waarvoor geldt dat $L(G) = L$ (Toon dit zelf aan). Hieruit volgt $L \in \mathcal{L}_2$

Uit (1) en (2) samen volgt hetgeen aan te tonen was.

Einde bewijs

De grammatica G uit het bewijs van eigenschap 3.14. is niet alleen contextvrij maar zelfs lineair. Om te kunnen aantonen dat er ook contextvrije talen zijn die niet lineair zijn, wordt er een pompstelling voor lineaire talen geformuleerd.

3.15. Eigenschap

Voor elke lineaire taal $L \subseteq \Sigma^*$ bestaat er een natuurlijk getal $p \geq 1$ zodanig dat elk woord $w \in L$ met $|w| > p$ geschreven kan worden als $w = xuvyz$, zodanig dat $0 < |xuyz| \leq p$, $|uy| \geq 1$ en $\forall i \geq 0\ [xu^i vy^i z \in L]$.

Bewijs

Zij $G = (V,\Sigma,P,S)$ een lineaire grammatica, zodanig dat $L(G) = L$ en P geen produktieregels van de vorm $A \to B$, $A,B \in V\setminus\Sigma$ bevat. (Toon aan dat bij elke lineaire grammatica G' een equivalente lineaire grammatica G te construeren is, die aan bovengenoemde eis voldoet). Zij m het maximum van de lengten van de rechterleden van de produktieregels in P. Kies nu $p = m \times \text{card}(V\setminus\Sigma)$. Zij $w \in L$, met $|w| > p$. Er bestaat een afleiding

$$S = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_r = w.$$

Hierin is $|w| \leq (r-1)(m-1) + m = rm - (r-1) < rm$, als $r \geq 2$. Aangezien $|w| > p$, is $r > \text{card}(V\setminus\Sigma)$, zodat er in de afleiding van w uit S een α_i

en een α_j, $1 \leq i < j < r$, bestaan, zodanig dat het hulpsymbool in α_i gelijk is aan het hulpsymbool in α_j; noem dit hulpsymbool X. De afleiding van w uit S is nu te schrijven als

$$S = \alpha_0 \overset{*}{\Longrightarrow} xXz \overset{*}{\Rightarrow} xuXyz \overset{*}{\Rightarrow} xuvyz = \alpha_r = w.$$

Omdat voor elke produktieregel $A \to wBw'$ in P, $w,w' \in \Sigma^*$, $A,B \in V\backslash\Sigma$, geldt dat $|wBw'| \geq 2$, is $uy \neq \lambda$. Het is verder duidelijk dat i en j zodanig gekozen kunnen worden dat $|xuyz| \leq p$: immers als $|xuyz| > p$, zou in de deelafleiding $\alpha_0 \overset{*}{\Longrightarrow} \alpha_{j-1}$ een herhaling van een hulpsymbool voorkomen. Uit $X \overset{*}{\Rightarrow} uXy$ volgt dat $\forall i \geq 0\ [xu^ivy^iz \in L]$.
Einde bewijs

3.16. Eigenschap

$$\mathcal{L}_3 \subset \mathcal{L}_{lin} \subset \mathcal{L}_2$$

Bewijs

(1) Uit de definities 1.36. en 1.39. volgt dat $\mathcal{L}_3$ een deelverzameling van $\mathcal{L}_{lin}$ is.

(2) Uit het bewijs van eigenschap 3.14. volgt dat voor de taal $L = \{a^nb^n \mid n \geq 0\}$ geldt dat $L \notin \mathcal{L}_3$ en $L \in \mathcal{L}_{lin}$, zodat $\mathcal{L}_3$ een echte deelverzameling van $\mathcal{L}_{lin}$ is.

Uit (1) en (2) samen volgt $\mathcal{L}_3 \subset \mathcal{L}_{lin}$.

(3) Uit de definities 1.35. en 1.36. volgt dat $\mathcal{L}_{lin}$ een deelverzameling van $\mathcal{L}_2$ is.

(4) Om aan te tonen dat $\mathcal{L}_{lin}$ een echte deelverzameling van $\mathcal{L}_2$ is, is het voldoende een taal L aan te geven, waarvoor geldt $L \notin \mathcal{L}_{lin}$ en $L \in \mathcal{L}_2$.

Beschouw de taal $L = \{a^mb^ma^nb^n \mid m,n \geq 0\}$.

- Stel dat $L \in \mathcal{L}_{lin}$, dan bestaat er volgens eigenschap 3.15. een natuurlijk getal $p \geq 1$, zodanig dat elk woord $w \in L$ met $|w| > p$ geschreven kan worden als $w = xuvyz$, zodanig dat $0 < |xuyz| \leq p$, $|uy| \geq 1$ en $\forall i \geq 0\ [xu^ivy^iz \in L]$. Beschouw

het woord $aPbPaPbP \in L$, dan geldt $|aPbPaPbP| > p$. Nu moet $aPbPaPbP$ te schrijven zijn als $xuvyz$, zodanig dat $\forall i > 0\ [xu^ivy^iz \in L]$. Het woord $bPaP$ is een deelwoord van v: het deelwoord u kan niet zowel a's als b's bevatten aangezien $|xu| \leq |xuyz| \leq p$, zodat u alleen a's kan bevatten. Op analoge wijze kan worden aangetoond dat y alleen uit b's kan bestaan. Voor zekere s en t geldt nu $v = a^sbPaPb^t$. Maar dan is $xu^0vy^0z = xa^sbPaPb^tz$ geen element van L, aangezien $|x| + s < p$ en/of $|z| + t < p$. Het woord $aPbPaPbP$ kan derhalve niet volgens eigenschap 3.15. gesplitst worden, zodat $L \notin \mathcal{L}_{lin}$.

- De grammatica $G = (\{S,T,a,b\}, \{a,b\}, P, S)$ met $P = \{S \to TT, T \to aTb, T \to \lambda\}$ is een type 2 grammatica waarvoor geldt dat $L(G) = L$ (Toon dit zelf aan).

Uit (3) en (4) samen volgt $\mathcal{L}_{lin} \subset \mathcal{L}_2$.
Einde bewijs

In het voorgaande is een afleiding telkens gerepresenteerd als een reeks $\alpha_0 \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots$ van zinsvormen. In het geval van contextvrije grammatica's kunnen afleidingen ook met behulp van (geordende geëtiketteerde gewortelde) bomen worden weergegeven. Een dergelijke boom wordt een afleidingsboom genoemd. In definitie 3.17. wordt het begrip afleidingsboom gedefinieerd en in eigenschap 3.20 wordt de relatie tussen afleidingsbomen en afleidingen gelegd.

3.17. Definitie: Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica. Een (geordende geëtiketteerde gewortelde) boom is een G-afleidingsboom (Eng.: "G-derivation tree") als aan de volgende condities is voldaan:

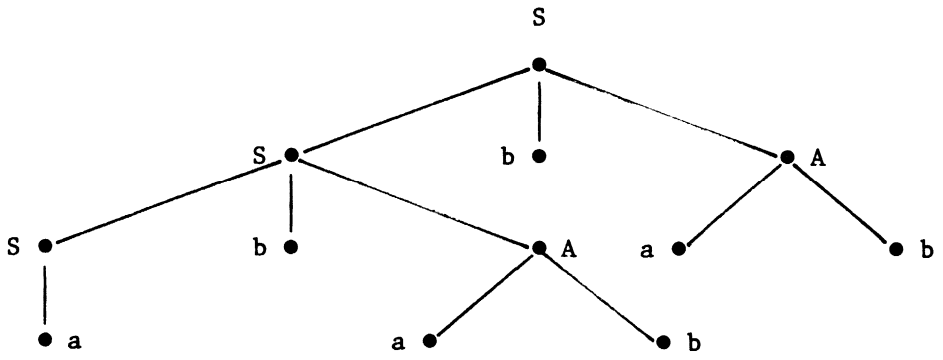
- elke knoop in de boom heeft een etiket uit V;
- inwendige knopen hebben etiketten uit $V \setminus \Sigma$;
- als een inwendige knoop het etiket A draagt en de zonen van A van links naar rechts de etiketten $X_1,\ldots,X_k$, $k \geq 1$, hebben, dan is $A \to X_1 \ldots X_k$ een produktieregel in P.

Als duidelijk is om welke grammatica het handelt, wordt de term afleidingsboom in plaats van G-afleidingsboom gehanteerd.

3.18. Definitie: De symboolrij die wordt verkregen door de etiketten van de bladeren van een afleidingsboom van links naar rechts na elkaar te schrijven, wordt de opbrengst (Eng.: "yield") van die afleidingsboom genoemd.

3.19. Voorbeeld

$G = (\{S,A,a,b\}, \{a,b\}, P, S)$ met $P = \{S \to SbA,\ S \to a,\ A \to ab\}$ is een contextvrije grammatica. De volgende boom is een G-afleidingsboom:



De opbrengst van deze afleidingsboom is de symboolrij ababbab. De opbrengst van de volgende afleidingsboom is de symboolrij a:

3.20. Eigenschap

Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica. Voor elke $X \in V \setminus \Sigma$ en $\alpha \in \Sigma^*$ geldt $X \overset{*}{\underset{G}{\Longrightarrow}} \alpha$, dan en slechts dan als er een G-afleidingsboom met worteletiket X en opbrengst α is.

Bewijs

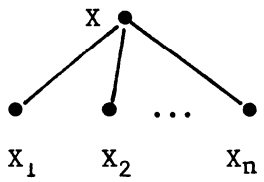
Het bewijs van deze eigenschap verloopt in twee stappen:

(1) Zij T een G-afleidingsboom met worteletiket X, $X \in V \setminus \Sigma$ en opbrengst α. Met inductie naar de diepte van de boom T wordt aangetoond dat $X \overset{*}{\underset{G}{\Rightarrow}} \alpha$.

Initialisatiestap: als T de diepte nul heeft, bestaat T uit één enkele knoop met etiket X. De opbrengst α van T is eveneens X, zodat $X \overset{*}{\underset{G}{\Rightarrow}} \alpha$.

Inductieveronderstelling: neem aan dat $\forall i \leq k$, $k \geq 0$ en voor elke G-afleidingsboom T met diepte i, worteletiket X, $X \in V \setminus \Sigma$ en opbrengst α, geldt dat $X \overset{*}{\underset{G}{\Rightarrow}} \alpha$.

Inductiestap: zij T een G-afleidingsboom met diepte k+1, worteletiket X, $X \in V \setminus \Sigma$ en opbrengst α. Stel dat de etiketten van de zonen van de wortel van T van links naar rechts $X_1, \ldots, X_n$ zijn:



dan moet $X \rightarrow X_1 \ldots X_n \in P$. Zij T_i de deelboom van T met als wortel de zoon met etiket X_i en opbrengst α_i, $1 \leq i \leq n$. Dan geldt $\alpha = \alpha_1 \ldots \alpha_n$. De diepte van T_i is kleiner dan of gelijk aan k, zodat volgens de inductieveronderstelling $X_i \overset{*}{\Rightarrow} \alpha_i$, $1 \leq i \leq n$. Uit het voorgaande volgt $X \Rightarrow X_1 \ldots X_n \overset{*}{\Rightarrow} \alpha_1 \ldots \alpha_n = \alpha$.

(2) Stel $X \overset{*}{\underset{G}{\Rightarrow}} \alpha$. Toon met inductie naar de lengte van de afleiding aan dat er een G-afleidingsboom T bestaat met worteletiket X en opbrengst α.

Einde bewijs

Bij een gegeven afleiding $X = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_n$, $n \geq 0$, hoort precies één afleidingsboom. Bij één afleidingsboom behoren in het algemeen verscheidene afleidingen. Twee daarvan worden apart onderscheiden: de lin-

ker- (Eng.: "left-most") en de rechter- (Eng.: "right-most") afleiding. Een afleiding is een linker-(rechter-)afleiding als in elke stap het meest linkse (rechtse) hulpsymbool wordt herschreven. Een meer formele definitie luidt:

3.21. Definitie: Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica.

(1) Een afleiding $\alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_n$ is een linkerafleiding als en alleen als voor $\forall i$, $0 \leq i < n$, geldt
$\alpha_i = w_i A \beta_i$ en $\alpha_{i+1} = w_i \delta \beta_i$, $A \in V \setminus \Sigma$, $A \to \delta \in P$, $\beta_i, \delta \in V^*$ en $w_i \in \Sigma^*$.

(2) Een afleiding $\alpha_0 \Rightarrow \alpha_1 \Rightarrow \ldots \Rightarrow \alpha_n$ is een rechterafleiding als en alleen als voor $\forall i$, $0 \leq i < n$, geldt
$\alpha_i = \beta_i A w_i$ en $\alpha_{i+1} = \beta_i \delta w_i$, $A \in V \setminus \Sigma$, $A \to \delta \in P$, $\beta_i, \delta \in V^*$ en $w_i \in \Sigma^*$.

3.22. Oefening

De contextvrije grammatica $G = (\{S,A,B,a,b\}, \{a,b\}, P, S)$ bevat de produktieregels

S → aB
S → bA
A → a
A → aS
A → bAA
B → b
B → bS
B → aBB

Bepaal voor het woord aaabbabbba
(1) een afleidingsboom;
(2) een linkerafleiding;
(3) een rechterafleiding.

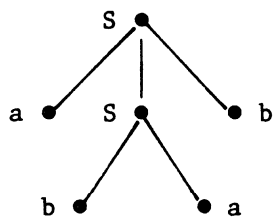
3.23. Voorbeeld

Het kan voorkomen dat in een grammatica twee of meer verschillende afleidingsbomen eenzelfde woord opbrengen. Zij G = ({S,a,b}, {a,b}, P,S) waarin P de volgende produktieregels bevat

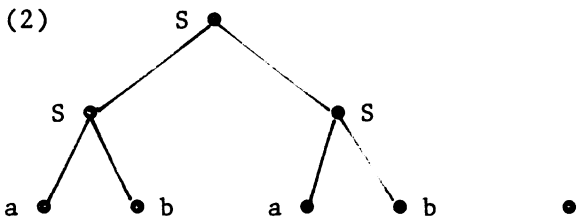
S → SS
S → aSb
S → bSa
S → ab
S → ba

Beschouw het woord abab ∈ L(G). Bij dit woord behoren minstens twee verschillende afleidingsbomen:

(1)



(2)



3.24. Definitie: Zij G een contextvrije grammatica. Een woord w heet (syntactisch) dubbelzinnig (Eng.: "ambiguous"), volgens G als er tenminste twee verschillende G-afleidingsbomen bestaan, die w opbrengen.

Een contextvrije grammatica G heet dubbelzinnig als er tenminste één woord w ∈ L(G) bestaat, dat volgens G dubbelzinnig is.

Een contextvrije taal L heet dubbelzinnig als elke grammatica die L voortbrengt, dubbelzinnig is.

3.25. Oefening

(1) De grammatica G = ($\{S,A,B,a,b\}$, $\{a,b\}$, P, S) bevat de produktieregels

$S \to bA$
$S \to aB$
$A \to a$
$A \to aS$
$A \to bAA$
$B \to b$
$B \to bS$
$B \to aBB$

Bepaal L(G) en laat zien dat G dubbelzinnig is.

(2) Beschouw de grammatica G uit voorbeeld 3.19. Is L(G) dubbelzinnig?

(3) Toon aan dat bij elke contextvrije grammatica G, zodanig dat $L(G) \neq \emptyset$, er een equivalente contextvrije grammatica geconstrueerd kan worden, die dubbelzinnig is.

3.26. Eigenschap

Het probleem U = [G is een contextvrije grammatica: G is dubbelzinnig] is algoritmisch onoplosbaar.

Bewijs

Het bewijs dat U algoritmisch onoplosbaar is, verloopt via reductie van U naar het PCP, dat blijkens 2.24. algoritmisch onoplosbaar is.

Zij Σ een alfabet, en PCP(n, $a_1,\ldots,a_n$, $b_1,\ldots,b_n$), $n \geq 1$, $a_1,\ldots,a_n$, $b_1,\ldots,b_n \in \Sigma^+$ een exemplaar van het PCP. Beschouw de grammatica $G = (\Sigma \cup \{S,S_1,S_2,x_1,\ldots,x_n\}, \Sigma \cup \{x_1,\ldots,x_n\}, P, S)$, waarin de x_i's nieuw gekozen symbolen zijn en P de volgende produktieregels bevat:

$S \rightarrow S_1$

$S \rightarrow S_2$

$S_1 \rightarrow a_i S_1 x_i, \qquad \forall\, i \leq n$

$S_1 \rightarrow a_i x_i, \qquad \forall\, i \leq n$

$S_2 \rightarrow b_i S_2 x_i, \qquad \forall\, i \leq n$

$S_2 \rightarrow b_i x_i, \qquad \forall\, i \leq n$

De grammatica G is dubbelzinnig dan en alleen dan als het gegeven exemplaar van het PCP de uitkomst waar oplevert:

- als het exemplaar van het PCP de uitkomst waar oplevert, dan is er een rijtje indices $i_1,\ldots,i_m$, zodanig dat

  $a_{i_1}\ldots a_{i_m} = b_{i_1}\ldots b_{i_m}$. Dan is er een woord $a_{i_1}\ldots a_{i_m} x_{i_m}\ldots x_{i_1} \in L(G)$

  dat gelijk is aan het woord

  $b_{i_1}\ldots b_{i_m} x_{i_m}\ldots x_{i_1} \in L(G)$.

  Er bestaat voor dit woord een linkerafleiding beginnend met

  $S \Rightarrow S_1 \overset{*}{\Rightarrow} a_{i_1}\ldots a_{i_m} x_{i_m}\ldots x_{i_1}$

  en een linkerafleiding beginnend met

  $S \Rightarrow S_2 \overset{*}{\Rightarrow} a_{i_1}\ldots a_{i_m} x_{i_m}\ldots x_{i_1}$.

  Hieruit volgt dat G dubbelzinnig is.

- Stel G is dubbelzinnig. Aangezien in elk woord $w \in L(G)$, zodanig dat $S \Rightarrow S_1 \overset{*}{\Rightarrow} w$, in het grootste deelwoord van w over $\{x_1,\ldots,x_n\}$ de volgorde van de in de afleiding van w toegepaste produktieregels

is vastgelegd, heeft w een unieke linkerafleiding. Een analoge redenering is geldig voor elk woord $w \in L(G)$, zodanig dat $S \Longrightarrow S_2 \overset{*}{\Longrightarrow} w$. Omdat G dubbelzinnig is, is er een woord $w \in L(G)$, zodanig dat $S \Longrightarrow S_1 \overset{*}{\Longrightarrow} w$ en $S \Longrightarrow S_2 \overset{*}{\Longrightarrow} w$. Als $w = yx_{i_m}..x_{i_1}$, $y \in \Sigma^*$, dan is $i_1,\ldots,i_m$ een rijtje indices waarvoor voor het exemplaar van het PCP geldt

$$a_{i_1}\ldots a_{i_m} = b_{i_1}\ldots b_{i_m}.$$

Het exemplaar van het PCP levert derhalve de uitkomst waar op.

Aangezien de grammatica G daadwerkelijk kan worden geconstrueerd uit het gegeven exemplaar van het PCP, bestaat er een algoritme voor het PCP als er een algoritme voor U bestaat. Hieruit volgt dat U algoritmisch onoplosbaar is.
Einde bewijs

3.27. Eigenschap

Bij elke contextvrije grammatica G is een equivalente contextvrije grammatica G' te construeren, waarin geen produktieregels van de vorm $A \to \lambda$ voorkomen, met als enige mogelijke uitzondering de regel $S' \to \lambda$, waarin S' het startsymbool van G' is. In het geval dat $S' \to \lambda$ een regel van G' is, komt S' niet in de rechterleden van de produktieregels van G' voor.

Bewijs
Stel $G = (V,\Sigma,P,S)$ en $G' = (V',\Sigma,P',S')$, en definieer $U = \{A \mid A \in V\backslash\Sigma$ en $A \overset{*}{\underset{G}{\Longrightarrow}} \lambda\}$. Definieer de verzameling P_0 van produktieregels aldus:

$P_0 = \{A \to w_0 \mid A \to w \in P$, $w_0 \neq \lambda$ en w_0 onstaat uit w door daaruit 0 of meer symbolen uit U te schrappen$\}$.

Er worden twee gevallen onderscheiden:

- $S \in U$, zodat $\lambda \in L(G)$. Voer een nieuw hulpsymbool S' in. Stel $V' = V \cup \{S'\}$ en $P' = P_0 \cup \{S' \to S, S' \to \lambda\}$.
- $S \notin U$, zodat $\lambda \notin L(G)$. Stel $S' = S$, $V' = V$ en $P' = P_0$.

Nu geldt $L(G) = L(G')$. Het bewijs van deze uitspraak verloopt in drie stappen:

(1) Uit de constructie van G' volgt direct dat $\lambda \in L(G')$ dan en slechts dan als $S' \to \lambda \in P'$, dan en slechts dan als $S \in U$, dan en slechts dan als $\lambda \in L(G)$.

(2) Voor elke produktieregel $A \to A_1 \ldots A_k$ in P', $k \geqslant 1$, is er een produktieregel $A \to \alpha_1 A_1 \ldots \alpha_k A_k \alpha_{k+1}$ in P waarbij $\alpha_i \in U^*$ $1 \leqslant i \leqslant k+1$. Aangezien $\alpha_i \overset{*}{\underset{G}{\Rightarrow}} \lambda$, geldt

$$A \underset{G}{\Rightarrow} \alpha_1 A_1 \ldots \alpha_k A_k \alpha_{k+1} \overset{*}{\underset{G}{\Rightarrow}} A_1 \ldots A_k.$$

Hieruit volgt met inductie naar de afleidingslengte:

als $\alpha \overset{*}{\underset{G'}{\Rightarrow}} \beta$ dan $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$. Dus $L(G') \subseteq L(G)$.

(3) Met inductie naar de afleidingslengte kan worden aangetoond dat $\forall X \in V \setminus \Sigma$, $\forall w \in \Sigma^*$ [<u>als</u> $X \overset{*}{\underset{G}{\Rightarrow}} w$ <u>dan</u> $X \overset{*}{\underset{G'}{\Rightarrow}} w$]. Dit houdt in dat $L(G) \subseteq L(G')$.

De in (1), (2) en (3) bepaalde grammatica kan worden geconstrueerd; want
(1) U kan daadwerkelijk worden geconstrueerd (zie 3.28.)
(2) Uit U kan P_0 daadwerkelijk worden geconstrueerd.
(3) De resterende stappen zijn eveneens effectief uitvoerbaar.
<u>Einde bewijs</u>

3.28. Oefening

Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica. Schrijf een algoritme dat de verzameling $U = \{A \mid A \in V\setminus\Sigma \text{ en } A \overset{*}{\underset{G}{\Longrightarrow}} \lambda\}$ bepaalt. Laat zien dat uw algoritme voor elke contextvrije grammatica termineert en correct is. (Aanwijzing: bepaal de rij verzamelingen $U_0,U_1,\ldots,U_n = U$ (met $n \leq \text{card}(V\setminus\Sigma)$) als volgt:

$$U_0 = \{A \mid A \in V\setminus\Sigma \text{ en } A \to \lambda \in P\}$$
$$U_{i+1} = U_i \cup \{A \mid A \in V\setminus\Sigma \text{ en } A \to \alpha \in P,\ \alpha \in U_i^+\}).$$

Bepaal de complexiteit van uw algoritme.

3.29. Voorbeeld

Beschouw de contextvrije grammatica $G = (\{S,A,B,C,a,b,c,d\}, \{a,b,c,d\}, P, S)$ met de volgende produktieregels:

$S \to aSa$
$S \to A$
$S \to B$
$A \to cA$
$A \to \lambda$
$B \to bBa$
$B \to AC$
$C \to dC$
$C \to \lambda$

$U = \{A \mid A \in V\setminus\Sigma \text{ en } A \overset{*}{\underset{G}{\Longrightarrow}} \lambda\} = \{S,A,B,C\}$. De verzameling P_0 van produktieregels bevat de volgende regels:

$S \to aSa$
$S \to aa$
$S \to A$
$S \to B$
$A \to cA$
$A \to c$
$B \to bBa$
$B \to ba$
$B \to AC$
$B \to A$
$B \to C$
$C \to dC$
$C \to d$

De met G equivalente contextvrije grammatica G', waarin geen produktieregels van de vorm $A \to \lambda$ voorkomen, met als enige uitzondering de regel

$S' \to \lambda$ waarin S' het startsymbool van G' is, is de grammatica G' = $(\{S,S',A,B,C,a,b,c\}, \{a,b,c\}, P_0 \cup \{S' \to S, S' \to \lambda\}, S')$. •

Zij k de som van de lengten van de rechterleden van de produktieregels in P. De verzameling P_0 kan dan hoogstens card(P) × (2^k-1) produktieregels bevatten: een produktieregel van de vorm $A \to B_1 \ldots B_k$, $k \geq 1$ in P, leidt immers tot 2^k-1 produktieregels in P_0. De constructie uit het bewijs van eigenschap 3.27. vergt derhalve exponentiële tijd. Zonder bewijs wordt vermeld dat er een algoritme bestaat dat de met G equivalente grammatica zonder regels van de vorm $A \to \lambda$ in lineaire tijd construeert.

3.30. Eigenschap

$\mathcal{L}_2 \subseteq \mathcal{L}_1$

Bewijs
De eigenschap volgt direct uit de definities 1.34. en 1.35. en eigenschap 3.27.
Einde bewijs

3.31. Eigenschap

Bij elke contextvrije grammatica G is er een equivalente contextvrije grammatica G' te construeren die geen produktieregels bevat van de vorm $A \to B$, waarin A en B hulpsymbolen zijn.

3.32. Oefening

Bewijs eigenschap 3.31. (Aanwijzing: bekijk het bewijs van eigenschap 3.6.).

In het voorgaande zijn er pompstellingen voor reguliere talen en lineaire talen geformuleerd; een dergelijke stelling bestaat ook voor contextvrije talen, de pompstelling voor contextvrije talen van Bar-Hillel, Perles en Shamir.

3.33. Eigenschap

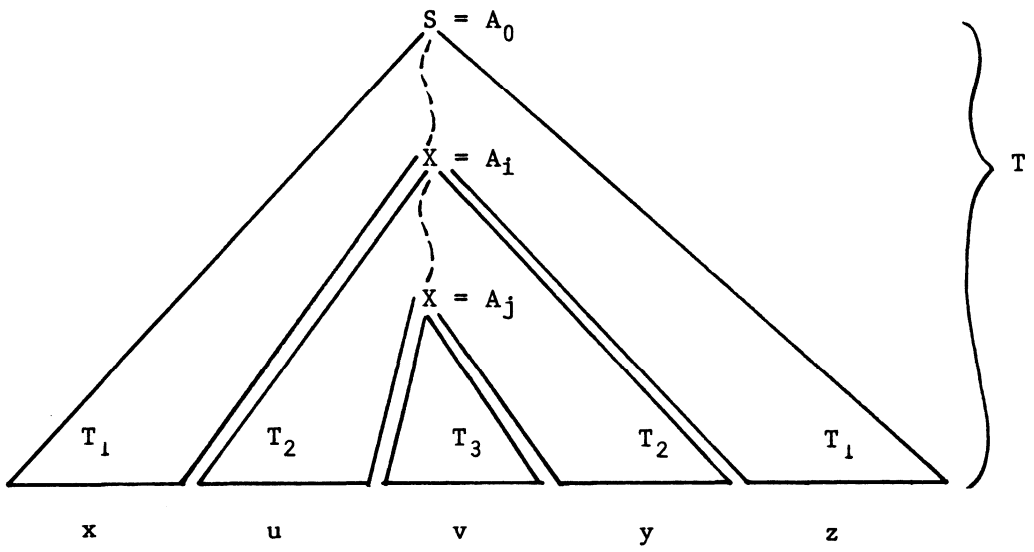
Voor elke contextvrije taal $L \subseteq \Sigma^*$ bestaat er een natuurlijk getal $p \geq 1$, zodanig dat elk woord $w \in L$ met $|w| > p$ geschreven kan worden als $w = xuvyz$, zodanig dat $|uy| \geq 1$, $|uvy| \leq p$ en $\forall i \geq 0\ [xu^ivy^iz \in L]$.

Bewijs

Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica, zodanig dat

- $L(G) = L$
- P geen produktieregels van de vorm $A \to B$, $A,B \in V\setminus\Sigma$, en geen regels van de vorm $A \to \lambda$, $A \in (V\setminus\Sigma)\setminus\{S\}$, bevat. Als $S \to \lambda \in P$, dan komt S nergens in het rechterlid van een produktieregel voor (Zie eigenschap 3.27.).

Zij m het maximum van de lengten van de rechterleden van de produktieregels van G. Kies $p = m^{card(V\setminus\Sigma)+1}$. Zij $w \in L$, met $|w| > p$. In elke G-afleidingsboom T met opbrengst w is er een pad met lengte $n \geq card(V\setminus\Sigma)+1$. Zij $A_0 = S, A_1, \ldots, A_n = a$, $A_i \in V\setminus\Sigma$, $0 \leq i < n$, $a \in \Sigma$, de etiketten van de knopen op dit pad. Op dit pad komen $n > card(V\setminus\Sigma)$ hulpsymbolen voor, die niet allemaal verschillend kunnen zijn. Zij i het grootste natuurlijke getal, waarvoor er een $j > i$ is, zodanig dat $A_i = A_j$. Zij $A_i = X$. Beschouw drie deelbomen van de G-afleidingsboom T (zie de onderstaande figuur):



- T_1 is de boom die van T overblijft nadat de subboom T' met worteletiket A_i uit T verwijderd is.
- T_2 is de boom die van T' overblijft nadat de subboom met worteletiket A_j uit T' verwijderd is.
- T_3 is de subboom met worteletiket A_j van T.

Met T_1 correspondeert een afleiding $S \overset{*}{\Longrightarrow} xXz$, $x,z \in \Sigma^*$. Met T_2 correspondeert een afleiding $X \overset{+}{\Longrightarrow} uXy$, $uy \in \Sigma^+$. Met T_3 tenslotte correspondeert een afleiding $X \overset{+}{\Longrightarrow} v$, $v \in \Sigma^+$. Het woord w kan dus geschreven worden als $w = xuvyz$, zodanig dat $\forall i \geq 0\ [xu^ivy^iz \in L]$. Bovendien is $uy \neq \lambda$, aangezien P geen produktieregels van de vorm $A \to B$, $A,B \in V\backslash\Sigma$ bevat. Uit het feit dat i maximaal gekozen is, volgt bovendien dat de diepte van T_2 ten hoogste $\text{card}(V\backslash\Sigma) + 1$ is, zodat $|uvy| \leq p$.
Einde bewijs

3.34. Eigenschap

$\mathcal{L}_2 \subset \mathcal{L}_1$

Bewijs
In eigenschap 3.30. is aangetoond dat $\mathcal{L}_2$ een deelverzameling van $\mathcal{L}_1$ is. Om aan te tonen dat $\mathcal{L}_2$ een echte deelverzameling van $\mathcal{L}_1$ is, is het voldoende een taal L aan te geven, waarvoor geldt $L \notin \mathcal{L}_2$ en $L \in \mathcal{L}_1$, zodat $L \in \mathcal{L}_1 \backslash \mathcal{L}_2$.

Beschouw de taal $L = \{a^nb^nc^n \mid n \geq 1\}$.

(1) Stel dat L contextvrij is, dan bestaat er volgens eigenschap 3.33. een natuurlijk getal p, $p \geq 1$, zodanig dat elk woord $w \in L$ met $|w| > p$, geschreven kan worden als $w = xuvyz$, zodanig dat $|uy| \geq 1$, $|uvy| \leq p$ en $\forall i \geq 0\ [xu^ivy^iz \in L]$. Beschouw het woord $aPbPcP \in L$. Hiervoor geldt $|aPbPcP| > p$. Nu moet aPbPcP te schrijven zijn als xuvyz, zodanig dat $\forall i \geq 0$ $[xu^ivy^iz \in L]$, waarbij $uy \neq \lambda$ en $|uvy| \leq p$. Als het deelwoord y of het deelwoord u meer dan één soort letter bevat, is in xu^2vy^2z de alfabetische volgorde verstoord. Als y en u elk hoogstens één soort letter bevatten, geldt $\#(a,xvz) = \#(b,xvz) = \#(c,xvz)$ niet.

Aangezien uy $\neq \lambda$ moet gelden, kan aPbPcP niet volgens eigenschap 3.33. gesplitst worden. Hieruit volgt dat $L \notin \mathcal{L}_2$.

(2) De grammatica G = ({S,A,B,C,D,E,a,b,c}, {a,b,c}, P, S) met de produktieregels

S → aSB
S → abC
CB → CD
CD → ED
ED → EC
EC → BC
bB → bbC
C → c

is een contextgevoelige grammatica waarvoor geldt dat L(G) = L. (Toon dit zelf aan). Hieruit volgt $L \in \mathcal{L}_1$.

Uit (1) en (2) samen volgt $\mathcal{L}_2 \subset \mathcal{L}_1$.
Einde bewijs

3.35. Eigenschap

$\mathcal{L}_1 \subseteq \mathcal{L}_0$

Bewijs
Deze eigenschap volgt direct uit de definities 1.32. en 1.34.
Einde bewijs

3.36. Eigenschap

$\mathcal{L}_1 = \mathcal{L}_{verl}$

Bewijs
Het bewijs van deze eigenschap verloopt in twee stappen.
(1) Uit de definities 1.33. en 1.34. volgt $\mathcal{L}_1 \subseteq \mathcal{L}_{verl}$.

(2) Aangetoond moet worden dat bij elke verlengende grammatica een equivalente contextgevoelige grammatica kan worden geconstrueerd. De onderstaande constructie geeft aan hoe één verlengende produktieregel vervangen kan worden door een stelsel contextgevoelige regels.

Zij $G' = (V,\Sigma,P,S)$ een contextgevoelige grammatica, waarin de produktieregel $S \to \lambda$ niet voorkomt, en laat $A_1 \ldots A_m$ en $B_1 \ldots B_n$, $2 \leq m \leq n$, twee woorden over $V \setminus \Sigma$ zijn. Definieer een grammatica $G_0 = (V,\Sigma,P \cup \{A_1 \ldots A_m \to B_1 \ldots B_n\}, S)$, dan is G_0 equivalent met de type 1 grammatica $G_0' = (V \cup \{S_1,\ldots,S_m\}, \Sigma, P \cup P', S)$, waarin $S_1,\ldots,S_m$ nieuwe symbolen zijn en P' de volgende produktieregels bevat:

$$A_1A_2\ldots A_m \to S_1A_2\ldots A_m$$

$$S_1A_2\ldots A_m \to S_1S_2A_3\ldots A_m$$

$$\vdots$$

$$S_1\ldots S_{m-1}A_m \to S_1\ldots S_{m-1}S_mB_{m+1}\ldots B_n$$

$$S_1\ldots S_mB_{m+1}\ldots B_n \to B_1S_2\ldots S_mB_{m+1}\ldots B_n$$

$$\vdots$$

$$B_1\ldots B_{m-1}S_mB_{m+1}\ldots B_n \to B_1\ldots B_{m-1}B_mB_{m+1}\ldots B_n$$

Het is duidelijk dat G_0' een contextgevoelige grammatica is en dat $L(G_0) \subseteq L(G_0')$; $L(G_0') \subseteq L(G_0)$ volgt uit het feit dat de produktieregels in P' zodanig zijn opgesteld dat ze of geen van allen of allemaal na elkaar worden uitgevoerd bij een afleiding in G_0'.

Zij $G = (V,\Sigma,P,S)$ een willekeurige verlengende grammatica. De met G equivalente contextgevoelige grammatica $G' = (V',\Sigma,P',S')$ wordt als volgt geconstrueerd:

- als P de produktieregel $S \to \lambda$ bevat, wordt er een nieuw hulpsymbool S' gekozen en wordt de regel $S \to \lambda$ vervangen door de regels $S' \to \lambda$ en $S' \to S$. Als P de produktieregel $S \to \lambda$ niet bevat, wordt het startsymbool S' van de te construeren grammatica gelijk aan S gekozen.

- vervang in alle produktieregels uit P elk eindsymbool $a \in \Sigma$ door het nieuwe hulpsymbool A_a en voeg aan de aldus verkregen verzameling produktieregels voor elk eindsymbool a de produktieregel $A_a \to a$ toe.

- pas op elke niet-contextgevoelige produktieregel uit de zo verkregen verzameling produktieregels de beschreven constructie toe.

Zij P' de resulterende verzameling produktieregels en V' het met de nieuwe hulpsymbolen uitgebreide alfabet, dan is het duidelijk dat G' een contextgevoelige grammatica is, die equivalent is met G.
Einde bewijs

3.37. Oefeningen

(1) Voer de constructie uit het bewijs van eigenschap 3.36. uit op de grammatica $G = (\{S,B,C,a,b,c\}, \{a,b,c\}, P, S)$, waarin $P = \{S \to aSB, S \to abC, CB \to BC, bB \to bbC, C \to c\}$.

(2) In de constructie uit het bewijs van eigenschap 3.36. wordt er in het geval dat de regel $S \to \lambda$ in de verlengende grammatica voorkomt, een nieuw startsymbool S' gekozen en worden de regels $S' \to \lambda$ en $S' \to S$ aan de contextgevoelige grammatica toegevoegd. Waarom kan de produktieregel $S \to \lambda$ niet direct in de contextgevoelige grammatica worden opgenomen?

(3) Stel dat in de constructie uit het bewijs van eigenschap 3.36. de regel $A_1A_2 \ldots A_m \to B_1B_2 \ldots B_n$ vervangen wordt door het stelsel regels

$$
\begin{aligned}
A_1A_2\cdots A_m &\rightarrow S_1A_2\cdots A_m\\
&\ \ \vdots\\
S_1S_2\cdots A_m &\rightarrow S_1S_2\cdots S_mB_{m+1}\cdots B_n\\
S_1\cdots S_mB_{m+1}\cdots B_n &\rightarrow S_1\cdots S_{m-1}B_mB_{m+1}\cdots B_n\\
&\ \ \vdots\\
S_1B_2\cdots B_n &\rightarrow B_1\cdots B_n
\end{aligned}
$$

Dan geldt in het algemeen niet langer dat $L(G') \subseteq L(G)$. Waarom niet?

3.38. Eigenschap

Voor elk contextgevoelige grammatica G geldt dat L(G) een beslisbare verzameling is.

Bewijs

Stel $w \in L(G)$ met $|w| = n$. We onderscheiden twee gevallen:

(1) $n = 0$, dat wil zeggen $w = \lambda$. Er geldt $\lambda \in L(G)$ dan en slechts dan als $S \rightarrow \lambda \in P$. Natuurlijk is voor elke contextgevoelige grammatica het probleem $S \rightarrow \lambda \in P$ beslisbaar, omdat P een eindige verzameling is.

(2) $n \geq 1$. Dan bestaat er een afleiding $S = \alpha_0 \Rightarrow \alpha_1 \Rightarrow \cdots \Rightarrow \alpha_r = w$, zodanig dat

(i) $\alpha_i = \alpha_j$ dan en slechts dan als $i = j$. Immers als er een $i < j$ is zodanig dat $\alpha_i = \alpha_j$, dan is ook $S = \alpha_0 \Rightarrow \cdots \Rightarrow \alpha_i \Rightarrow \alpha_{j+1} \Rightarrow \cdots \Rightarrow \alpha_r = w$ een afleiding van w.

(ii) $|\alpha_i| \leq |\alpha_{i+1}|$, $0 \leq i < r$. Voor elke produktieregel $\alpha \rightarrow \beta \in P$ geldt immers $|\alpha| \leq |\beta|$.

De lengte van de afleiding van $w \neq \lambda$ is dus ten hoogste gelijk aan het aantal verschillende woorden over V met een lengte kleiner dan of gelijk aan $|w|$:

$$r \leq \sum_{i=1}^{|w|} (\text{card}(V))^i.$$

Bepaal nu de verzameling

$$W = \{w \mid \exists k \leq \sum_{i=0}^{|w|} (\text{card}(V))^i \; [S \overset{k}{\Rightarrow} w]\}.$$

Dan geldt $w \in W$ dan en slechts dan als $w \in L(G)$. Omdat W een eindige verzameling is, is W beslisbaar. Omdat het, gegeven een woord $w \in \Sigma^*$ mogelijk is de verzameling W daadwerkelijk te construeren, is L(G) beslisbaar.

Einde bewijs

3.39. Eigenschap

$\mathcal{L}_1$ is een echte deelverzameling van de klasse van beslisbare verzamelingen.

Bewijs

Zij Σ een alfabet en ϕ: $N \hookrightarrow\!\!\!\!\rightarrow \Sigma^*$ een opsomming van Σ^* (zie voorbeeld 2.8.). Zij h: $N \hookrightarrow\!\!\!\!\rightarrow \mathcal{L}_{1,\Sigma}$ een opsomming van de contextgevoelige talen over het eindalfabet Σ (zie het bewijs van eigenschap 3.2.). Definieer nu $L = \{\phi(i) \mid \phi(i) \notin h(i)\}$. Aangetoond wordt dat L beslisbaar en niet contextgevoelig is.

(1) L is beslisbaar. Zij $w \in \Sigma^*$. Gebruikmakend van de opsomming van Σ^*, kan er een natuurlijk getal i bepaald worden, zodanig dat $\phi(i) = w$. Gebruikmakend van de opsomming van $\mathcal{L}_{1,\Sigma}$ wordt de met i corresponderende taal h(i) bepaald. Volgens eigenschap 3.38. is

h(i) beslisbaar en kan het beslissingsalgoritme voor $w \in h(i)$ daadwerkelijk worden geconstrueerd. Hieruit volgt dat voor w bepaald kan worden of $w \in L$ dan wel $w \notin L$. De verzameling L is derhalve beslisbaar.

(2) L is niet contextgevoelig. Neem aan dat L contextgevoelig is, dan is er een natuurlijk getal i_0 zodanig dat $h(i_0) = L$. Beschouw nu het woord $\phi(i_0)$:

- $\phi(i_0) \in L$ dan en slechts dan als $\phi(i_0) \notin h(i_0) = L$. Dus $\phi(i_0) \notin L$.
- $\phi(i_0) \notin L$ dan en slechts dan als niet $\phi(i_0) \notin h(i_0)$, dan en slechts dan als $\phi(i_0) \in h(i_0) = L$. Dus $\phi(i_0) \in L$.

Hieruit volgt dat $\phi(i_0) \in L$ dan en slechts dan als $\phi(i_0) \notin L$, hetgeen een tegenspraak is. Hieruit mag geconcludeerd worden dat $L \notin \mathcal{L}_{1,\Sigma}$.

Uit (1) en (2) samen volgt hetgeen te bewijzen was.
Einde bewijs

3.40. Eigenschap

$\mathcal{L}_0$ is gelijk aan de klasse van de opsombare verzamelingen.

Bewijs
Het bewijs van deze eigenschap verloopt in twee stappen:

(1) Zij G = (V,Σ,P,S) een type 0 grammatica. Het volgende algoritmeschema definieert een berekenbare partiële functie waarvan het definitiegebied met L(G) samenvalt:

```
invoer : w ∈ Σ*, G
uitvoer: u ∈ N
begin
   W := {S}
   while w ∉ W do
      W := {z | ∃y ∈ W[y ⇒G z]};
   u := 1;
end
```

Hieruit volgt dat L(G) opsombaar is.

(2) Het bewijs dat elke opsombare verzameling door een type 0 grammatica kan worden voortgebracht, valt buiten het bestek van dit dictaat.

Einde bewijs

3.41. Eigenschap

$\mathcal{L}_1 \subset \mathcal{L}_0$

Bewijs

Deze eigenschap volgt direct uit de eigenschappen 2.31., 3.39. en 3.40.

Einde bewijs

De essentie van type 0 grammatica's is derhalve de mogelijkheid gebruik te maken van verkortende regels.

Dit hoofdstuk zal worden afgesloten met een onderzoek naar de algoritmische oplosbaarheid van een aantal problemen uit de theorie van de formele talen.

3.42. Eigenschap

In het onderstaande overzicht wordt met "ja" aangegeven dat het betreffende probleem algoritmisch oplosbaar is en met "nee" dat het betreffende probleem algoritmisch onoplosbaar is. Definieer $\mathcal{G}_i$ als de klasse van alle type i grammatica's.

probleem \ i	0	1	2	3
$[G \in \mathcal{G}_i,\ w \in \Sigma^*: w \in L(G)]$	nee	ja	ja	ja
$[G \in \mathcal{G}_i: L(G) = \emptyset]$	nee	nee	ja	ja
$[G_1, G_2 \in \mathcal{G}_i: L(G_1) \cap L(G_2) = \emptyset]$	nee	nee	nee	ja

Bewijs

Ga na dat als een probleem algoritmisch oplosbaar is voor $\mathcal{G}_i$, het equivalente probleem ook algoritmisch oplosbaar is voor $\mathcal{G}_j$, $j > i$, $i,j \in \{0,1,2,3\}$; ga na dat als een probleem algoritmisch onoplosbaar is voor $\mathcal{G}_i$, het equivalente probleem ook algoritmisch onoplosbaar is voor $\mathcal{G}_j$, $j < i$; $i,j \in \{0,1,2,3\}$.

(1) De algoritmische oplosbaarheid van het probleem $[G \in \mathcal{G}_1,\ w \in \Sigma^*: w \in L(G)]$ volgt uit eigenschap 3.38. Met het bovenstaande volgt verder de algoritmische oplosbaarheid van de problemen $[G \in \mathcal{G}_2,\ w \in \Sigma^*: w \in L(G)]$ en $[G \in \mathcal{G}_3,\ w \in \Sigma^*: w \in L(G)]$.

(2) De algoritmische onoplosbaarheid van het probleem $[G \in \mathcal{G}_0,\ w \in \Sigma^*: w \in L(G)]$ volgt uit eigenschap 3.40.

(3) De algoritmische oplosbaarheid van het probleem $[G \in \mathcal{G}_2;\ L(G) = \emptyset]$ wordt als volgt aangetoond:

Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica. Zij m het maximum van de lengten van de rechterleden van de produktieregels uit P. Kies $p = m^{\text{card}(V \setminus \Sigma)+1}$. Dan geldt $L(G) \neq \emptyset$ dan en slechts dan als $\exists w \in L(G)$ zodanig dat $|w| \leq p$:

- als $\exists w \in L(G)$ met $|w| \leq p$, dan $L(G) \neq \emptyset$.
- stel $L(G) \neq \emptyset$, dan bevat $L(G)$ een woord w met een minimale lengte $|w| = n$, $n \leq p$. Als immers $n > p$, dan kan w volgens eigenschap 3.33. geschreven worden als $w = xuyvz$, zodanig dat $|uy| \geq 1$, $|uvy| \leq p$ en $\forall i \geq 0\ [xu^ivy^iz \in L(G)]$. Maar dan is $xuz \in L(G)$, met $|xuz| < |w|$. Dit is in tegenspraak met het feit dat w de minimale lengte heeft.

Dit betekent dat voor slechts een eindig aantal woorden w, ten hoogste $\sum_{i=0}^{p} (\text{card}(\Sigma))^i$, bepaald hoeft te worden of $w \in L(G)$. Aangezien $L(G)$ beslisbaar is (zie tabel), is $[G \in \mathcal{G}_2: L(G) = \emptyset]$ algoritmisch oplosbaar. Hieruit volgt tevens de algoritmische oplosbaarheid van $[G \in \mathcal{G}_3: L(G) = \emptyset]$.

(4) Toon zelf de algoritmische onoplosbaarheid van $[G \in \mathcal{G}_0: L(G) = \emptyset]$ en $[G \in \mathcal{G}_1: L(G) = \emptyset]$ aan.

(5) Toon zelf de algoritmische oplosbaarheid van $[G_1,G_2 \in \mathcal{G}_3: L(G_1) \cap L(G_2) = \emptyset]$ aan.

(6) De algoritmische onoplosbaarheid van $[G_1,G_2 \in \mathcal{G}_2: L(G_1) \cap L(G_2) = \emptyset]$ wordt met reductie naar het PCP aangetoond:

Zij Σ een alfabet en $PCP(n,a_1,\ldots,a_n,b_1,\ldots,b_n)$, $n \geq 1$, $a_1,\ldots,a_n$, $b_1,\ldots,b_n \in \Sigma^+$ een exemplaar van het PCP. Beschouw de grammatica $G_1 = (\Sigma \cup \{S_1,x_1,\ldots,x_n\}, \Sigma \cup \{x_1,\ldots,x_n\}, P_1, S_1)$, waarin P_1 de volgende produktieregels bevat:

$$S_1 \rightarrow a_i S_1 x_i, \quad i \leq n$$
$$S_1 \rightarrow a_i x_i, \quad i \leq n$$

en de grammatica $G_2 = (\Sigma \cup \{S_2,x_1,\ldots,x_n\}, \Sigma \cup \{x_1,\ldots,x_n\}, P_2, S_2)$, waarin P_2 de volgende produktieregels bevat:

$$S_2 \rightarrow b_i S_2 x_i, \quad i \leq n$$
$$S_2 \rightarrow b_i x_i, \quad i \leq n$$

Dan is $L(G_1) \cap L(G_2) = \emptyset$ dan en slechts dan als het exemplaar van het PCP de uitkomst onwaar oplevert (Ga dit na).
Als $[G_1,G_2 \in \mathcal{G}_2: L(G_1) \cap L(G_2) = \emptyset]$ algoritmisch oplosbaar zou zijn, dan is daarmee ook het PCP algoritmisch oplosbaar aangezien de grammatica's G_1 en G_2 daadwerkelijk geconstrueerd kunnen worden. Aangezien het PCP algoritmisch onoplosbaar is, mag geconcludeerd worden dat $[G_1,G_2 \in \mathcal{G}_2: L(G_1) \cap L(G_2) = \emptyset]$ algoritmisch onoplosbaar is. Hieruit volgt tevens de algoritmische onoplosbaarheid van $[G_1,G_2 \in \mathcal{G}_1: L(G_1) \cap L(G_2) = \emptyset]$ en $[G_1,G_2 \in \mathcal{G}_0: L(G_1) \cap L(G_2) = \emptyset]$.

Einde bewijs

4. EINDIGE AUTOMATEN

In hoofdstuk 1 werden de begrippen grammatica en de door een grammatica voortgebrachte taal ingevoerd. Uit de gegeven definities volgt onmiddellijk, dat een grammatica $G = (V,\Sigma,P,S)$ een eindig wiskundig object is (want V, Σ en P zijn eindige verzamelingen), waarmee men een aftelbare (d.w.z. een eindige of aftelbaar oneindige) verzameling, namelijk L(G), kan specificeren. Zo kan men bijvoorbeeld met behulp van een grammatica voor de programmeertaal Pascal (zie: K. Jensen & N. Wirth: "PASCAL - User Manual and Report" (1978) Second Edition, Springer) op eindige wijze de aftelbaar oneindige verzameling van alle syntactisch correcte Pascal-programma's vastleggen.

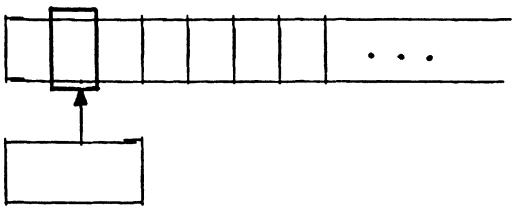
Is eenmaal een programmeertaal, zoals Pascal, door een grammatica gedefinieerd, dan komt het volgende probleem veelvuldig naar voren: bepaal voor een gegeven tekst of deze al dan niet een syntactisch correct Pascal-programma is. Dit is een voorbeeld van het meer algemene probleem: gegeven is een woord en gevraagd wordt of dit woord tot de taal behoort. In het geval van een programmeertaal behoort het tot de taken van een vertaler ("compiler") om deze vraag op efficiënte wijze te beantwoorden.

Merk tevens op, dat als er een methode bestaat om, voor elke x in Σ^* te bepalen of x tot een taal L over Σ behoort, daarmee L gedefinieerd is. Een automaat is een eindig wiskundig object, dat op een dergelijke wijze een taal definieert. Talen kunnen dus gedefinieerd worden door onder andere grammatica's en automaten: bij verschillende soorten grammatica's behoren verschillende soorten automaten en omgekeerd.

In dit hoofdstuk wordt de eindige automaat behandeld, een automaat geschikt om te bepalen of een gegeven woord al dan niet behoort tot de taal, die voortgebracht wordt door een bepaalde type 3 grammatica. De in dit hoofdstuk te behandelen automaat wordt eindig genoemd omdat deze automaat slechts een begrensde hoeveelheid informatie kan onthouden, en dus niet vanwege het feit dat een eindige automaat een eindig wiskundig object is (Alle in dit college ter sprake komende automaten zijn eindige wiskundige objecten, al bezitten sommige soorten automaten het vermogen om een onbegrensde hoeveelheid informatie te onthouden).

Hoewel de eindige automaat een abstract wiskundig begrip is (zie definitie 4.4.) laat het - in tegenstelling tot het begrip grammatica - ook een min of meer concrete fysische interpretatie toe. Deze interpretatie zal samen met enkele voorbeelden besproken worden, waarna de formele definitie van eindige automaat en de daaruit af te leiden eigenschappen volgen.

De fysische interpretatie van een eindige automaat bestaat uit een invoerband verdeeld in cellen, een leeskop, die van links naar rechts beweegt, en een programma:



Aan deze automaat kan een woord ter analyse aangeboden worden door het betreffende woord op de invoerband te schrijven (één letter per cel en opeenvolgende letters in opeenvolgende cellen). De uitvoering van het analyseprogramma, waarin de eisen zijn vastgelegd waaraan een woord moet voldoen om tot de gespecificeerde taal te behoren, dient te leiden tot de acceptatie of de verwerping van het woord op de invoerband al naar gelang dit woord aan de gestelde eisen voldoet.

4.1. Voorbeeld

Een deelprobleem bij de bouw van programmeertaal-vertalers is het herkennen van identifiers: een symboolrij is een identifier als het eerste symbool een letter is en de resterende symbolen letters of cijfers zijn. Het volgende stukje programma is een analyseprogramma voor de bepaling of een gegeven woord behoort tot de taal die uit alle identifiers bestaat. De procedure leessymbool(C) schuift de leeskop één cel naar rechts en kent aan de variabele C de inhoud van de cel waarbij de leeskop staat, toe. De procedure letter(C) levert de waarheidswaarde true op als C een van de letters a,...,z is, en de waarheidswaarde false in alle andere gevallen; de procedure cijfer(C) levert de waarheidswaarde true

op als C een van de cijfers 0,...,9 is en de waarheidswaarde false in alle andere gevallen.

```
toestand0: leessymbool(C);
           if letter(C) then goto toestand1
                        else goto toestand2;

toestand1: leessymbool(C);
           if letter(C) or cijfer(C) then goto toestand1
                                     else goto toestand2;

toestand2:
```

Dit analyseprogramma is geen "net" Pascal-programma. In de eerste plaats wordt er veelvuldig gebruik gemaakt van goto opdrachten en in de tweede plaats is het niet compleet: er dient een omringend programma geschreven te worden dat een actie onderneemt zodra het te analyseren woord geaccepteerd of verworpen is, en dat zorg draagt voor het positioneren van de leeskop en voor het stoppen van het analyseprogramma zodra het laatste symbool op de invoerband gelezen is. Bovenstaand analyseprogramma beschrijft echter de werking van de eindige automaat die geschikt is voor het herkennen van identifiers. •

In de uitvoering van een analyseprogramma worden achtereenvolgens een aantal acties ondernomen die afhangen van de van de invoerband gelezen symbolen. Deze acties kunnen als toestandsveranderingen worden opgevat bij deze eenvoudige deelprogramma's. Eén bepaalde toestand duidt de begintoestand aan, en nul of meer toestanden duiden acceptatie van het woord op de invoerband aan.

4.2. Voorbeeld

Beschouw nogmaals voorbeeld 4.1. Hierin zijn drie toestanden gedefinieerd waarbij een toestand correspondeert met een programma-label

toestand0 "Er is nog geen enkel symbool van de invoerband gelezen"
toestand1 "De van de band gelezen symboolrij is een identifier"
toestand2 "De van de band gelezen symboolrij is geen identifier"

Toestand0 is de begintoestand. Als het programma in toestand1 is, als alle invoer is gelezen, dient de symboolrij op de invoerband geaccepteerd te worden, en als het programma in een van de twee andere toestanden is, als alle invoer is gelezen, dient de symboolrij op de invoerband verworpen te worden. Derhalve leidt alleen toestand1 tot accepteren. •

4.3. Oefening

Schrijf een analyseprogramma voor de herkenning van de reals in Pascal (Bijvoorbeeld -2.38E+21). Geef een verbale beschrijving van elke door u gedefinieerde toestand en geef de begintoestand en de toestanden die tot accepteren leiden aan.

4.4. Definitie: Een deterministische eindige automaat A is een vijftal $A = (Q,\Sigma,\sigma,q_0,F)$, met

(i) Q is een niet-lege eindige verzameling toestanden;
(ii) Σ is een alfabet van invoersymbolen;
(iii) σ: $Q \times \Sigma \rightarrow Q$ is de transitiefunctie;
(iv) $q_0 \in Q$ is de begintoestand;
(v) $F \subseteq Q$ is de verzameling van accepterende toestanden.

Definitie 4.4. geeft het formele model van de besproken deelprogramma's aan. Met $\sigma(q,a) = q'$ wordt het volgende stukje programma weergegeven:

```
q: leessymbool(C);
   if C = a then goto q';
```

Met een deterministische eindige automaat $A = (Q,\Sigma,\sigma,q_0,F)$ wordt als volgt een (gerichte, geëtiketteerde) graaf $G = (K,T)$ geassocieerd, die het transitiediagram van A wordt genoemd. Als verzameling knopen wordt de verzameling toestanden van A gekozen, zodat $K = Q$. Als verzameling takken wordt de verzameling $\{(q_i,q_j) \mid q_i,q_j \in Q \text{ en } \exists a \in \Sigma [q_j = \sigma(q_i,a)]\}$ gekozen: als in A een toestandsverandering van toestand q_i naar toestand q_j kan plaatsvinden onder het invoersymbool $a \in \Sigma$, is er in het transitiediagram van A een tak van q_i naar q_j die het etiket a draagt. De begintoestand wordt in het transitiediagram

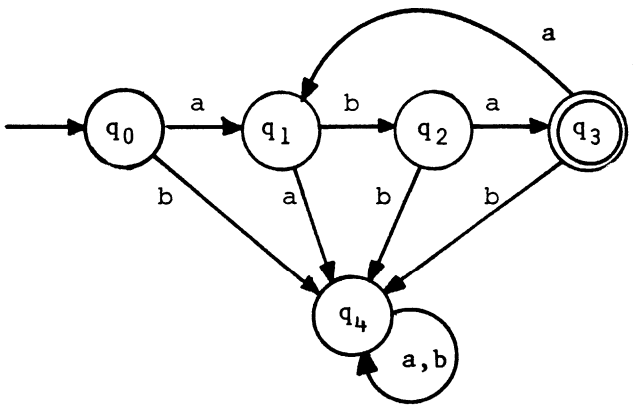
aangegeven met een pijltje en de knopen die met een accepterende toestand corresponderen, worden omcirkeld.

4.5. Voorbeeld

Zij A = $(\{q_0,q_1,q_2,q_3,q_4\}, \{a,b\}, \sigma, q_0, \{q_3\})$. De transitiefunctie σ wordt gegeven in onderstaande tabel:

Q \ Σ	a	b
q_0	q_1	q_4
q_1	q_4	q_2
q_2	q_3	q_4
q_3	q_1	q_4
q_4	q_4	q_4

Bij deze automaat behoort het volgende transitiediagram:



•

4.6. Oefening

Geef de formele beschrijving van en het transitiediagram behorende bij de automaat uit de voorbeelden 4.1. en 4.2.

Het gedrag van $A = (Q,\Sigma,\sigma,q_0,F)$ bestaat uit het herhaald toepassen van de transitiefunctie σ en het verplaatsen van de leeskop. Het zou mogelijk zijn de verplaatsingsrichting van de leeskop in σ op te nemen; bij

eindige automaten is dit echter overbodig, omdat afgesproken is dat de leeskop telkens één plaats naar rechts gaat. Het gedrag van A wordt wiskundig beschreven met het begrip configuratie.

4.7. Definitie: Zij $A = (Q,\Sigma,\sigma,q_0,F)$ een deterministische eindige automaat. Een configuratie van A is een paar $(q,w) \in Q \times \Sigma^*$, waarmee aangegeven wordt dat A in toestand q verkeert en w de resterende invoer is. De leeskop staat dus bij het eerste symbool van w. De verzameling van alle configuraties van A wordt aangegeven met C(A).

4.8. Voorbeeld

Beschouw nogmaals de automaat uit voorbeeld 4.5. Bij de verwerking van de symboolrij abaaba worden achtereenvolgens de volgende configuraties doorlopen:

$(q_0,abaaba)$
$(q_1,baaba)$
$(q_2,aaba)$
(q_3,aba)
(q_1,ba)
(q_2,a)
(q_3,λ) •

4.9. Definitie: Zij $A = (Q,\Sigma,\sigma,q_0,F)$ een deterministische eindige automaat. Een stap is een overgang van configuratie $c_1 \in C(A)$ naar configuratie $c_2 \in C(A)$, die door de transitiefunctie σ is toegestaan. Notatie: $c_1 \vdash_A c_2$, of $c_1 \vdash c_2$ als duidelijk is om welke eindige automaat het gaat. Dus $\forall q,q' \in Q\ \forall a \in \Sigma\ \forall w \in \Sigma^*\ [(q,aw) \vdash_A (q',w)$ dan en slechts dan als $q' = \sigma(q,a)]$. (Vergelijk het verschil tussen σ en $\vdash$ met het verschil tussen $\rightarrow$ en $\Rightarrow$ voor grammatica's).

4.10. Definitie: - het doen van nul stappen in een deterministische eindige automaat A wordt genoteerd als $\vdash^0$:
$c \vdash^0 c'$ dan en slechts dan als $c = c'$.
- het doen van k, $k \geq 1$, stappen in A wordt genoteerd als $\vdash^k$:
$c \vdash^k c'$ dan en slechts dan als er configuraties $c = c_0, c_1, c_2, \ldots, c_k = c'$ bestaan, zodanig dat voor $\forall i$, $0 \leq i < k$, geldt $c_i \vdash c_{i+1}$.
- het doen van een willekeurig aantal maar tenminste één stap in A wordt genoteerd als $\vdash^+$:
$c \vdash^+ c'$ dan en slechts dan als er een k, $k \geq 1$, is, zodanig dat $c \vdash^k c'$.
- het doen van een willekeurig aantal stappen in A wordt genoteerd als $\vdash^*$:
$c \vdash^* c'$ dan en slechts dan als $c \vdash^+ c'$ of $c \vdash^0 c'$.

In de literatuur wordt het herhaald toepassen van de transitiefunctie σ dikwijls beschreven met behulp van de uitgebreide transitiefunctie $\hat{\sigma}$, die gedefinieerd is door

$\hat{\sigma}: Q \times \Sigma^* \to Q$ met
$\forall q \in Q \; [\hat{\sigma}(q,\lambda) = q]$
$\forall q \in Q, \forall a \in \Sigma, \forall w \in \Sigma^* \; [\hat{\sigma}(q,aw) = \hat{\sigma}(\sigma(q,a),w)]$

4.11. Oefeningen

(1) Zij $A = (Q,\Sigma,\sigma,q_0,F)$ een deterministische eindige automaat. Toon aan: $\forall q \in Q \; \forall x,y \in \Sigma^* \; [\hat{\sigma}(q,xy) = \hat{\sigma}(\hat{\sigma}(q,x),y)]$

(2) Zij $A = (Q,\Sigma,\sigma,q_0,F)$ een deterministische eindige automaat. Toon aan: $\forall q,q' \in Q \; \forall w \in \Sigma^* \; [(q,w) \vdash^* (q',\lambda)$ dan en slechts dan als $\hat{\sigma}(q,w) = q']$.

4.12. Definitie: Een woord $w \in \Sigma^*$ wordt door de deterministische eindige automaat $A = (Q,\Sigma,\sigma,q_0,F)$ geaccepteerd dan en slechts dan als $\exists q \in F \; [(q_0,w) \vdash^* (q,\lambda)]$.

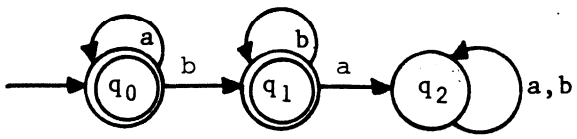
De taal L(A) die door de deterministische eindige automaat $A = (Q,\Sigma,\sigma,q_0,F)$ wordt geaccepteerd, is de verzameling $L(A) = \{w \mid \exists q \in F\ [(q_0,w) \vdash^* (q,\lambda)]\}$.

4.13. Voorbeeld

Zij $A = (Q,\Sigma,\sigma,q_0,F)$ een deterministische eindige automaat, waarin $Q = \{q_0,q_1,q_2\}$, $\Sigma = \{a,b\}$, $F = \{q_0,q_1\}$ en de transitiefunctie σ gedefinieerd is door onderstaande tabel:

Q \ Σ	a	b
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_2	q_2

Bij A behoort het volgende transitiediagram:



Het woord $aaabb \in \Sigma^*$ wordt door A geaccepteerd aangezien

$(q_0,aaabb) \vdash (q_0,aabb) \vdash (q_0,abb) \vdash (q_0,bb) \vdash (q_1,b) \vdash (q_1,\lambda)$ en $q_1 \in F$.

Hieruit volgt $aaabb \in L(A)$. Daarentegen wordt het woord $abbab \in \Sigma^*$ niet door A geaccepteerd daar

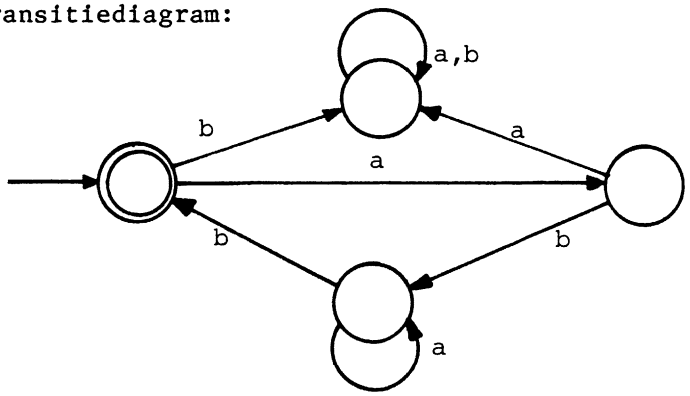
$(q_0,abbab) \vdash (q_0,bbab) \vdash (q_1,bab) \vdash (q_1,ab) \vdash (q_2,b) \vdash (q_2,\lambda)$

en $q_2 \notin F$, zodat $abbab \notin L(A)$. •

4.14. Oefeningen

(1) Beschouw nogmaals de automaat A uit voorbeeld 4.14. Ga na dat $L(A) = \{a^i b^j \mid i \geq 0,\ j \geq 0\}$.

(2) Zij A een deterministische eindige automaat behorend bij onderstaand transitiediagram:



Bepaal L(A).

(3) Construeer een deterministische eindige automaat $A = (Q,\Sigma,\sigma,q_0,F)$ zodanig dat $L(A)=\Sigma^*$. Bepaal het minimale aantal toestanden voor A.

Zij $\mathcal{L}_{DEA}$ de klasse van talen die door deterministische eindige automaten worden geaccepteerd.

4.15. Eigenschap

$\mathcal{L}_{DEA} \subseteq \mathcal{L}_3$.

Bewijs

Zij $A = (Q,\Sigma,\sigma,q_0,F)$ een deterministische eindige automaat. Definieer een reguliere grammatica $G_A = (V,\Sigma,P,S)$ aldus: $V = \Sigma \cup Q$ (zorg dat $Q \cap \Sigma = \emptyset$, eventueel door het herbenoemen van elementen uit Q), $S = q_0$ en $P = \{q \to aq' \mid q,q' \in Q,\ a \in \Sigma \text{ en } \sigma(q,a) = q'\} \cup \{q \to \lambda \mid q \in F\}$. Aangetoond wordt dat $L(G_A) = L(A)$.

Er geldt

$(*)\forall p,q \in Q\ \forall w \in \Sigma^*\ [(p,w) \vdash^* (q,\lambda)$ dan en slechts dan als $p \overset{*}{\Rightarrow} wq]$.

Toon dit zelf aan met inductie naar het aantal afleidingsstappen, respectievelijk het aantal rekenstappen.

(1) Zij $w \in L(A)$ dan $(q_0,w) \vdash^* (q_f,\lambda)$, $q_f \in F$, zodat volgens (*) geldt $q_0 \overset{*}{\Longrightarrow} wq_f$. Aangezien $q_f \in F$, is $q_f \to \lambda \in P$, zodat $q_0 \overset{*}{\underset{G_A}{\Longrightarrow}} wq_f \underset{G_A}{\Longrightarrow} w$ en $w \in L(G_A)$.

(2) Zij $w \in L(G_A)$ dan $q_0 \overset{*}{\underset{G_A}{\Longrightarrow}} wq_f \underset{G_A}{\Longrightarrow} w$. Aangezien $q_f \to \lambda \in P$, is $q_f \in F$, zodat $(q_0,w) \vdash^* (q_f,\lambda)$ en $w \in L(A)$.

Uit (1) en (2) volgt $L(G_A) = L(A)$.
Einde bewijs

4.16. Voorbeeld

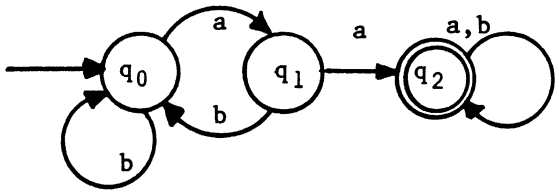
Beschouw nogmaals de automaat van voorbeeld 4.13. Toepassen van bovengenoemde constructie op deze automaat levert een grammatica $G = (V,\Sigma,P,S)$ met $\Sigma = \{a,b\}$, $V = \Sigma \cup \{q_0,q_1,q_2\}$, $S = q_0$ terwijl P de volgende produktieregels bevat.

1. $q_0 \to \lambda$
2. $q_0 \to aq_0$
3. $q_0 \to bq_1$
4. $q_1 \to \lambda$
5. $q_1 \to bq_1$
6. $q_1 \to aq_2$
7. $q_2 \to aq_2$
8. $q_2 \to bq_2$

Merk op dat men de regels 6, 7 en 8 kan weglaten zonder dat daardoor L(G) verandert. •

4.17. Oefening

Zij $A = (Q,\Sigma,\sigma,q_0,F)$ de deterministische eindige automaat bepaald door het onderstaande transitiediagram:



Bepaal een type 3 grammatica G zodanig dat L(G) = L(A).

In een deterministische eindige automaat A is er bij een gegeven toestand q en een gegeven invoersymbool a precies één toestandsverandering gedefinieerd: de transitiefunctie σ is immers een functie van $Q\times\Sigma$ naar Q. Een niet-deterministische eindige automaat wordt gedefinieerd door bij een gegeven toestand q en een gegeven invoersymbool a een verzameling van mogelijke toestandsveranderingen toe te staan; de transitiefunctie van een niet-deterministische eindige automaat is een functie van $Q \times \Sigma$ naar $P(Q)$, de verzameling van alle deelverzamelingen van Q.

4.18. Definitie: Een niet-deterministische eindige automaat A is een vijftal $A = (Q,\Sigma,\sigma,Q_0,F)$ met

(i) Q is een niet-lege eindige verzameling toestanden
(ii) Σ is een alfabet van invoersymbolen
(iii) σ: $Q\times\Sigma \to P(Q)$ is de transitiefunctie
(iv) $Q_0 \subseteq Q$ is de verzameling begintoestanden
(v) $F \subseteq Q$ is de verzameling accepterende toestanden.

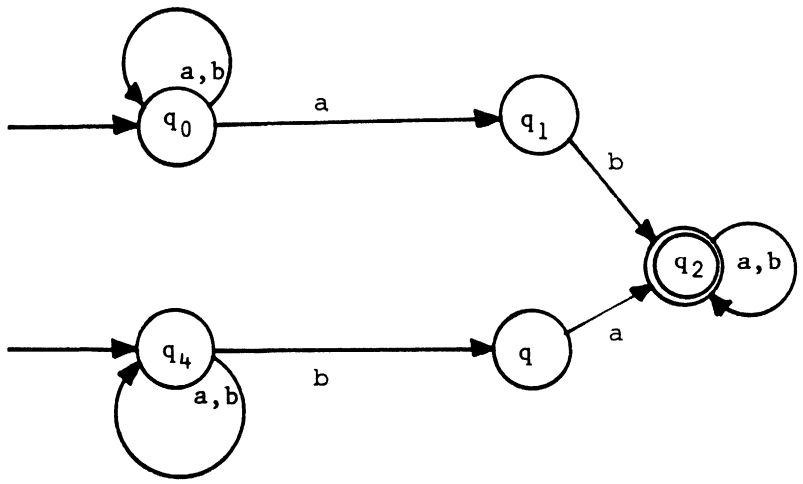
Evenals met elke deterministische eindige automaat wordt met elke niet-deterministische eindige automaat een transitiediagram geassocieerd.

4.19. Voorbeeld

Zij $A = (\{q_0,q_1,q_2,q_3,q_4\}, \{a,b\}, \sigma, \{q_0,q_3\}, \{q_2\})$. De transitiefunctie wordt gegeven in onderstaande tabel:

Q \ Σ	a	b
q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$
q_3	$\{q_3\}$	$\{q_3,q_4\}$
q_4	$\{q_2\}$	$\emptyset$

Bij deze niet-deterministische eindige automaat hoort het volgende transitiediagram:



Het begrip configuratie (zie definitie 5.7.) is onveranderd van toepassing op niet-deterministische automaten. Het begrip stap wordt voor niet-deterministische automaten opnieuw gedefinieerd.

4.20. Definitie: Zij $A = (Q,\Sigma,\sigma,Q_0,F)$ een niet-deterministische eindige automaat. Een stap is een overgang van een configuratie $c_1 \in C(A)$ naar een configuratie $c_2 \in C(A)$, die door de transitiefunctie σ is toegestaan. Notatie: $c_1 \vdash_A c_2$, of $c_1 \vdash c_2$ als duidelijk is om welke eindige automaat het gaat. Dus $\forall q,q' \in Q,\ \forall a \in \Sigma,\ \forall w \in \Sigma^*[(q,aw) \vdash_A (q',w)$ dan en slechts dan als $q' \in \sigma(q,a)]$.

De notaties $\vdash^0$, $\vdash^k$, $\vdash^*$ en $\vdash^+$ (zie definitie 4.10.) zijn ook van toepassing op niet-deterministische automaten als men in definitie 4.10. $\vdash$ leest in de betekenis van definitie 4.20. in plaats van 4.9.

4.21. Definitie: Een woord $w \in \Sigma^*$ wordt door de niet-deterministische eindige automaat $A = (Q,\Sigma,\sigma,Q_0,F)$ geaccepteerd dan en slechts dan als $\exists q_0 \in Q_0 \exists q \in F[(q_0,w) \vdash^* (q,\lambda)]$.

De taal L(A) die door de niet-deterministische eindige automaat A wordt geaccepteerd, is de verzameling
$L(A) = \{w \mid \exists q_0 \in Q_0 \exists q \in F[(q_0,w) \vdash^* (q,\lambda)]\}$.

4.22. Voorbeeld

Beschouw nogmaals de niet-deterministische eindige automaat $A = (Q,\Sigma,\sigma,Q_0,F)$ uit voorbeeld 4.19. Aangezien $Q_0 \cap F = \emptyset$ behoort het woord λ niet tot L(A). Het woord aaba wordt wel door A geaccepteerd. Er bestaan voor dit woord twee berekeningen die tot accepteren leiden:

(1) $(q_0,aaba) \vdash (q_0,aba) \vdash (q_1,ba) \vdash (q_2,a) \vdash (q_2,\lambda)$
(2) $(q_4,aaba) \vdash (q_4,aba) \vdash (q_4,ba) \vdash (q_3,a) \vdash (q_2,\lambda)$

Er bestaan ook berekeningen die onder de invoer aaba niet tot accepteren leiden; bijvoorbeeld

$$(q_0,aaba) \vdash (q_0,aba) \vdash (q_0,ba) \vdash (q_0,a) \vdash (q_0,\lambda)$$

Desalniettemin wordt aaba door A geaccepteerd en geldt: aaba $\in$ L(A). Beschouw vervolgens het woord aa. Voor dit woord bestaan er vier berekeningen, namelijk

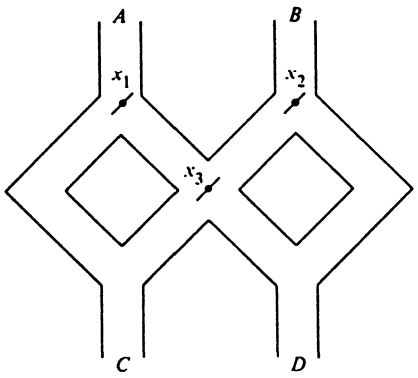
(1) $(q_0,aa) \vdash (q_1,a)$
(2) $(q_0,aa) \vdash (q_0,a) \vdash (q_0,\lambda)$
(3) $(q_0,aa) \vdash (q_0,a) \vdash (q_1,\lambda)$
(4) $(q_4,aa) \vdash (q_4,a) \vdash (q_4,\lambda)$

Geen enkele berekening voor aa leidt tot acceptatie, want de berekening wordt of geblokkeerd (1) of eindigt in een toestand, die niet accepterend is (2,3,4). Dus geldt aa $\notin$ L(A).

Ga na dat een berekening in een automaat correspondeert met een pad in het bij die automaat behorende transitiediagram, en omgekeerd. •

4.23. Oefening

Beschouw het spelletje uit de volgende figuur:



In de ingangen A en B kunnen knikkers geworpen worden. Met x_1, x_2 en x_3 worden scharnieren aangeduid die in één van twee toestanden verkeren en die ervoor zorgen dat een knikker naar links of naar rechts valt. De begintoestanden van deze scharnieren zijn in de figuur aangegeven. Als langs een scharnier een knikker naar rechts (links) is gevallen, neemt deze scharnier de andere toestand aan, zodat de volgende knikker langs de scharnier naar links (rechts) valt. Een ingeworpen knikker verlaat via uitgang C of via uitgang D het spel. Een reeks ingeworpen knikkers wordt geaccepteerd als de laatst ingeworpen knikker het spel via uitgang D verlaat. Ontwerp een niet-deterministische eindige automaat voor dit spelletje.

In de eigenschappen 4.25. en 4.26. wordt aangetoond dat de klasse van talen die door deterministische eindige automaten worden geaccepteerd gelijk is aan de klasse van talen die door niet-deterministische eindige automaten worden geaccepteerd.

4.24. Definitie: Als A_1 en A_2 eindige automaten zijn, dat heten A_1 en A_2 equivalent als $L(A_1) = L(A_2)$.

4.25. Eigenschap

Bij elke deterministische eindige automaat kan een equivalente niet-deterministische eindige automaat geconstrueerd worden.

Bewijs

Zij $A_D = (Q,\Sigma,\sigma,q_0,F)$ een deterministische eindige automaat. Construeer een niet-deterministische eindige automaat $A_N = (Q,\Sigma,\sigma_N,\{q_0\},F)$, waarin σ_N: $Q \times \Sigma \to P(Q)$ met

$$\sigma_N(q,a) = \{\sigma(q,a)\} \text{ voor alle } q \in Q \text{ en alle } a \in \Sigma.$$

Toon aan dat $L(A_D) = L(A_N)$.

Einde bewijs

4.26. Eigenschap

Bij elke niet-deterministische eindige automaat kan een equivalente deterministische eindige automaat geconstrueerd worden.

Bewijs

Zij $A_N = (Q,\Sigma,\sigma,Q_0,F)$ een niet-deterministische eindige automaat. Construeer een deterministische eindige automaat $A_D = (P(Q),\Sigma,\sigma_D,Q_0,F_D)$, waarin σ_D: $P(Q) \times \Sigma \to P(Q)$ en F_D als volgt worden gedefinieerd

$$\sigma_D(X,a) = \cup \{\sigma(q,a) \mid q \in X\} \text{ voor alle } X \in P(Q) \text{ en alle } a \in \Sigma,$$
$$F_D = \{X \in P(Q) \mid X \cap F \neq \emptyset\}.$$

Aangetoond wordt dat $L(A_D) = L(A_N)$.

Er geldt
(*) $\forall X,Y \in P(Q) \forall w \in \Sigma^*[(X,w) \overset{*}{\underset{A_D}{\vdash}} (Y,\lambda)$ dan en slechts dan als $Y = \{q \mid \exists p \in X[(p,w) \overset{*}{\underset{A_N}{\vdash}} (q,\lambda)]\}]$. Toon dit zelf aan met inductie naar het aantal rekenstappen van A_N respectievelijk A_D.

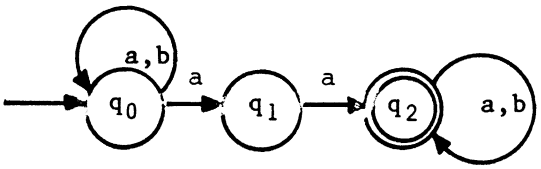
(1) Zij $w \in L(A_N)$. Dan $\exists p \in Q_0 \, \exists q \in F[(p,w) \overset{*}{\underset{A_N}{\vdash}} (q,\lambda)]$. Volgens (*) volgt dan uit $(Q_0,w) \overset{*}{\underset{A_D}{\vdash}} (Y,\lambda)$ dat $Y \cap F \neq \emptyset$ zodat $Y \in F_D$ en $w \in L(A_D)$.

(2) Zij $w \in L(A_D)$. Dan $\exists Y \in F_D[(Q_0,w) \overset{*}{\underset{A_D}{\vdash}} (Y,\lambda)]$. Dan is $Y \cap F \neq \emptyset$ en is er dus een $q \in Y \cap F$. Volgens (*) is er dan ook een $p \in Q_0$ zodanig dat $(p,w) \overset{*}{\underset{A_D}{\vdash}} (q,\lambda)$ zodat $w \in L(A_N)$.

Uit (1) en (2) volgt $L(A_N) = L(A_D)$.
Einde bewijs

4.27. Voorbeeld

$A_N = (Q,\Sigma,\sigma,Q_0,F)$ is de niet-deterministische eindige automaat met onderstaand transitiediagram



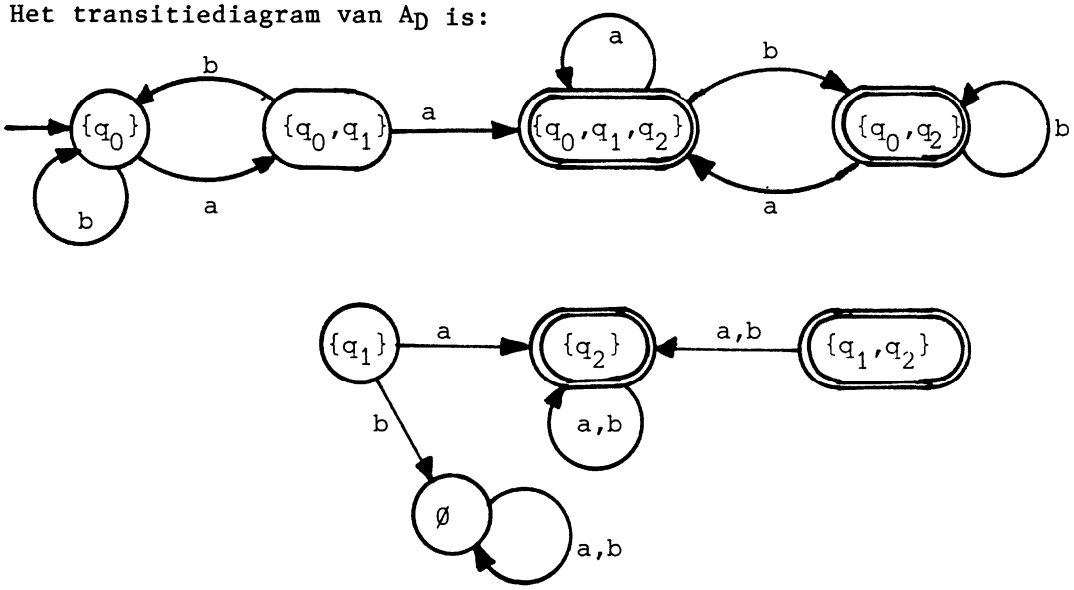
In onderstaande tabel wordt σ weergegeven:

σ	a	b
q_0	$\{q_0,q_1\}$	$\{q_0\}$
q_1	$\{q_2\}$	$\emptyset$
q_2	$\{q_2\}$	$\{q_2\}$

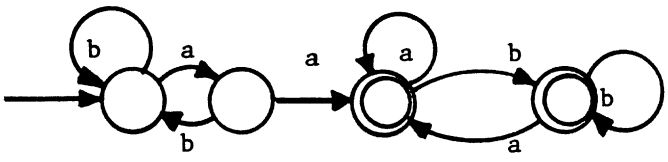
Er wordt een met A_N equivalente deterministische eindige automaat A_D geconstrueerd: $A_D = (P(Q), \Sigma, \sigma_D, \{q_0\}, \{\{q_0,q_1,q_2\}, \{q_0,q_2\}, \{q_1,q_2\}, \{q_2\}\})$. De transitiefunctie σ_D wordt in onderstaande tabel weergegeven:

σ_D	a	b
$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0\}$	$\{q_0,q_1\}$	$\{q_0\}$
$\{q_1\}$	$\{q_2\}$	$\emptyset$
$\{q_2\}$	$\{q_2\}$	$\{q_2\}$
$\{q_0,q_1\}$	$\{q_0,q_1,q_2\}$	$\{q_0\}$
$\{q_0,q_2\}$	$\{q_0,q_1,q_2\}$	$\{q_0,q_2\}$
$\{q_1,q_2\}$	$\{q_2\}$	$\{q_2\}$
$\{q_0,q_1,q_2\}$	$\{q_0,q_1,q_2\}$	$\{q_0,q_2\}$

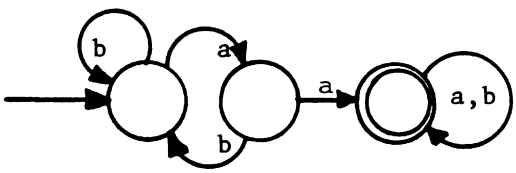
Het transitiediagram van A_D is:



Met A_D equivalente deterministische eindige automaten zijn bijvoorbeeld

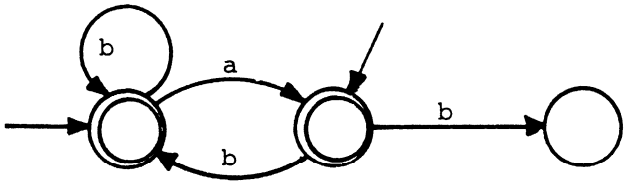


en ook



4.28. Oefening

Beschouw de niet-deterministische eindige automaat A met onderstaand transitiediagram.



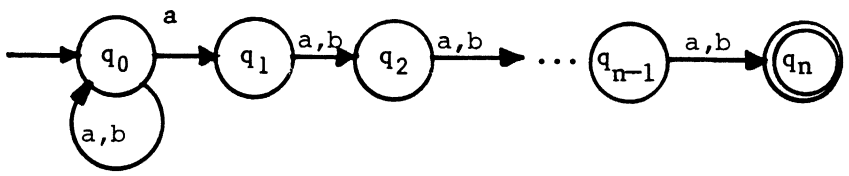
Construeer een met A equivalente deterministische eindige automaat volgens de constructie uit het bewijs van eigenschap 4.26.

Als $A_N = (Q,\Sigma,\sigma,Q_0,F)$ een niet-deterministische eindige automaat is, resulteert de constructie uit het bewijs van eigenschap 4.27. in een deterministische eindige automaat met card(P(Q)) toestanden. Vaak is het mogelijk met A equivalente deterministische eindige automaten te construeren met veel minder dan card(P(Q)) toestanden, zoals in voorbeeld 4.27. is gedaan. In voorbeeld 4.29. wordt aangetoond dat dit niet altijd mogelijk is.

4.29. Voorbeeld

Voor elk natuurlijk getal n bestaat er een taal $L_n \subseteq \{a,b\}^*$, zodanig dat er een niet-deterministische eindige automaat A_N met n+1 toestanden bestaat die L_n accepteert en dat elke deterministische eindige automaat die L_n accepteert tenminste 2^n toestanden heeft.

Definieer $L_n = \{wav \mid w,v \in \{a,b\}^* \text{ en } |av| = n\}$. De niet-deterministische eindige automaat A_N met onderstaand transitiediagram heeft n+1 toestanden en accepteert L_n:



Zij $A_D = (Q,\{a,b\},\sigma,q_0,F)$ een deterministische eindige automaat die L_n accepteert. Dan heeft A_D tenminste 2^n toestanden immers:

Zij v_1 en v_2 willekeurige woorden uit $\{a,b\}^*$ zodanig dat $v_1 \neq v_2$, en $|v_1| = |v_2| = n$. Dan is er een woord u zodanig dat $v_1u \in L$ en $v_2u \notin L$. Immers als $v_1 \neq v_2$, dan zijn er woorden w_1, w_2 en w_2' uit $\{a,b\}^*$ zodanig dat $v_1 = w_1aw_2$ en $v_2 = w_1bw_2'$ (eventueel na herbenoemen van v_1 en v_2). Dus v_1 en v_2 kunnen zodanig worden uitgebreid met een $u \in \{a,b\}^*$ dat $|aw_2u| = |bw_2'u| = n$. Volgens de definitie van L geldt dan $v_1u \in L$ en $v_2u \notin L$.

Stel nu dat $(q_0,v_1) \vdash^* (q_1,\lambda)$ en $(q_0,v_2) \vdash^* (q_2,\lambda)$ en $q_1 = q_2$. Dan geldt $(q_0,v_1u) \vdash^* (q_1,u) \vdash^* (q,\lambda)$ voor zekere $q \in F$, en $(q_0,v_2u) \vdash^* (q_2,u) = (q_1,u) \vdash^* (q',\lambda)$. Omdat A_D deterministisch is, geldt $q' = q$, zodat $v_1u \in L$ dan en slechts dan als $v_2u \in L$. Dit is in tegenspraak met $v_1u \in L$ en $v_2u \notin L$.

Het aantal toestanden van A_D is derhalve groter dan of gelijk aan het aantal verschillende woorden over $\{a,b\}$ ter lengte n, dat wil zeggen $\text{card}(Q) \geq 2^n$.

•

4.30. Eigenschap

$\mathcal{L}_3 \subseteq \mathcal{L}_{DEA}$.

Bewijs

Zij $G = (V,\Sigma,P,S)$ een type 3 grammatica. Construeer eerst een niet-deterministische eindige automaat $A = (Q,\Sigma,\sigma,Q_0,F)$, waarin

$Q = V \setminus \Sigma$

$Q_0 = \{S\}$

$F = \{A \mid A \in V \setminus \Sigma \text{ en } A \to \lambda \in P\}$

$\sigma(A,a) = \{B \mid A \to aB \in P\}$ voor alle $A \in Q$ en alle $a \in \Sigma$.

Aangetoond wordt dat $L(A) = L(G)$.

Er geldt

(*) $\forall A,B \in V \setminus \Sigma \forall w \in \Sigma^* [A \overset{*}{\Longrightarrow} wB$ dan en slechts dan als $(A,w) \vdash^* (B,\lambda)]$.

Toon dit zelf aan met inductie naar het aantal afleidingsstappen, respectievelijk rekenstappen.

(1) Zij $w \in L(G)$. Dan $\exists B \in V \setminus \Sigma [S \overset{*}{\Rightarrow} wB \Rightarrow w]$ en $B \to \lambda \in P$. Dan geldt volgens (*) dat $(S,w) \overset{*}{\vdash} (B,\lambda)$ en $B \in F$, zodat $w \in L(A)$.

(2) Zij $w \in L(A)$. Dan $\exists B \in F[(S,w) \overset{*}{\vdash} (B,\lambda)]$. Uit (*) en uit het feit dat $B \to \lambda \in P$, volgt $S \overset{*}{\Rightarrow} wB \Rightarrow w$, zodat $w \in L(G)$.

Uit (1) en (2) volgt $L(A) = L(G)$. Met behulp van 4.26. volgt tenslotte de bewering.

Einde bewijs

Uit de eigenschappen 4.15. en 4.30. volgt $\mathcal{L}_3 = \mathcal{L}_{DEA}$.

4.31. Oefeningen

(1) Zij $G = (\{S,A,a,b\}, \{a,b\}, P, S)$ waarin $P = \{S \to bS, S \to aA, S \to \lambda, A \to bS, A \to \lambda\}$. Construeer een deterministische eindige automaat A, zodanig dat $L(A) = L(G)$.

(2) Ontwerp een deterministische eindige automaat die de taal $L = \{w \in \{a,b\}^* \mid w = w_1 babw_2,\ w_1,w_2 \in \{a,b\}^*\}$ accepteert.

(3) Zij A een eindige automaat met n toestanden. Toon aan dat L(A) dan en alleen dan niet leeg is als A een woord w met $|w| < n$ accepteert.

Dit hoofdstuk wordt besloten met de behandeling van het begrip transitiesysteem. Een transitiesysteem kan gezien worden als een eindige automaat waarin het mogelijk is in plaats van telkens één symbool van de invoerband te lezen en te verwerken, symboolrijen te lezen en te verwerken.

4.32. Definitie: Een transitiesysteem A is een vijftal $A = (Q,\Sigma,\sigma_T,Q_0,F)$ waarin Q, Σ, Q_0 en F als bij een niet-deterministische eindige automaat gedefinieerd zijn en de transitiefunctie σ_T een functie van een eindige deelverzameling van $Q \times \Sigma^*$ naar P(A) is.

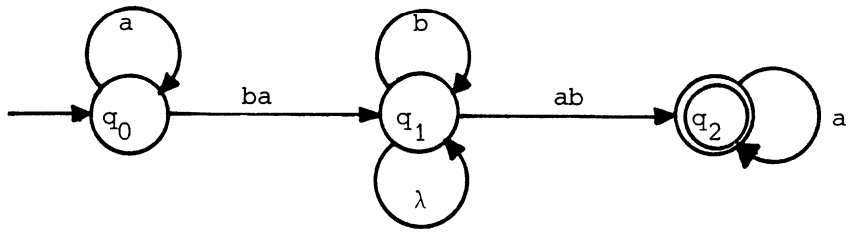
Evenals met elke deterministische en elke niet-deterministische eindige automaat wordt met elk transitiesysteem een transitiediagram geassocieerd. Elke tak in dit transitiediagram draagt als etiket een woord over het alfabet Σ.

4.33. Voorbeeld

Zij $A = (\{q_0,q_1,q_2\}, \{a,b\}, \sigma_T, \{q_0\}, \{q_2\})$. De transitiefunctie σ_T wordt gegeven in onderstaande tabel:

$\sigma_T(q_0,a) = \{q_0\}$
$\sigma_T(q_0,ba) = \{q_1\}$
$\sigma_T(q_1,b) = \{q_1\}$
$\sigma_T(q_1,ab) = \{q_2\}$
$\sigma_T(q_1,\lambda) = \{q_1\}$
$\sigma_T(q_2,a) = \{q_2\}$

Bij A hoort het volgende transitiediagram:



•

Het begrip configuratie (zie definitie 4.7.) is onveranderd van toepassing op transitiesystemen. Het begrip stap wordt voor transitiesystemen opnieuw gedefinieerd.

4.34. Definitie: Zij $A = (Q,\Sigma,\sigma_T,Q_0,F)$ een transitiesysteem. Een stap is een overgang van een configuratie $c_1 \in C(A)$ naar een configuratie $c_2 \in C(A)$, die door de transitiefunctie σ is toegestaan. Notatie: $c_1 \vdash_A c_2$, of $c_1 \vdash c_2$ als duidelijk is om welk transitiesysteem het gaat. Dus $\forall q,q' \in Q\, \forall v,w \in \Sigma^*$ $[(q,vw) \vdash_A (q',w)$ dan en slechts dan als $q' \in \sigma_T(q,v)]$.

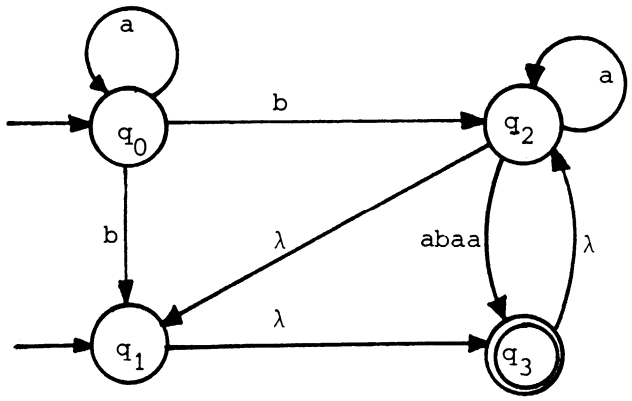
Uitgaande van definitie 4.34. zijn de notaties $\vdash^0$, $\vdash^k$, $\vdash^*$ en $\vdash^+$ (zie definitie 4.10.) ook van toepassing op transitiesystemen.

4.35. Definitie: Een woord $w \in \Sigma^*$ wordt door een transitiesysteem $A = (Q,\Sigma,\sigma_T,Q_0,F)$ geaccepteerd dan en slechts dan als $\exists q_0 \in Q_0\, \exists q \in F[(q_0,w) \vdash^* (q,\lambda)]$.

De taal L(A) die door het transitiesysteem A wordt geaccepteerd, is de verzameling $L(A) = \{w \mid \exists q_0 \in Q_0 \exists q \in F [(q_0,w) \overset{*}{\vdash} (q,\lambda)]\}$.

4.36. Voorbeeld

Beschouw het transitiesysteem $A = (Q,\Sigma,\sigma_T,Q_0,F)$ met onderstaand transitiediagram:



$Q_0 = \{q_0,q_1\}$
$F = \{q_3\}$

Het woord ababaa wordt door A geaccepteerd. Er bestaan oneindig veel berekeningen die tot acceptatie van dit woord leiden. Drie mogelijke berekeningen zijn:

(1) $(q_0,ababaa) \vdash (q_0,babaa) \vdash (q_2,abaa) \vdash (q_3,\lambda)$,

(2) $(q_0,ababaa) \vdash (q_0,babaa) \vdash (q_2,abaa) \vdash (q_1,abaa) \vdash (q_3,abaa) \vdash (q_2,abaa) \vdash (q_3,\lambda)$.

(3) $(q_0,ababaa) \vdash (q_0,babaa) \vdash (q_1,abaa) \vdash (q_3,abaa) \vdash (q_2,abaa) \vdash (q_3,\lambda)$. ●

Aangezien elke niet-deterministische eindige automaat een transitiesysteem is, is $\mathcal{L}_3$ bevat in de klasse van talen die door transitiesystemen geaccepteerd worden. In de volgende eigenschap komt tot uitdrukking dat transitiesystemen niet "machtiger" zijn dan eindige automaten.

4.37. Eigenschap

De klasse van talen die door transitiesystemen geaccepteerd worden, is gelijk aan $\mathcal{L}_3$.

Toon de eigenschap zelf aan (Aanwijzing: breid de constructie uit het bewijs van eigenschap 4.15. uit voor rechts-lineaire grammatica's en transitiesystemen en breid op analoge wijze de constructie uit het bewijs van eigenschap 4.30. uit).

4.38. Oefening

Construeer een transitiesysteem A zodanig dat L(A) = L(G), waarin G = ($\{S,A,B,a,b\}$, $\{a,b\}$, P, S) waarin P = $\{S \to abS,\ S \to abA,\ A \to B,\ B \to baB,\ B \to ba\}$. Beschrijf L(A).

5. REGULIERE EXPRESSIES

In de vorige twee hoofdstukken werden verschillende gelijkwaardige definities van de klasse $\mathcal{L}_3$ van reguliere talen gegeven, namelijk in termen van type 3 grammatica's, linkslineaire grammatica's, rechtslineaire grammatica's, deterministische eindige automaten, niet-deterministische eindige automaten en transitiesystemen. Dit hoofdstuk is gewijd aan de karakterisering van $\mathcal{L}_3$ door middel van zogenaamde reguliere expressies. Daartoe wordt er eerst enige aandacht aan een aantal operaties op talen besteed.

5.1. Definitie: Zij $L_1 \subseteq \Sigma_1^*$ en $L_2 \subseteq \Sigma_2^*$. De vereniging $L_1 \cup L_2$ van L_1 en L_2 is de taal over $\Sigma_1 \cup \Sigma_2$ gedefinieerd door:

$$L_1 \cup L_2 = \{x \mid x \in L_1 \text{ of } x \in L_2\},$$

en de concatenatie L_1L_2 van L_1 en L_2 is de taal over $\Sigma_1 \cup \Sigma_2$ gedefinieerd door

$$L_1L_2 = \{x_1x_2 \mid x_1 \in L_1 \text{ en } x_2 \in L_2\}.$$

Deze definitie is in het bijzonder van toepassing op het geval $\Sigma_1 = \Sigma_2$, waartoe dit hoofdstuk zich beperkt. In hoofdstuk 1 werd opgemerkt dat voor elk alfabet Σ, Σ^* een monoïde is met concatenatie (van woorden) als operatie en het lege woord λ als neutraal element. Evenzo is het eenvoudig om aan te tonen dat de machtsverzameling $P(\Sigma^*)$ van Σ^*

(1) een monoïde vormt met vereniging als operatie en de lege verzameling $\emptyset$ als neutraal element (Bovendien is deze monoïde commutatief, d.w.z. dat $L_1 \cup L_2 = L_2 \cup L_1$ voor alle L_1 en L_2 uit $P(\Sigma^*)$);

(2) een monoïde vormt met concatenatie van talen als operatie, $\{\lambda\}$ als neutraal element en $\emptyset$ als nulelement (want $\{\lambda\}L = L\{\lambda\} = L$ en respectievelijk $\emptyset L = L\emptyset = \emptyset$ gelden voor alle L in $P(\Sigma^*)$);

(3) de eigenschap bezit dat voor alle L_1, L_2 en L_3 uit $P(\Sigma^*)$ geldt dat $L_1(L_2 \cup L_3) = L_1L_2 \cup L_1L_3$

en $(L^1 \cup L^2)L^3 = L^1L^3 \cup L^2L^3$.

Een structuur bestaande uit twee op deze wijze gekoppelde monoïden waarvan de eerste commutatief is, wordt een halfring genoemd. De tweede hier genoemde monoïde is niet commutatief, zoals blijkt uit het volgende

5.2. Voorbeeld

Zij $\Sigma = \{a,b\}$ en $L_1 = \{a^n b^n a^n \mid n \geq 0\}$, $L_2 = \{a^i \mid i \geq 0\}$. Dan geldt $L_1, L_2 \in P(\Sigma^*)$, terwijl
$L_1L_2 = \{a^n b^n a^{n+i} \mid n,i \geq 0\} = \{a^n b^n a^k \mid 0 \leq n \leq k\}$, en
$L_2L_1 = \{a^{n+i} b^n a^n \mid n,i \geq 0\} = \{a^k b^n a^n \mid 0 \leq n \leq k\}$,
zodat $L_1L_2 \neq L_2L_1$. •

5.3. Definitie: Zij Σ een alfabet en L een taal over Σ.

De 0-iteratie van L, notatie L^0, wordt gedefinieerd door $L^0 = \{\lambda\}$.

Voor elke $k \geq 1$, wordt de k-iteratie van L, notatie L^k, gedefinieerd door $L^k = L^{k-1}L$.

De iteratie van L, notatie L^*, wordt gedefinieerd door $L^* = \bigcup_{i \geq 0} L^i$.

De λ-vrije iteratie van L, notatie L^+, wordt gedefinieerd door $L^+ = \bigcup_{i \geq 1} L^i$.

5.4. Voorbeeld

Als $L_1 = \{(ab)^n \mid n \geq 1\}$, dan is $L_1^+ = L_1$ en $L_1^* = \{(ab)^n \mid n \geq 0\}$. Voor

de taal $L_2 = \{a^n b^n \mid n \geq 1\}$ is $L_2^* = \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \ldots a^{n_k} b^{n_k} \mid k \geq 0;\ n_i \geq 1,\ 1 \leq i \leq k\}$.

Voorts is $\emptyset^+ = \emptyset$, $\emptyset^* = \{\lambda\}$ en $L^* = L^+ \cup \{\lambda\}$ voor alle L. •

5.5. Oefening

Ga na dat voor elke taal L geldt dat $L^+ = L^*$ dan en slechts dan als $\lambda \in L$.

De volgende twee definities (5.6. en 5.9.) spelen de hoofdrol in de theorie van de reguliere expressies.

5.6. Definitie: Zij Σ een alfabet dat de symbolen), (, Λ, $\emptyset$, + en * niet bevat. De verzameling R_Σ van alle reguliere expressies over Σ is als volgt gedefinieerd:

(1) $\emptyset, \Lambda \in R_\Sigma$ en voor alle $a \in \Sigma$ geldt: $a \in R_\Sigma$;

(2) als $\alpha, \beta \in R_\Sigma$, dan ook $(\alpha+\beta)$, $(\alpha\beta)$, $\alpha^* \in R_\Sigma$;

(3) een symboolrij is alleen dan een reguliere expressie over Σ als hij gevormd kan worden door (1) en (2) een eindig aantal malen toe te passen.

5.7. Voorbeeld

Zij $\Sigma = \{a,b\}$. Dan zijn $\emptyset$, Λ, a, b^*, (ba) en $(((a+b)^*(ba))^*(aa))$ reguliere expressies over Σ, maar de symboolrijen ab), $a+^*b$ en $)\Lambda^*b$ behoren niet tot R_Σ. De verzameling R_Σ kan worden voortgebracht door de context-vrije grammatica $G = (V,\Delta,P,S)$ met $\Delta = \Sigma \cup \{),(,\Lambda,\emptyset,+,^*\}$, $V = \{S\} \cup \Delta$ en $P = \{S \to (S+S),\ S \to (SS),\ S \to S^*,\ S \to \emptyset,\ S \to \Lambda\} \cup \{S \to a \mid a \in \Sigma\}$. Ga na dat $L(G) = R_\Sigma$. •

Evenals bij rekenkundige expressies en formules in de logica beschouwt men ook voor reguliere expressies naast de formele schrijfwijze (zoals vastgelegd in definitie 5.6.) een minder formele, waarin men door aan operaties een prioriteit toe te kennen, het totaal aantal haakjes in de expressies in vele gevallen terugdringt, terwijl men ook het buitenste paar haakjes weglaat. Zo krijgt * de hoogste prioriteit, dan het aaneenschrijven en tenslotte +. Dus schrijft men $\alpha+\beta$ in plaats van $(\alpha+\beta)$ maar niet $\alpha\beta^*$ in plaats van $(\alpha\beta)^*$, want $\alpha\beta^*$ wordt geïnterpreteerd als $(\alpha\beta^*)$.

5.8. Voorbeeld

De in voorbeeld 5.7. genoemde reguliere expressies over $\{a,b\}$ schrijft men kortheidshalve respectievelijk als $\emptyset$, Λ, a, b^*, ba en $((a+b)^*ba)^*aa$. Evenzo schrijft men $a+ba^*$ in plaats van $(a+(b(a^*)))$. •

Reguliere expressies over Σ worden ingevoerd om reguliere talen over Σ te karakteriseren (zie eigenschap 5.18. hierbeneden). Daartoe wordt aan iedere reguliere expressie α over Σ een taal over Σ toegevoegd.

5.9. Definitie: Aan reguliere expressies over Σ wordt een betekenis toegekend door de functie $\rho: R_\Sigma \to P(\Sigma^*)$, die inductief gedefinieerd is door

(1) $\rho(\emptyset) = \emptyset$, $\rho(\Lambda) = \{\lambda\}$ en voor alle $a \in \Sigma$ geldt $\rho(a) = \{a\}$;
(2) voor alle α en β in R_Σ geldt, $\rho(\alpha+\beta) = \rho(\alpha) \cup \rho(\beta)$;
(3) voor alle α en β in R_Σ geldt, $\rho(\alpha\beta) = \rho(\alpha)\rho(\beta)$;
(4) voor alle α in R_Σ geldt, $\rho(\alpha^*) = (\rho(\alpha))^*$.

5.10. Voorbeeld

Voor de in voorbeeld 5.8. genoemde reguliere expressies over $\{a,b\}$ geldt respectievelijk

$\rho(\emptyset) = \emptyset$, $\rho(\Lambda) = \{\lambda\}$, $\rho(a) = \{a\}$, $\rho(b^*) = \{b\}^* = \{b^n \mid n \geq 0\}$,
$\rho(((a+b)^*ba)^*aa) = \{x_1bax_2ba\ldots x_nbaaa \mid n \geq 1;\ x_i \in \{a,b\}^*,$
$1 \leq i \leq n\} \cup \{aa\}$,
$\rho(a+ba^*) = \{a\} \cup \{ba^n \mid n \geq 0\}$. •

5.11. Definitie: Twee reguliere expressies α en β over Σ zijn gelijkwaardig (Eng.: "equivalent"), notatie $\alpha \equiv \beta$, als en alleen als $\rho(\alpha) = \rho(\beta)$.

Het spreekt vanzelf, dat als twee reguliere expressies α en β identiek zijn, dat dan ook $\alpha \equiv \beta$ geldt. Omgekeerd is het echter zeer wel mogelijk, dat twee verschillende expressies gelijkwaardig zijn. Beschouw, als zeer eenvoudig voorbeeld: $\alpha = aa^*$ en $\beta = a^*a$. Dan geldt $\rho(\alpha) = \rho(\beta) = \{a^n \mid n > 1\}$, dus $\alpha \equiv \beta$ terwijl $\alpha \neq \beta$.

5.12. Oefening

Toon aan dat de relatie $\equiv$ een equivalentie-relatie over R_Σ is, dat wil zeggen: toon aan dat $\equiv$ reflexief, transitief en symmetrisch is.

5.13. Eigenschap

Voor alle reguliere expressies α, β, γ over Σ gelden de volgende gelijkwaardigheden:

(1) $\alpha + \beta \equiv \beta + \alpha$
(2) $(\alpha+\beta) + \gamma \equiv \alpha + (\beta+\gamma)$
(3) $\alpha(\beta+\gamma) \equiv \alpha\beta + \alpha\gamma$
(4) $(\alpha+\beta)\gamma \equiv \alpha\gamma + \beta\gamma$
(5) $\alpha\Lambda \equiv \Lambda\alpha \equiv \alpha$
(6) $\alpha^* \equiv \Lambda + \alpha\alpha^*$
(7) $\alpha + \emptyset \equiv \alpha$
(8) $(\alpha+\beta)^* \equiv (\alpha^*+\beta^*)^* \equiv (\alpha^*\beta^*)^*$
(9) $\emptyset^* \equiv \Lambda$
(10) $(\alpha\beta)\gamma \equiv \alpha(\beta\gamma)$
(11) $\alpha + \alpha \equiv \alpha$
(12) $\alpha\emptyset \equiv \emptyset\alpha \equiv \emptyset$
(13) $\alpha\alpha^* \equiv \alpha^*\alpha$
(14) $\Lambda^* \equiv \Lambda$
(15) $(\alpha^*)^* \equiv \alpha^*$

Bewijs

(3) $\rho(\alpha(\beta+\gamma)) = \rho(\alpha)(\rho(\beta) \cup \rho(\gamma))$
$= \{xy \mid x \in \rho(\alpha);\ y \in \rho(\beta) \cup \rho(\gamma)\} =$
$= \{xy \mid x \in \rho(\alpha);\ y \in \rho(\beta) \text{ of } y \in \rho(\gamma)\} =$
$= \{xy \mid x \in \rho(\alpha);\ y \in \rho(\beta)\} \cup \{xy \mid x \in \rho(\alpha);\ y \in \rho(\gamma)\} =$
$= \rho(\alpha)\rho(\beta) \cup \rho(\alpha)\rho(\gamma) = \rho(\alpha\beta + \alpha\gamma)$; dus $\alpha(\beta+\gamma) \equiv \alpha\beta + \alpha\gamma$.

(15) $\rho((\alpha^*)^*) = (\rho(\alpha^*))^* = ((\rho(\alpha))^*)^* =$

$= \{x_1x_2\cdots x_n \mid n \geq 0;\ x_i \in (\rho(\alpha))^*,\ 1 \leq i \leq n\} =$

$= \{x_{11}\cdots x_{1n_1}x_{21}\cdots x_{2n_2}\cdots x_{n1}\cdots x_{nn_n} \mid n \geq 0,\ n_i \geq 0$

voor $1 \leq i \leq n;\ x_{ij} \in \rho(\alpha)\} =$

$= \{y_1\cdots y_m \mid m \geq 0;\ y_i \in \rho(\alpha),\ 1 \leq i \leq m\} = (\rho(\alpha))^* = \rho(\alpha^*).$

Dus $(\alpha^*)^* \equiv \alpha^*$.

De overige bewijzen worden ter oefening aan de lezer overgelaten.
Einde bewijs

Met behulp van eigenschap 5.13. kan men vaak reguliere expressies herschrijven tot eenvoudiger gelijkwaardige expressies; bijvoorbeeld

$(a^*b^*)^*a^* \equiv$ (8)

$(a+b)^*a^* \equiv$ (6)

$(a+b)^*(\Lambda+aa^*) \equiv$ (3)

$(a+b)^*\Lambda + (a+b)^*aa^* \equiv$ (5)

$(a+b)^* + (a+b)^*aa^*$

Nu is $\rho((a+b)^*aa^*) \subseteq \{a,b\}^*$ en $\rho((a+b)^*) = \{a,b\}^*$, zodat $(a^*b^*)^*a^* \equiv (a+b)^*$.

De rest van dit hoofdstuk is gewijd aan het bewijs, dat de klasse van talen, die met reguliere expressies kunnen worden gedefinieerd, samenvalt met $\mathscr{L}_3$. In hoofdstuk 1 werd al, vooruitlopend op dit feit, opgemerkt, dat een type 3 taal of grammatica ook wel een reguliere taal respectievelijk grammatica genoemd wordt. Zoals gebruikelijk verloopt het bewijs in twee stappen:

5.14. Eigenschap

Voor elk alfabet Σ en voor elke reguliere expressie α geldt, dat $\rho(\alpha) \in \mathscr{L}_3$.

Bewijs 1:
Met inductie naar de opbouw van α wordt aangetoond dat er een rechtslineaire grammatica G_α bestaat met $L(G_\alpha) = \rho(\alpha)$.

Initialisatiestap: $|\alpha| = 1$. Dus
of $\alpha = \emptyset$; beschouw $G_\emptyset = (V,\Sigma,\{S \to S\},S)$;
dan is $L(G_\emptyset) = \emptyset = \rho(\alpha)$.
of $\alpha = \Lambda$; beschouw $G_\Lambda = (V,\Sigma,\{S \to \lambda\},S)$;
dan is $L(G_\Lambda) = \{\lambda\} = \rho(\Lambda)$.
of $\alpha = a$; voor één of andere a in Σ; beschouw $G_a = (V,\Sigma,\{S \to a\},S)$;
dan is $L(G_a) = \{a\} = \rho(a)$.
In bovengenoemde grammatica's is steeds $V = \Sigma \cup \{S\}$.

Inductieveronderstelling: Stel dat voor alle reguliere expressies α over Σ met $|\alpha| \leq m$ $(m \geq 1)$ er een rechtslineaire grammatica G_α bestaat met $L(G_\alpha) = \rho(\alpha)$.

Inductiestap: Zij β een reguliere expressie over Σ met $|\beta| > m$, terwijl voor elk deelwoord x van β geldt: als $x \neq \beta$ en $x \in R_\Sigma$, dan $|x| \leq m$. Dan bezit β één van de volgende drie vormen:

(1) $\beta = \alpha_1 + \alpha_2$; (2) $\beta = \alpha_1\alpha_2$; (3) $\beta = \alpha_1^*$,

waarin $\alpha_i \in R_\Sigma$ met $|\alpha_i| \leq m$ $(i=1,2)$. Volgens de inductieveronderstelling bestaan er rechtslineaire grammatica's $G_i=(V_i,\Sigma,P_i,S_i)$ met $L(G_i) = \rho(\alpha_i)$, $i = 1,2$. Zonder verlies van algemeenheid mag men aannemen, dat $(V_1 \setminus \Sigma) \cap (V_2 \setminus \Sigma) = \emptyset$. In alle bovengenoemde gevallen zal aangetoond worden dat er een rechtslineaire grammatica G_β bestaat met $L(G_\beta) = \rho(\beta)$.

Ad (1): Definieer een grammatica $G_\beta = (V,\Sigma,P,S)$ met $V = V_1 \cup V_2 \cup \{S\}$, S is een nieuw hulpsymbool, $P = P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\}$. Dan is $L(G_\beta) = L(G_1) \cup L(G_2) = \rho(\alpha_1) \cup \rho(\alpha_2) = \rho(\beta)$ (Ga dit na).

Ad (2): Definieer een grammatica $G\beta = (V,\Sigma,P,S_1)$ met $V = V_1 \cup V_2$ en $P = \{A \to \xi \mid A \to \xi \in P_1 \text{ en } \xi \notin \Sigma^*\} \cup \{A \to \xi S_2 \mid A \to \xi \in P_1 \text{ en } \xi \in \Sigma^*\} \cup P_2$. Dan is $L(G\beta) = L(G_1)L(G_2) = \rho(\alpha_1)\rho(\alpha_2) = \rho(\beta)$ (Toon dit aan).

Ad (3): Definieer een grammatica $G_\beta = (V,\Sigma,P,S)$ met $V = V_1 \cup \{S\}$, S is een nieuw hulpsymbool, en $P = \{S \to \lambda,\ S \to S_1\} \cup \{A \to \xi \mid A \to \xi \in P_1 \text{ en } \xi \notin \Sigma^*\} \cup \{A \to \xi S \mid A \to \xi \in P_1 \text{ en } \xi \in \Sigma^*\}$. Nu is $L(G_\beta) = (L(G_1))^* = (\rho(\alpha_1))^* = \rho(\beta)$ (Bewijs dit).

Hiermee is het bewijs door middel van inductie naar de opbouw van de reguliere expressie α voltooid en mag worden geconcludeerd, dat voor elke reguliere expressie α geldt $\rho(\alpha) \in \mathfrak{L}_3$ (zie eigenschap 3.6.).

Bewijs 2:

In plaats van rechtslineaire grammatica's worden in dit alternatieve bewijs de in hoofdstuk 4 behandelde transitiesystemen gebruikt. Ook dit bewijs verloopt met inductie naar de opbouw van de reguliere expressie α en heeft dus dezelfde globale structuur als het vorige bewijs.

Initialisatiestap: $|\alpha| = 1$. Dus

of $\alpha = \emptyset$; beschouw $A_\emptyset = (\{q_0,f\}, \Sigma, \sigma_T, \{q_0\}, \{f\})$ waarin geldt $\sigma_T(q,x) = \emptyset$ voor alle $q \in \{q_0,f\}$ en alle $x \in \Sigma^*$. Dan geldt $L(A_\emptyset) = \emptyset = \rho(\alpha)$

of $\alpha = \Lambda$; beschouw $A_\Lambda = (\{q_0,f\}, \Sigma, \sigma_T, \{q_0\}, \{f\})$ waarin σ_T alleen ongelijk $\emptyset$ is voor (q_0,λ): namelijk $\sigma_T(q_0,\lambda) = \{f\}$. Dan is $L(A_\Lambda) = \{\lambda\} = \rho(\Lambda)$.

of $\alpha = a$; voor één of andere a in Σ; beschouw A_a die alleen van A_Λ verschilt in de definitie van σ_T, nl. $\sigma_T(q_0,a) = \{f\}$. Dan geldt: $L(A_a) = \{a\} = \rho(a)$.

Inductieveronderstelling: Stel dat voor alle reguliere expressies α over Σ met $|\alpha| \leq m$ $(m \geq 1)$ geldt dat er transitiesystemen A_α bestaan met $L(A_\alpha) = \rho(\alpha)$.

Inductiestap: Zij β een reguliere expressie over Σ met $|\beta| > m$, terwijl voor elk deelwoord x van β geldt: als $x \neq \beta$ en $x \in R_\Sigma$, dan $|x| \leq m$. Dan bezit β één van de volgende drie vormen:

(1) $\beta = \alpha_1 + \alpha_2$; (2) $\beta = \alpha_1\alpha_2$; (3) $\beta = \alpha_1^*$,

waarin $\alpha_i \in R_\Sigma$ met $|\alpha_i| \leq m$ $(i = 1,2)$. Volgens de inductieveronderstelling bestaan er transitiesystemen

$A_i = (Q_i, \Sigma, \sigma_{Ti}, Q_{0i}, F_i)$ met $L(A_i) = \rho(\alpha_i)$, $i = 1,2$. Zonder verlies van algemeenheid mag men aannemen dat $Q_1 \cap Q_2 = \emptyset$. Voor de gevallen (1), (2) en (3) zal worden aangetoond, dat er een transitiesysteem A_β bestaat met $L(A_\beta) = \rho(\beta)$.

Ad (1): Definieer een transitiesysteem $A_\beta = (Q, \Sigma, \sigma_T, \{q_0\}, \{f\})$ met $Q = Q_1 \cup Q_2 \cup \{q_0, f\}$, q_0 en f zijn nieuwe toestanden, terwijl

$\sigma_T(q_0, \lambda) = Q_{01} \cup Q_{02}$
$\sigma_T(q,w) = \sigma_{T1}(q,w)$ voor alle $q \in Q_1$ en alle $w \in \Sigma^*$,
$\sigma_T(q,w) = \sigma_{T2}(q,w)$ voor alle $q \in Q_2$ en alle $w \in \Sigma^*$,
$\sigma_T(q,\lambda) = \{f\}$ voor alle $q \in F_1 \cup F_2$.

Dan is $L(A_\beta) = L(A_1) \cup L(A_2) = \rho(\alpha_1) \cup \rho(\alpha_2) = \rho(\beta)$ (Ga dit na).

Ad (2): Definieer een transitiesysteem $A_\beta = (Q, \Sigma, \sigma_T, Q_{01}, F_2)$ met $Q = Q_1 \cup Q_2$ en

$\sigma_T(q,w) = \sigma_{T1}(q,w)$ voor alle $q \in Q_1$ en alle $w \in \Sigma^*$,
$\sigma_T(q,w) = \sigma_{T2}(q,w)$ voor alle $q \in Q_2$ en alle $w \in \Sigma^*$,
$\sigma_T(q,\lambda) = Q_{02}$ voor alle $q \in F_1$.

Dan is $L(A_\beta) = L(A_1)L(A_2) = \rho(\alpha_1)\rho(\alpha_2) = \rho(\beta)$ (Toon dit aan).

Ad (3): Definieer een transitiesysteem $A_\beta = (Q, \Sigma, \sigma_T, \{q_0\}, \{f\})$ met $Q = Q_1 \cup \{q_0, f\}$, q_0 en f zijn nieuwe toestanden, terwijl

$\sigma_T(q_0, \lambda) = Q_{01} \cup \{f\}$
$\sigma_T(q,w) = \sigma_{T1}(q,w)$ voor alle $q \in Q_1$ en alle $w \in \Sigma^*$,
$\sigma_T(q,\lambda) = Q_{01} \cup \{f\}$ voor alle $q \in F_1$.

Dan geldt $L(A_\beta) = (L(A_1))^* = (\rho(\alpha_1))^* = \rho(\beta)$.

De inductie is hiermee voltooid. Daar transitiesystemen reguliere talen accepteren (eigenschap 4.37.) mag worden geconcludeerd, dat voor elke $\alpha \in R\Sigma$ geldt: $\rho(\alpha) \in \mathcal{L}_3$.
Einde bewijzen

De constructies in deze bewijzen kunnen gebruikt worden om voor een reguliere expressie α een rechtslineaire grammatica G_α of een transitiesysteem A_α te construeren met $\rho(\alpha) = L(G_\alpha) = L(A_\alpha)$. Met de constructies uit de hoofdstukken 3 en 4 kunnen G_α of A_α dan verder gemodificeerd worden tot een type 3 grammatica en, respectievelijk, een eindige automaat voor $\rho(\alpha)$, indien dat gewenst wordt.

5.15. Voorbeeld

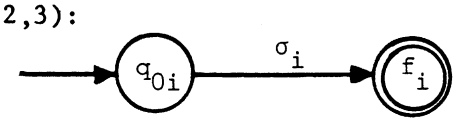
Beschouw de reguliere expressie $\alpha = a + ba^*$. Deze expressie is te schrijven als $\alpha = \alpha_1 + \alpha_2\alpha_3^*$ met $\alpha_1 = \alpha_3 = a$ en $\alpha_2 = b$. Toepassen van de constructie levert in eerste instantie (initialisatiestap): $G_i = (V_i,\Sigma,\{S_i \to \sigma_i\},S_i)$ met $\Sigma = \{a,b\}$, $V_i = \Sigma \cup \{S_i\}$, $\sigma_1 = \sigma_3 = a$ en $\sigma_2 = b$. Een grammatica voor $\alpha_4 = \alpha_3^*$ volgt uit G_3 met behulp van (3) in de inductiestap: $G_4 = (V_4,\Sigma,P_4,S_4)$ met S_4 is een nieuw hulpsymbool, $V_4 = V_3 \cup \{S_4\}$, $P_4 = \{S_4 \to \lambda,\ S_4 \to S_3,\ S_3 \to aS_4\}$; $L(G_4) = \rho(\alpha_3^*) = \{a^n \mid n \geq 0\}$.

Vervolgens wordt $G_5 = (V_5,\Sigma,P_5,S_5)$ met (2) uit de inductiestap geconstrueerd uitgaande van G_2 en G_4: $V_5 = V_2 \cup V_4$, $S_5 = S_2$, $P_5 = \{S_2 \to bS_4,\ S_4 \to \lambda,\ S_4 \to S_3,\ S_3 \to aS_4\}$; $L(G_5) = \rho(\alpha_2\alpha_3^*) = \{ba^n \mid n \geq 0\}$.

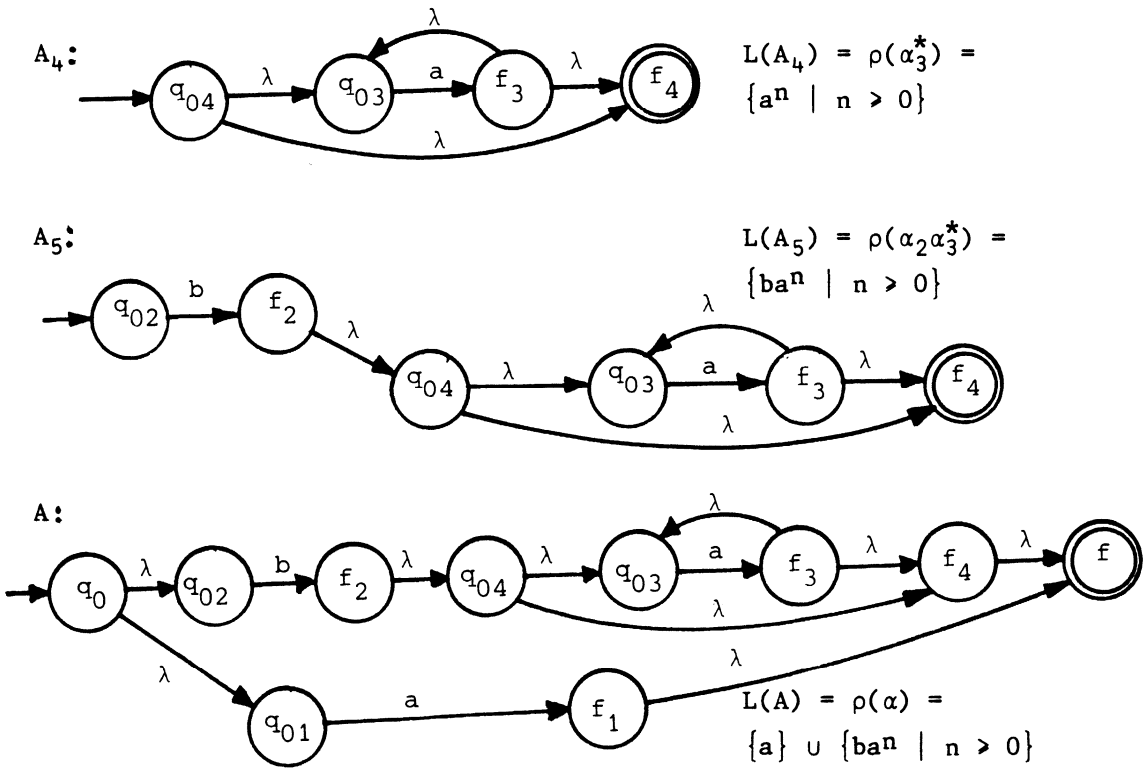
Tenslotte wordt met (1) uit de inductiestap een rechtslineaire grammatica $G = (V,\Sigma,P,S)$ voor $\rho(\alpha)$ gevonden uitgaande van G_1 en G_5: $V = V_1 \cup V_5 \cup \{S\}$, S is een nieuw hulpsymbool, $P = \{S \to S_1,\ S_1 \to S_2,\ S_1 \to a,\ S_2 \to bS_4,\ S_4 \to \lambda,\ S_4 \to S_3,\ S_3 \to aS_4\}$ en $L(G) = \rho(\alpha) = \{a\} \cup \{ba^n \mid n \geq 0\}$.

Op soortgelijke wijze kan men met behulp van het tweede bewijs van eigenschap 5.14. een transitiesysteem verkrijgen, dat $\rho(\alpha)$ accepteert: achtereenvolgens levert dit de volgende transitiesystemen op, die elk door hun transitiediagram worden weergegeven.

A_i $(i = 1,2,3)$:



$\sigma_1 = \sigma_3 = a$;
$\sigma_2 = b$; $L(A_i) = \{\sigma_i\}$



5.16. Eigenschap

Voor elke taal $L \subseteq \Sigma^*$ in $\mathcal{L}_3$ bestaat er een reguliere expressie α over Σ, zodanig dat $\rho(\alpha) = L$.

Bewijs

Als $L \subseteq \Sigma^*$ een type 3 taal is, dan bestaat er volgens eigenschap 4.15. een deterministische eindige automaat $A = (Q,\Sigma,\sigma,q_0,F)$, die L accepteert. Neem aan dat $Q = \{q_0,q_1,\ldots,q_n\}$. Definieer nu voor alle i en j ($0 \leq i,j \leq n$) en alle k ($0 \leq k \leq n+1$) de verzameling

$$R_{ij}^k = \{x \in \Sigma^* \mid (q_i,x) \vdash^* (q_j,\lambda) \text{ en als voor een of andere } y \in \Sigma^* \quad (q_i,x) \vdash^* (q_m,y) \text{ dan geldt: } m < k, \text{ of } y = \lambda \text{ en } m = j, \text{ of } y = x \text{ en } m = i\}.$$

Informeel betekent dit, dat R_{ij}^k de verzameling van die woorden over Σ is, die A vanuit toestand q_i overvoeren naar toestand q_j zonder onderweg een toestand q_m te bezoeken met $m \geq k$. In het bijzon-

der betekent dit, dat als k = n+1, alle toestanden onderweg bezocht mogen worden, zodat

$$R_{ij}^{n+1} = \{x \in \Sigma^* \mid (q_i,x) \overset{*}{\vdash} (q_j, \lambda)\}, \text{ en } L(A) = \cup \{R_{0j}^{n+1} \mid q_j \in F\}.$$

Eerst zal nu worden aangetoond, dat er voor elke verzameling R_{ij}^k een reguliere expressie α_{ij}^k over Σ bestaat zodanig, dat $\rho(\alpha_{ij}^k) = R_{ij}^k$. Met behulp van dit resultaat wordt dan bewezen dat er een reguliere expressie α over Σ bestaat met $\rho(\alpha) = L(A)$.
Het bewijs van de eerstgenoemde bewering verloopt met inductie naar k.

Initialisatiestap: k = 0. Uit de definitie van R_{ij}^k volgt

$R_{ij}^0 = \{a \in \Sigma \mid \sigma(q_i,a) = q_j\} \cup \{\lambda\}$ als i = j, en
$R_{ij}^0 = \{a \in \Sigma \mid \sigma(q_i,a) = q_j\}$ als i ≠ j.

De overeenkomstige reguliere expressie is dan respectievelijk

$\alpha_{ij}^0 = a_1 + \ldots + a_s + \Lambda$ als i = j, en
$\alpha_{ij}^0 = a_1 + \ldots + a_s$ als i ≠ j,

vooropgesteld dat $R_{ij}^0 \cap \Sigma = \{a_1,\ldots,a_s\}$ voor een of andere s ⩾ 0. In beide gevallen is het evident, dat $\rho(\alpha_{ij}^0) = R_{ij}^0$.

Inductieveronderstelling: Neem aan dat voor alle k ⩽ p (p ⩾ 0) en voor alle i en j (0 ⩽ i,j ⩽ n) er reguliere expressies α_{ij}^k bestaan zodanig dat $\rho(\alpha_{ij}^k) = R_{ij}^k$.

Inductiestap: De verzameling R_{ij}^{p+1} voldoet aan de betrekking

$$R_{ij}^{p+1} = R_{ij}^p \cup R_{ip}^p (R_{pp}^p)^* R_{pj}^p.$$

Deze gelijkheid stelt, dat A toestand q_j vanuit q_i kan bereiken zonder een toestand q_m met m ⩾ p+1 te bezoeken door

of q_j vanuit q_i te bereiken zonder een toestand q_m met $m > p$ te bezoeken,

of (1) vanuit q_i naar q_p te gaan, dan (2) bij herhaling uit q_p te vertrekken en weer naar q_p terug te keren, en (3) tenslotte vanuit q_p de toestand q_j te bereiken; dit alles onder de veronderstelling, dat in geen van de stappen (1), (2) en (3) een toestand q_m met $m > p$ bezocht wordt.

Volgens de inductieveronderstelling bestaan er dan reguliere expressies α_{ij}^p, α_{ip}^p, α_{pp}^p en α_{pj}^p over Σ zodanig dat

$\rho(\alpha_{ij}^p) = R_{ij}^p$, $\rho(\alpha_{ip}^p) = R_{ip}^p$, $\rho(\alpha_{pp}^p) = R_{pp}^p$ en $\rho(\alpha_{pj}^p) = R_{pj}^p$. Dan geldt ook

$R_{ij}^{p+1} = \rho(\alpha_{ij}^p) \cup \rho(\alpha_{ip}^p)(\rho(\alpha_{pp}^p))^* \rho(\alpha_{pj}^p)$. Dus $R_{ij}^{p+1} = \rho(\alpha_{ij}^{p+1})$ waarin de gezochte expressie α_{ij}^{p+1} gedefinieerd wordt door

$$\alpha_{ij}^{p+1} = \alpha_{ij}^p + \alpha_{ip}^p (\alpha_{pp}^p)^* \alpha_{pj}^p.$$

Hiermee is de inductie voltooid. Rest nu nog de constructie van de reguliere expressie α: indien

$F = \{q_{j_1}, \ldots, q_{j_t}\}$ dan volgt uit $L(A) = \cup \{R_{0j}^{n+1} \mid q_j \in F\}$ dat de reguliere expressie $\alpha = \alpha_{0j_1}^{n+1} + \ldots + \alpha_{0j_t}^{n+1}$ voldoet aan $\rho(\alpha) = L(A)$.

Einde bewijs

5.17. Voorbeeld

Gegeven is de deterministische eindige automaat $A = (\{q_0, q_1, q_2\}, \{a,b\}, \sigma, q_0, \{q_1\})$ waarvan de transitiefunctie σ luidt:

$\sigma(q_0,a) = q_0$; $\sigma(q_1,a) = q_1$; $\sigma(q_2,a) = q_2$;

$\sigma(q_0,b) = q_1$; $\sigma(q_1,b) = q_2$; $\sigma(q_2,b) = q_0$.

Aan de hand van het bewijs van eigenschap 5.16. gaan we een reguliere expressie voor L(A) construeren:

$$L(A) = \cup \{R^3_{0j} \mid q_j \in \{q_1\}\} = R^3_{01} = R^2_{01} \cup R^2_{02}(R^2_{22})^* R^2_{21}, \qquad (1)$$

terwijl voor de vier hierin voorkomende verzamelingen geldt:

$$\begin{aligned} R^2_{01} &= R^1_{01} \cup R^1_{01}(R^1_{11})^* R^1_{11} \\ R^2_{02} &= R^1_{02} \cup R^1_{01}(R^1_{11})^* R^1_{12} \\ R^2_{22} &= R^1_{22} \cup R^1_{21}(R^1_{11})^* R^1_{12} \\ R^2_{21} &= R^1_{21} \cup R^1_{21}(R^1_{11})^* R^1_{11} \end{aligned} \qquad (2)$$

In de rechterleden komen in totaal zes verzamelingen voor, waarvoor geldt:

$$\begin{aligned} R^1_{01} &= R^0_{01} \cup R^0_{00}(R^0_{00})^* R^0_{01} \\ R^1_{11} &= R^0_{11} \cup R^0_{10}(R^0_{00})^* R^0_{01} \\ R^1_{02} &= R^0_{02} \cup R^0_{00}(R^0_{00})^* R^0_{02} \\ R^1_{12} &= R^0_{12} \cup R^0_{10}(R^0_{00})^* R^0_{02} \\ R^1_{22} &= R^0_{22} \cup R^0_{20}(R^0_{00})^* R^0_{02} \\ R^1_{21} &= R^0_{21} \cup R^0_{20}(R^0_{00})^* R^0_{01} \end{aligned} \qquad (3)$$

Voor de negen in de rechterleden voorkomende verzamelingen R^0_{ij} vinden we met behulp van de initialisatiestap uit het bewijs de volgende reguliere expressies:

$$\begin{aligned} \alpha^0_{00} &= \alpha^0_{11} = \alpha^0_{22} = a + \Lambda \\ \alpha^0_{01} &= \alpha^0_{12} = \alpha^0_{21} = b \\ \alpha^0_{10} &= \alpha^0_{21} = \alpha^0_{02} = \emptyset \end{aligned}$$

Met (3), en vereenvoudigen tot gelijkwaardige reguliere expressies levert dit

$$\begin{aligned} \alpha^1_{01} &= b + (a + \Lambda)(a + \Lambda)^* b \equiv a^* b \\ \alpha^1_{11} &= (a + \Lambda) + \emptyset(a + \Lambda)^* b \equiv a + \Lambda \\ \alpha^1_{02} &= \emptyset + (a + \Lambda)(a + \Lambda)^* \emptyset \equiv \emptyset \\ \alpha^1_{12} &= b + \emptyset(a + \Lambda)^* \emptyset \equiv b \\ \alpha^1_{22} &= (a + \Lambda) + b(a + \Lambda)^* \emptyset \equiv a + \Lambda \\ \alpha^1_{21} &= \emptyset + b(a + \Lambda)^* b \equiv ba^* b \end{aligned}$$

Uit (2) en de α^1_{ij}, verkrijgen we nu de α^2_{ij}

$\alpha^2_{01} = a^*b + a^*b(a + \Lambda)^*a \equiv a^*ba^*$
$\alpha^2_{02} = \emptyset + a^*b(a + \Lambda)^*b \equiv a^*ba^*b$
$\alpha^2_{22} = (a + \Lambda) + ba^*b(a + \Lambda)^*b \equiv a + \Lambda + ba^*ba^*b$
$\alpha^2_{21} = ba^*b + ba^*b(a + \Lambda)^*a \equiv ba^*ba^*$

Tenslotte vinden we met behulp van (1) de gevraagde reguliere expressie α:

$\alpha = \alpha^3_{01} = a^*ba^* + a^*ba^*b(a + \Lambda + ba^*ba^*b)^*ba^*ba^*.$

Er geldt $L(A) = \rho(\alpha) = \{w \mid w \in \{a,b\}^* \text{ en } \#(b,w) = 1 \underline{\text{mod}}\ 3\}$ (Toon dit aan). •

5.18. Eigenschap.

Een taal $L \subseteq \Sigma^*$ is een type 3 taal dan en slechts dan als L door een reguliere expressie over Σ gedefinieerd wordt.

Bewijs
De bewering volgt direct uit de eigenschappen 5.14. en 5.16.
Einde bewijs

5.19. Oefeningen

(1) Construeer een eindige automaat, die de taal gedefinieerd door de reguliere expressie $(0^*+11)^*01$, accepteert.

(2) Construeer een type 3 grammatica, die de taal gedefinieerd door de reguliere expressie $(11+00)^*(0+1)^*$, voortbrengt.

(3) Zij $A = (\{0,1,2\}, \{a,b\}, \sigma, 0, \{2\})$ een eindige automaat met

$\sigma(0,a) = 1$; $\sigma(1,a) = 2$; $\sigma(2,a) = 0$;
$\sigma(0,b) = 2$; $\sigma(1,b) = 0$; $\sigma(2,b) = 1$.

Construeer een reguliere expressie α over $\{a,b\}$, zodanig dat $\rho(\alpha) = L(A)$.

(4) Beschouw de grammatica $G = (V,\Sigma,P,S)$ met $\Sigma = \{a,b\}$, $V = \Sigma \cup \{S,A,B\}$ en $P = \{S \to bA,\ A \to aB,\ B \to aS,\ B \to bA,\ B \to a\}$. Construeer een reguliere expressie α over $\{a,b\}$ zodanig dat $\rho(\alpha) = L(G)$.

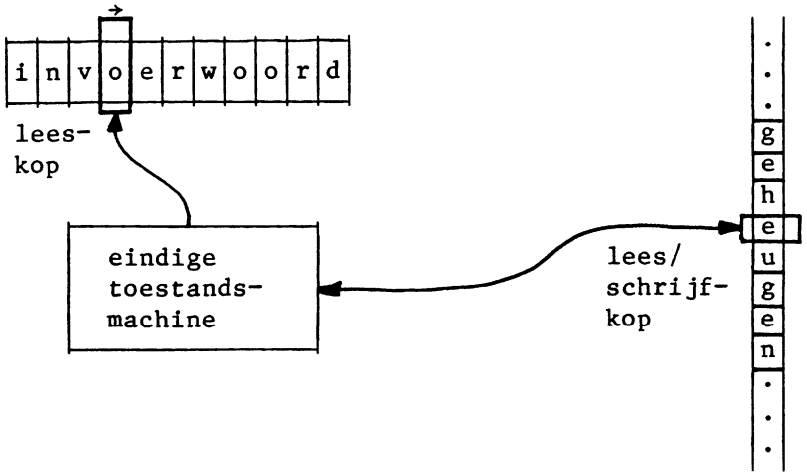
6. STAPELAUTOMATEN

De eindige automaten uit hoofdstuk 4 kunnen elk eindig veel informatie bewaren. De hoeveelheid informatie, die een bepaalde automaat kan onthouden is dus begrensd en wordt alleen bepaald door het aantal toestanden van deze automaat; ze is dus in het bijzonder onafhankelijk van de grootte van de invoer.

Teneinde een type automaat te verkrijgen, dat geschikt is om ook niet-reguliere talen te accepteren, moet de hoeveelheid informatie, die de automaat kan onthouden worden vergroot. De hoeveelheid informatie, die de automaat moet bewaren, kan afhankelijk zijn van de lengte van de invoer.

Beschouw namelijk de volgende niet-reguliere, en dus niet door een eindige automaat te accepteren, taal $L = \{a^n b^n \mid n \geq 0\}$. Onder de aanname, dat een invoerwoord w slechts éénmaal van links naar rechts gelezen mag worden, zal een automaat A, die L accepteert, eerst de waarde van n bepalen door het aantal a's in w te tellen, en dan controleren of deze waarde overeenkomt met het aantal b's in w. Om het aantal a's te tellen is $\lceil \log_2 \#(a,w) \rceil$ bit geheugenruimte vereist. Daar de functie $f = \lambda n \lceil \log_2 n \rceil$ onbegrensd is (f(n) kan willekeurig groot worden als n voldoende toeneemt.) dient A over een (werk)geheugen te beschikken, dat tenminste toeneemt tot de grootte $\lceil \log_2 \frac{1}{2}|w| \rceil$ als een invoerwoord w wordt aangeboden (Merk op, dat dit geen bewijs van het feit "$L \notin \mathcal{L}_3$" is, maar slechts een intuïtieve gedachtengang. Een bewijs van "$L \notin \mathcal{L}_3$" maakt gebruik van de pompstelling voor reguliere talen; zie voorbeeld 3.12.(1)).

Om de hoeveelheid informatie, die kan worden bewaard, te vergroten, voegt men aan de (fysische interpretatie van de) eindige automaat een geheugen toe in de vorm van een band verdeeld in cellen:



Turing-machine

Iedere cel van deze geheugenband kan eindig veel informatie bevatten (d.w.z. een symbool uit een of ander speciaal "geheugen"-alfabet). Langs de geheugenband loopt een lees/schrijfkop. De eindige toestandsmachine is vergelijkbaar met de eindige automaat uit hoofdstuk 4: er is een eindige verzameling Q van toestanden, een invoeralfabet Σ, begin- en eindtoestanden, en een transitiefunctie, die bij een drietal bestaande uit huidige toestand, invoersymbool en geheugensymbool bepaalt wat de volgende toestand zal zijn, welk symbool naar het geheugen wordt geschreven en hoe de leeskop en de lees/schrijfkop worden verplaatst. Op elk moment in de berekening wordt de leeskop 0 of 1 plaatsen naar rechts verplaatst: zodoende kan een berekening zeer veel langer duren dan de lengte van het invoerwoord (zie ook de transitiesystemen uit hoofdstuk 4).

Bij het begin van een berekening veronderstelt men dat de (informatie) inhoud van de oneindige geheugenband leeg is: elke cel bevat een speciaal blanco-symbool. Op elk tijdstip gedurende een berekening geldt dan, dat er slechts een eindig deel van het (oneindige) geheugen gebruikt is (dat wil zeggen: een eindig aantal cellen met een niet-blanco symbool gevuld is), daar de automaat tijdens dit deel van de berekening slechts een eindig aantal cellen bezocht kan hebben. Maar dit aantal kan wel willekeurig groot worden naarmate de berekening voortschrijdt.

Het wiskundig object dat overeenkomt met de hiervoor intuïtief omschreven machine heet een "Turing-machine" (naar de Engelse wiskundige A.M.

Turing (1912-1954)). De Turing-machine wordt uitvoeriger behandeld in het college al74 Theoretische Informatica II. De Turing-machine is een zeer algemeen machinemodel en kan als uitgangspunt dienen voor talrijke andere machinemodellen door het opleggen van beperkingen aan het oorspronkelijke model. (Vergelijk in dit verband hoe uitgaande van de algemene definitie 1.21. van grammatica, andere typen grammatica's werden afgeleid: definities 1.32 - 1.39.). Deze beperkingen hebben vanzelfsprekend betrekking op het gebruik van de geheugenband. Daarnaast kan men voor elke beperking een deterministische en een niet-deterministische machine beschouwen, al naar gelang de onderliggende eindige toestandsmachine deterministisch of niet-deterministisch is.

In dit dictaat komen alleen de volgende beperkingen en de daarbij behorende typen automaten ter sprake:

(0) geen beperking: dit levert de reeds besproken deterministische Turing-machine (DTM) en niet-deterministische Turing-machine (NTM).
(1) het aantal cellen van de geheugenband, dat gedurende een berekening van een invoerwoord w gebruikt mag worden is naar boven begrensd door $|w|$. De bijbehorende automaat heet de (niet)deterministische lineair begrensde automaat (DLBA en NLBA; Eng.: "(non)deterministic linear bounded automaton").
(2) de toegankelijkheid van de informatie op de geheugenband kan worden beperkt. De enige vorm daarvan, die hier besproken wordt, is de beperking tot het gebruik als een stapel.

Een stapelgeheugen moet worden gebruikt zoals een stapel borden of dienbladen in een kantine:
- het bovenste element (bord of blad) kan van de stapel worden gehaald;
- één of meer elementen (borden of bladen) kunnen boven op de stapel worden geplaatst.

In termen van de geheugenband betekent dit dat de lees/schrijfkop alleen mag opereren op de bovenste beschreven en de vlak daarboven gelegen onbeschreven cellen van de geheugenband; d.w.z. de automaat kan

- de inhoud van de bovenste beschreven cel overschrijven met een ander symbool;
- de inhoud van de bovenste beschreven cel wissen (d.i. overschrijven met een blanco-symbool) en de lees/schrijfkop één plaats naar beneden bewegen;
- de lees/schrijfkop bewegen naar de vlak daarboven gelegen aansluitende cellen en daar de blanco-symbolen vervangen door niet-blanco symbolen.

De bijbehorende automaat heet de (niet-)deterministische stapelautomaat (DPDA en NPDA) naar het Engelse "(non)deterministic push-down automaton". (De voorkeur wordt hier gegeven aan de Engelse afkorting omdat SA ook kan verwijzen naar de hier niet behandelde "stack automaton", die krachtiger is dan de PDA).

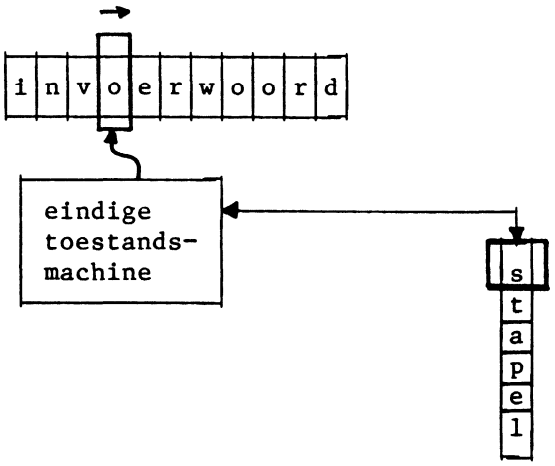
(3) De geheugenband mag in het geheel niet gebruikt worden; dit levert de al bekende (niet-)deterministische eindige automaat (DEA en NEA). Een gelijkwaardige beperking is: er mag slechts een constant aantal cellen van de geheugenband gebruikt worden (Ga dit na).

Zij nu voor elke bovengenoemde afkorting XYZ, $\mathcal{L}_{XYZ}$ de klasse van alle talen, die door XYZ-automaten geaccepteerd worden. Enkele belangrijke resultaten betreffende deze klassen van talen zijn:

(0) $\mathcal{L}_{NTM} = \mathcal{L}_{DTM} = \mathcal{L}_0$.
(1) $\mathcal{L}_{DLBA} \subseteq \mathcal{L}_{NLBA} = \mathcal{L}_1$ en het is tot op heden onbekend of de inclusie van $\mathcal{L}_{DLBA}$ in $\mathcal{L}_{NLBA}$ echt is, danwel dat beide klassen gelijk zijn: "Het LBA-probleem is nog steeds niet opgelost".
(2) $\mathcal{L}_3 \subset \mathcal{L}_{DPDA} \subset \mathcal{L}_{NPDA} = \mathcal{L}_2$.
(3) $\mathcal{L}_{DEA} = \mathcal{L}_{NEA} = \mathcal{L}_3$.

De gelijkheden onder (3) werden reeds in hoofdstuk 4 bewezen, terwijl de bewijzen van (0) en van (1) buiten het bestek van dit college vallen. De rest van dit hoofdstuk zal voornamelijk gewijd zijn aan de bewijzen van de resultaten onder (2); hetgeen tevens de titel van dit hoofdstuk verklaart.

Tekeningen van stapelautomaten wekken de indruk, dat de lees/schrijfkop vast staat, terwijl de stapel naar beneden groeit:



Stapelautomaat

In dit beeld is er geen sprake van cellen, die blanco zijn. Als het bovenste symbool (de "s") van de stapel wordt gehaald, dan wordt de stapel korter; het topstapelsymbool is daarna "t". Als wiskundig object wordt een stapelautomaat gedefinieerd in

6.1. Definitie: Een stapelautomaat A is een zevental $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$, met

(i) Q is een niet-lege eindige verzameling toestanden.

(ii) Σ is een alfabet van invoersymbolen.

(iii) Γ is een alfabet van stapelsymbolen.

(iv) σ is een functie van $Q \times (\Sigma \cup \{\lambda\}) \times \Gamma$ in de verzameling van eindige deelverzamelingen van $Q \times \Gamma^*$; σ heet de transitiefunctie van A.

(v) $q_0 \in Q$ is de begintoestand.

(vi) $\gamma_0 \in \Gamma$ is het bodemstapelsymbool.

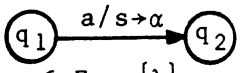
(vii) $F \subseteq Q$ is de verzameling van accepterende toestanden.

In bovenstaande definitie is in het geheel geen sprake van een stapel, alleen van een alfabet van stapelsymbolen. In de definitie van een stapelautomaat wordt vastgelegd wat de alfabetten zijn, wat de begintoestand is en dergelijke, en verder wat de transitiefunctie is. Pas als vastgelegd is, op welke wijze stapelautomaten berekeningen uitvoeren (zie de definities 6.4. en 6.6.) blijkt het mogelijk een fysische interpretatie in termen van stapels te geven.

Net als eindige automaten, kunnen ook stapelautomaten worden weergegeven met transitiediagrammen.

6.2. Definitie: Het transitiediagram van stapelautomaat A is een (gerichte, geëtiketteerde) graaf waarvoor geldt:

(1) de verzameling knopen is de verzameling toestanden van A. De markering van accepterende toestanden en de begintoestand is net als bij eindige automaten (zie 4.4. en 4.5.)

(2) er is een tak 

als en alleen als $(q_2,\alpha) \in \sigma(q_1,a,s)$, $a \in \Sigma \cup \{\lambda\}$, $s \in \Gamma$.

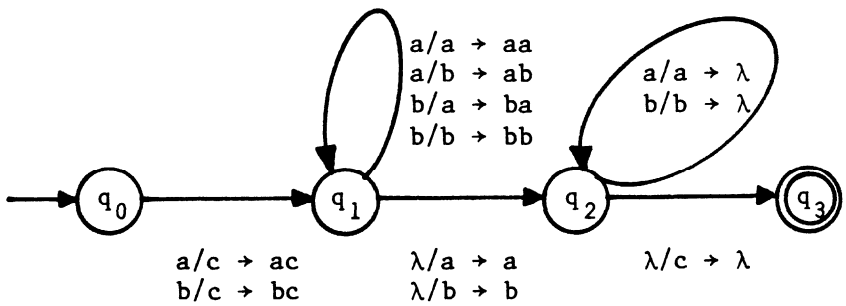
6.3. Voorbeeld

Gegeven is de stapelautomaat $A = (\{q_0,q_1,q_2,q_3\}, \{a,b\}, \{a,b,c\}, \sigma, q_0, c, \{q_3\})$ waarvan de transitiefunctie σ hieronder is weergegeven.

$\sigma(q_0,a,c) = \{(q_1,ac)\}$
$\sigma(q_1,a,b) = \{(q_1,ab)\}$
$\sigma(q_2,a,a) = \{(q_2,\lambda)\}$
$\sigma(q_0,b,c) = \{(q_1,bc)\}$
$\sigma(q_1,b,b) = \{(q_1,bb)\}$
$\sigma(q_2,b,b) = \{(q_2,\lambda)\}$

$\sigma(q_1,a,a) = \{(q_1,aa)\}$
$\sigma(q_1,\lambda,a) = \{(q_2,a)\}$
$\sigma(q_2,\lambda,c) = \{(q_3,\lambda)\}$
$\sigma(q_1,b,a) = \{(q_1,ba)\}$
$\sigma(q_1,\lambda,b) = \{(q_2,b)\}$
$\sigma(q,x,y) = \emptyset$ in alle overige gevallen.

Bij de stapelautomaat A hoort het onderstaande transitiediagram:



•

Het gedrag van $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ bestaat uit het herhaald toepassen van de transitiefunctie σ. Wiskundig wordt dit beschreven met behulp van het begrip configuratie.

6.4. Definitie: Zij $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ een stapelautomaat. Een configuratie (Eng.: "instantaneous description") van A is een drietal $(q,w,\xi) \in Q \times \Sigma^* \times \Gamma^*$. De verzameling van alle configuraties van A wordt aangegeven met C(A).

6.5. Voorbeeld

Beschouw nogmaals de stapelautomaat A uit voorbeeld 6.3. Configuraties van A zijn bijvoorbeeld:

$(q_0,aaaa,c)$ (q_1,aaa,ac)
(q_1,aa,aac) (q_2,aa,aac)
(q_2,a,ac) (q_2,λ,c)
(q_3,λ,λ) •

6.6. Definitie: Zij $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ een stapelautomaat. Een stap is een overgang van configuratie $c_1 \in C(A)$ naar configuratie $c_2 \in C(A)$, die door de transitiefunctie σ is toegestaan.
Notatie: $c_1 \vdash_A c_2$, of $c_1 \vdash c_2$ als duidelijk is om welke automaat het gaat. Zij $c_i = (q_i,w_i,\zeta_i)$, $i = 1,2$, configuraties van A. Dan geldt: $c_1 \vdash_A c_2$ als en alleen als $\exists a \in \Sigma \cup \{\lambda\} \exists \gamma \in \Gamma \exists \alpha, \xi \in \Gamma^*$:
- $aw_1 = w_2$ en
- $\zeta_1 = \gamma\xi$ en $\zeta_2 = \alpha\xi$ èn
- $(q_2,\alpha) \in \sigma(q_1,a,\gamma)$

6.7. Definitie:
- het doen van nul stappen in een berekening door een stapelautomaat A wordt genoteerd als $\vdash_A^0$:
  $c \vdash_A^0 c'$ dan en slechts dan als $c = c'$;
- het doen van k, $k \geq 1$, stappen wordt genoteerd als $\vdash_A^k$:
  $c \vdash_A^k c'$ dan en slechts dan als er configuraties $c = c_0,c_1,c_2,\ldots,c_k = c'$ bestaan, zodanig dat voor $\forall i$, $0 \leq i < k$, geldt $c_i \vdash_A c_{i+1}$;

- het doen van een willekeurig aantal maar tenminste één stap wordt genoteerd als $\vdash_A^+$:
  $c \vdash_A^+ c'$ dan en slechts dan als er een k, $k \geq 1$ is, zodanig dat $c \vdash_A^k c'$.

- het doen van een willekeurig aantal stappen wordt genoteerd als $\vdash_A^*$:
  $c \vdash_A^* c'$ dan en slechts dan als $c \vdash_A^+ c'$ of $c \vdash_A^0 c'$.

De configuraties van een stapelautomaat A en het stappen van configuratie naar configuratie hebben in de fysische interpretatie van A de volgende betekenis:

- $c = (q,w,\xi)$.
  Dit duidt aan dat A in toestand q verkeert, w de resterende invoer en ξ de stapelinhoud is. De eerste letter van ξ is het topstapelsymbool en de laatste letter van ξ is het onderste stapelsymbool (Deze hoeft niet in elke configuratie het bodemstapelsymbool van A te zijn).

- $(q,aw,\gamma\xi) \vdash (q',w,\alpha\xi)$.
  De automaat gaat van toestand q naar toestand q', haalt het topstapelsymbool γ van de stapel en schrijf $\alpha \in \Gamma^*$ boven op de stapel. In het vervangen van γ door α zijn alle drie de in de inleiding genoemde mogelijkheden voor de lees/schrijfkop vervat. Merk op, dat een berekening niet kan worden voortgezet als de stapel leeg is; immers bij elke stap moet er een topstapelsymbool zijn. Bij de overgang van $(q,aw,\gamma\xi)$ naar $(q',w,\alpha\xi)$ "verbruikt" de stapelautomaat $a \in \Sigma \cup \{\lambda\}$. Dit moet als volgt worden geinterpreteerd:

  - als $a \in \Sigma$ verschuift de leeskop op de invoerband een plaats naar rechts, zodat het volgende invoersymbool ter beschikking kan komen;
  - als $a = \lambda$ dan verandert de positie van de leeskop niet. Merk op dat λ geen symbool is dat op de invoerband staat. Als er onder de leeskop een symbool staat, is dat een symbool uit Σ. De notatie $a = \lambda$ wordt gebruikt om aan te geven dat er geen echte invoer wordt verbruikt.

Het accepteren van woorden door een stapelautomaat kan op twee verschillende manieren gedefinieerd worden: accepteren met accepterende toestanden (Eng.: "by final state") en accepteren met een lege stapel (Eng.: "by empty stack", "by null stack").

6.8. Definitie: Een woord $w \in \Sigma^*$ wordt door de stapelautomaat $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ met accepterende toestanden geaccepteerd dan en slechts dan als $\exists q \in F \exists \alpha \in \Gamma^* [(q_0,w,\gamma_0) \overset{*}{\vdash} (q,\lambda,\alpha)]$.

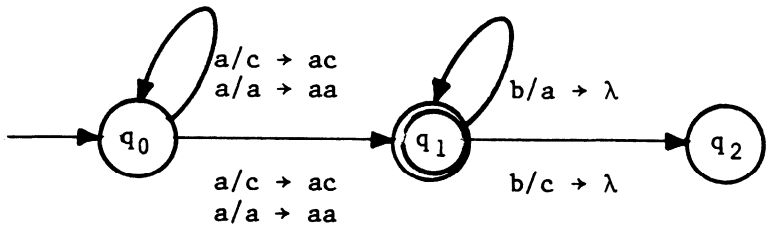
De taal $L_F(A)$ die door de stapelautomaat A met accepterende toestanden wordt geaccepteerd, is de verzameling
$L_F(A) = \{w \mid \exists q \in F \exists \alpha \in \Gamma^* [(q_0,w,\gamma_0) \overset{*}{\vdash} (q,\lambda,\alpha)]\}$

6.9. Definitie: Een woord $w \in \Sigma^*$ wordt door de stapelautomaat $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ met lege stapel geaccepteerd dan en slechts dan als $\exists q \in Q [(q_0,w,\gamma_0) \overset{*}{\vdash} (q,\lambda,\lambda)]$.

De taal $L_\Lambda(A)$ die door de stapelautomaat A met lege stapel wordt geaccepteerd, is de verzameling
$L_\Lambda(A) = \{w \mid \exists q \in Q [(q_0,w,\gamma_0) \overset{*}{\vdash} (q,\lambda,\lambda)]\}$.

6.10. Voorbeeld

Beschouw de stapelautomaat $A = (\{q_0,q_1,q_2\}, \{a,b\}, \{a,c\}, q_0, c, \{q_1\})$ met onderstaand transitiediagram:



Het woord aaa wordt door A met accepterende toestand geaccepteerd, immers

$(q_0,aaa,c) \vdash (q_0,aa,ac) \vdash (q_0,a,aac) \vdash (q_1,\lambda,aaac)$ en $q_1 \in F$

Het woord aaa kan niet door A met lege stapel worden geaccepteerd, want het bodemstapelsymbool c kan alleen van de stapel worden verwijderd met gebruikmaking van een symbool b in de invoer. Dus geldt
$a^3 \in L_F(A) \setminus L_\Lambda(A)$.

Het woord aabbb wordt door A met lege stapel geaccepteerd, immers

$(q_0,aabbb,c) \vdash (q_0,abbb,ac) \vdash (q_1,bbb,aac) \vdash (q_1,bb,ac) \vdash$
$(q_1,b,c) \vdash (q_2,\lambda,\lambda)$.

Het is eenvoudig in te zien, dat a^2b^3 niet met accepterende toestanden door A kan worden geaccepteerd. Dus geldt $a^2b^3 \in L_\Lambda(A) \setminus L_F(A)$. Eenvoudig is aan te tonen dat $L_\Lambda(A) = \{a^n b^{n+1} \mid n \geq 1\}$ en dat $L_F(A) = \{a^n b^m \mid n \geq 1, 0 \leq m \leq n\}$. •

In de nu volgende voorbeelden van het ontwerpen van stapelautomaten en het aantonen van de correctheid van het ontwerp, wordt de spiegelingsoperatie gebruikt. Voor elk woord $w = \alpha_1\alpha_2\ldots\alpha_k$ over Σ met $\alpha_i \in \Sigma$ $(1 \leq i \leq k)$ is het spiegelbeeld w^s van w gedefinieerd door $w^s = \alpha_k\ldots\alpha_2\alpha_1$. Merk op dat voor alle $w_1,w_2 \in \Sigma^*$ geldt $(w_1w_2)^s = w_2^s w_1^s$. Verder spreken we af $\lambda^s = \lambda$.

6.11. Voorbeeld

Gevraagd wordt een stapelautomaat te ontwerpen die de taal $L = \{wcw^s \mid w \in \{a,b\}^+\}$ accepteert, in de zin van accepteren met accepterende toestanden. Om tot een ontwerp te komen, omschrijven we in intuitieve termen die berekening van de stapelautomaat die tot acceptatie leidt van een woord uit L:

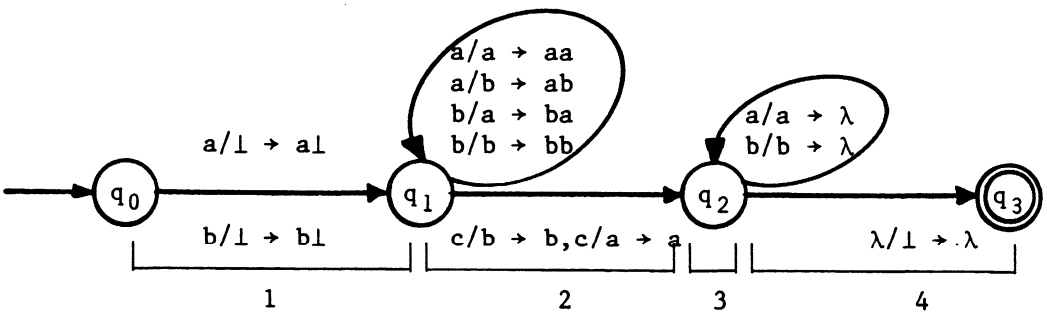
(1) Schrijf alle a's en b's vanuit de invoer op de stapel totdat de c verschijnt. Dan is w gelezen. De eerste letter van w staat dan onder op de stapel en de laatste staat bovenaan.
(2) De c wordt gelezen en aan de stapelinhoud wordt niets veranderd. Wel gaat de automaat naar een nieuwe toestand.

(3) Schrap nu de opeenvolgende symbolen van de invoerband en de symbolen op de stapel tegen elkaar weg als ze overeenkomen.

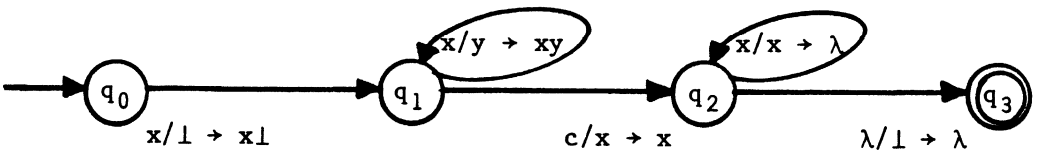
(4) Zodra het bodemstapelsymbool weer topstapelsymbool is, is een woord van de vorm wcw^s gelezen en gaat de automaat naar een accepterende toestand.

Deze beschrijving wordt omgezet in een transitiediagram, waarbij de ontwerper er op moet letten, dat de bedoelde berekening inderdaad mogelijk is volgens dit diagram en dat het transitiediagram geen berekeningen toelaat die tot acceptatie leiden van woorden die niet tot L behoren.

Definieer $A = (\{q_0,q_1,q_2,q_3\}, \{a,b\}, \{a,b,\perp\}, \sigma, q_0, \perp, \{q_3\})$ in overeenstemming met het transitiediagram



Dit transitiediagram is nogal overladen met tak-etiketten. Daarom wordt het ook wel verkort genoteerd aldus:



$x \in \{a,b\}$ en $y \in \{a,b\}$

Nu het ontwerp volledig is, moet nog worden bewezen dat inderdaad geldt $L_F(A) = L$. Om dit aan te tonen kunnen dezelfde methoden worden gehanteerd, die worden gebruikt om aan te tonen dat voor een grammatica G geldt $L(G) = L$. Daarbij komt inductie naar het aantal rekenstappen in de plaats van inductie naar het aantal afleidingsstappen.

De methode van de invarianten is eveneens van toepassing. Doordat een berekening doorgaans is opgebouwd als een rij achtereenvolgens uit te

voeren deelberekeningen, zal de methode over het algemeen alleen handig zijn voor deze deelberekeningen en niet zozeer voor de gehele rij. In het huidige voorbeeld kan het bewijs aldus verlopen

I $L \subseteq L_F(A)$

1. Toon met inductie naar $|w|$ aan, dat $(q_1,w,\gamma\xi) \vdash^* (q_1,\lambda,w^s\gamma\xi)$.
2. Toon aan, ook met inductie naar $|w|$, dat $(q_2,w,w\xi) \vdash^* (q_2,\lambda,\xi)$.
3. Zij $w \in L$. Dan is er een $x \in \{a,b\}$ en een $v_1 \in \{a,b\}^*$ zodat $w = vcv^s$ en is dus de volgende berekening mogelijk

$$\begin{aligned}(q_0,xv_1c(xv_1)^s,\perp) &\vdash (q_1,v_1cv_1^sx,x\perp)\\ &\vdash^* (q_1,cv_1^sx,v_1^sx\perp) \quad \text{(sub 1)}\\ &\vdash (q_2,v_1^sx,v_1^sx\perp)\\ &\vdash^* (q_2,\lambda,\perp) \quad \text{(sub 2)}\\ &\vdash (q_3,\lambda,\lambda)\end{aligned}$$

Dus $w \in L_F(A)$, want q_3 is een accepterende toestand.

II $L_F(A) \subseteq L$

Elke accepterende berekening begint met toestand q_0 en eindigt met toestand q_3. Het invoerwoord moet daarom geschreven kunnen worden als $w = xv_1cv_2$ met $x \in \{a,b\}$ en $v_1,v_2 \in \{a,b\}^*$; anders kan q_0 niet verlaten worden of kan de overgang van q_1 naar q_2 niet plaatsvinden. Zo'n accepterende berekening heeft dus de volgende structuur

$$\begin{aligned}(q_0,xv_1cv_2,\perp) &\vdash (q_1,v_1cv_2,x\perp)\\ &\vdash^* (q_1,cv_2,\xi\perp)\\ &\vdash (q_2,v_2,\xi\perp)\\ &\vdash^* (q_2,\lambda,\perp)\\ &\vdash (q_3,\lambda,\lambda).\end{aligned}$$

Toon nu met inductie naar de lengte van de berekening aan dat uit $(q_1,w,u) \vdash^* (q_1,\lambda,\xi)$ volgt $\xi = w^su$ en dat uit $(q_2,w,u) \vdash^* (q_2,\lambda,v)$ volgt $u = wv$.

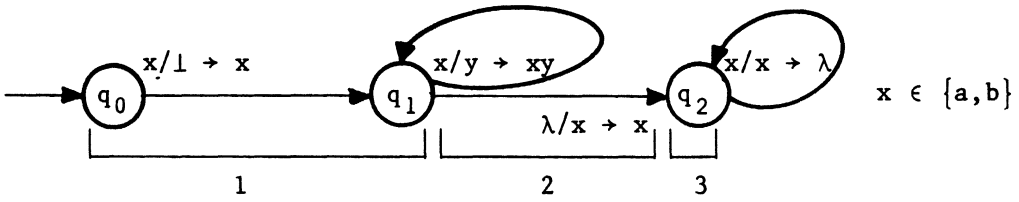
Samenvattend volgt uit $w \in L_F(A)$ dat er een $x \in \{a,b\}$ en $v_1,v_2 \in \{a,b\}^*$ bestaan zodanig dat $w = xv_1cv_2$ en $v_2 = v_1^sx = (xv_1)^s$. Met andere woorden $w \in L$. •

6.12. Voorbeeld

Gevraagd wordt een stapelautomaat te ontwerpen die de taal $L = \{ww^s \mid w \in \{a,b\}^+\}$ accepteert met lege stapel. Het verschil met het voorgaande voorbeeld is, dat nu niet locaal bepaald kan worden of een zekere letter behoort tot het deel w dan wel het deel w^s. Een stapelautomaat voor L zou aldus kunnen werken

1. Zet de a's en de b's uit de invoer op de stapel.
2. Gok dat het deel w nu voorbij is en ga zonder invoer te verbruiken en zonder de stapel te veranderen naar een nieuwe toestand.
3. Schrap overeenkomstige letters van de invoerband en de stapel tegen elkaar weg.

Definieer daarom $A = (\{q_0,q_1,q_2\}, \{a,b\}, \{a,b,\perp\}, \sigma, q_0, \perp, \emptyset)$ in overeenstemming met het transitiediagram:



Het bewijs dat inderdaad $L_\Lambda(A) = L$ verloopt net zo als in voorbeeld 6.11.

De term "gok" in 2 moet niet te letterlijk worden opgevat. Wat betreft de werking drukt deze term uit dat in een tot acceptatie leidende berekening op het passende moment de overgang van q_2 naar q_3 moet worden gemaakt. De stapelautomaat gokt echter niet. Deze bepaalt alleen een verzameling berekeningen. Indien onder de berekeningen die bij een bepaald invoerwoord behoren er tenminste één is die tot acceptatie leidt (volgens de definitie "accepterende toestanden" dan wel "lege stapel", al naar gelang de vraag) behoort het betreffende invoerwoord tot de taal geaccepteerd door de stapelautomaat. •

In de eigenschappen 6.13. en 6.14. zal worden aangetoond dat de klasse van talen, die door stapelautomaten met accepterende toestanden worden

geaccepteerd, gelijk is aan de klasse van talen, die door stapelautomaten met lege stapel worden geaccepteerd.

6.13. Eigenschap

Bij elke stapelautomaat A is een stapelautomaat A' te construeren, zodanig dat $L_F(A') = L_\Lambda(A)$.

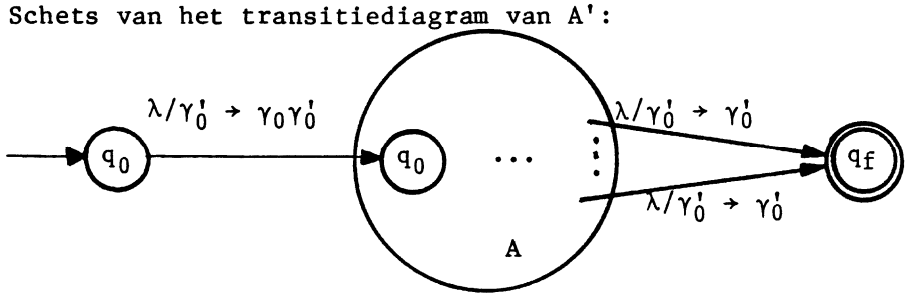
Bewijs

Zij $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ en definieer de stapelautomaat A' aldus: $A' = (Q \cup \{q_0',q_f\},\Sigma,\Gamma \cup \{\gamma_0'\},\sigma',q_0',\gamma_0',\{q_f\})$, waarin q_0' en q_f nieuwe toestanden zijn en γ_0' een nieuw stapelsymbool is. De transitiefunctie σ' wordt gegeven door

(1) $\sigma'(q_0',\lambda,\gamma_0') = \{(q_0,\gamma_0\gamma_0')\}$
(2) $\forall q \in Q \forall a \in \Sigma \cup \{\lambda\} \forall \gamma \in \Gamma[\sigma'(q,a,\gamma) = \sigma(q,a,\gamma)]$.
(3) $\forall q \in Q[\sigma'(q,\lambda,\gamma_0') = \{(q_f,\gamma_0')\}]$

De bedoelde werking van A' is als volgt:
vanuit de begintoestand q_0' van A' vindt zonder verwerking van een invoersymbool een transitie plaats naar de begintoestand q_0 van A, waarbij het startsymbool γ_0 van A op de stapel wordt geschreven, zodat de stapelinhoud $\gamma_0\gamma_0'$ is. Zolang het topstapelsymbool ongelijk γ_0' is, voert A' dezelfde rekenstappen uit, als A zou uitvoeren. Als het topstapelsymbool gelijk aan γ_0' is, is nog de λ-transitie naar de accepterende q_f mogelijk.

Schets van het transitiediagram van A':



Aangetoond moet worden dat $L_F(A') = L_\Lambda(A)$.

(1) Zij $w \in L_\Lambda(A)$, dan $\exists q \in Q[(q_0,w,\gamma_0) \overset{*}{\underset{A}{\vdash}} (q,\lambda,\lambda)]$. Dus dan $(q_0',w,\gamma_0') \underset{A'}{\vdash} (q_0,w,\gamma_0\gamma_0') \overset{*}{\underset{A'}{\vdash}} (q,\lambda,\gamma_0') \underset{A'}{\vdash} (q_f,\lambda,\gamma_0')$.

Aangezien q_f een accepterende toestand van A' is, is $w \in L_F(A')$. Derhalve is $L_\Lambda(A) \subseteq L_F(A')$.

(2) Zij $w \in L_F(A')$, dan
$\exists q \in Q[(q_0',w,\gamma_0') \underset{A'}{\vdash} (q_0,w,\gamma_0\gamma_0') \overset{*}{\underset{A'}{\vdash}} (q,\lambda,\gamma_0') \vdash (q_f,\lambda,\gamma_0')]$. Dus $(q_0,w,\gamma_0) \overset{*}{\underset{A'}{\vdash}} (q,\lambda,\lambda)$ en derhalve ook $(q_0,w,\gamma_0) \overset{*}{\underset{A}{\vdash}} (q,\lambda,\lambda)$ zodat $w \in L_\Lambda(A)$. Derhalve is $L_F(A') \subseteq L_\Lambda(A)$.

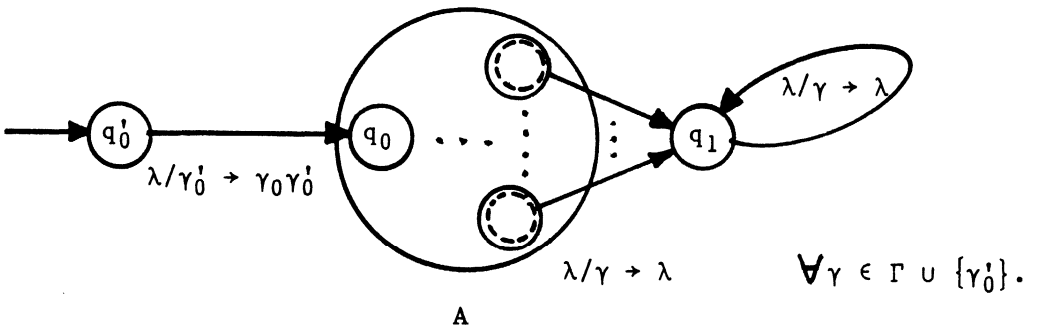
Uit (1) en (2) volgt hetgeen te bewijzen was.
Einde bewijs

6.14. Eigenschap

Bij elke stapelautomaat A is een stapelautomaat A' te construeren, zodanig dat $L_\Lambda(A') = L_F(A)$.

Bewijs

Zij $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$. Kies twee nieuwe toestanden $q_0',q_1 \notin Q$, een nieuw stapelsymbool $\gamma_0' \notin \Gamma$ en definieer de transitiefunctie van $A' = (Q \cup \{q_0',q_1\}, \Sigma, \Gamma \cup \{\gamma_0'\}, \sigma', q_0', \gamma_0', \emptyset)$ in overeenstemming met de volgende schets van het transitiediagram.



A

Dus σ' wordt aldus gedefinieerd:

$$\sigma'(q_0',\lambda,\gamma_0') = \{(q_0,\gamma_0\gamma_0')\}$$
$$\forall\gamma \in \Gamma\cup\{\gamma_0'\}[\sigma'(q_1,\lambda,\gamma) = \{(q_1,\lambda)\}]$$
$$\forall q \in Q\ \forall a \in \Sigma\ \forall\gamma \in \Gamma[\sigma'(q,a,\gamma) = \sigma(q,a,\gamma)]$$
$$\forall q \in Q\ \forall\gamma \in \Gamma[\sigma'(q,\lambda,\gamma) = \underline{\text{als}}\ q \notin F\ \underline{\text{dan}}\ \sigma(q,\lambda,\gamma)\ \underline{\text{anders}}\ \sigma(q,\lambda,\gamma) \cup \{(q_1,\lambda)\}]$$
$$\forall q \in F[\sigma'(q,\lambda,\gamma_0') = \{(q_1,\lambda)\}].$$

De werking van A' is kort beschreven de volgende: vanuit de begintoestand q_0' van A' vindt zonder verwerking van een invoersymbool een transitie plaats naar de begintoestand q_0 van A, waarbij het startsymbool γ_0 van A op de stapel wordt geschreven, zodat de stapel het woord $\gamma_0\gamma_0'$ bevat. Vanaf dit punt gedraagt A' zich als A eventueel totdat alle invoer verwerkt is en A in een toestand $q \in F$ verkeert. Als deze situatie optreedt, kan er een transitie plaatsvinden naar toestand q_1 van A' en kan de stapel geleegd worden.

Toon zelf aan dat $L_\Lambda(A') = L_F(A)$.
Einde bewijs

6.15. Oefeningen

(1) Ontwerp een stapelautomaat A zodanig dat geldt $L_\Lambda(A) = \{w \in \{a,b\}^+ \mid \#(a,w) = 2 \times \#(b,w)\}$ en tegelijk ook geldt $L_F(A) = \emptyset$.

(2) Ontwerp een stapelautomaat A zodanig dat geldt $L_\Lambda(A) = \emptyset$ en $L_F(A) = \{ucv \mid u,v \in \{a,b\}^+ \text{ en } \#(a,u) = \#(b,v)\}$.

(3) Ontwerp een stapelautomaat A zodanig dat geldt $L_\Lambda(A) = L_F(A) = \{w \in \{a,b,c\}^+ \mid \#(a,w) + \#(b,w) = \#(c,w)\}$.

(4) Bewijs dat er voor elke stapelautomaat A, stapelautomaten B en C bestaan zodanig dat
(1) $L_\Lambda(B) = L_F(B) = L_\Lambda(A)$.
(2) $L_\Lambda(C) = L_F(C) = L_F(A)$.

6.16. Eigenschap

Als L een contextvrije taal is, dan bestaat er een stapelautomaat A zodanig dat $L = L_\Lambda(A)$.

Bewijs

Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica die L voortbrengt. Definieer de stapelautomaat A aldus: $A = (\{q\},\Sigma,V,\sigma,q,S,\emptyset)$, waarin de transitiefunctie σ gegeven wordt door

(1) $\forall X \in V\setminus\Sigma[\sigma(q,\lambda,X) = \{(q,\alpha) \mid X \to \alpha \in P\}]$
(2) $\forall a \in \Sigma[\sigma(q,a,a) = \{(q,\lambda)\}]$

De berekening van stapelautomaat A op invoer w bestaat uit de simulatie van een linkerafleiding van w volgens G. Om te bewijzen dat $L(G) = L_\Lambda(A)$ wordt de volgende hulpeigenschap gebruikt:

(*) $\forall X \in V\ \forall w \in \Sigma^*[X \overset{*}{\Rightarrow} w$ dan en slechts dan als $(q,w,X) \overset{*}{\vdash} (q,\lambda,\lambda)]$

Toon dit zelf aan met inductie naar de lengte van de afleiding volgens G respectievelijk het aantal rekenstappen van A.

(1) Zij $w \in L(G)$, dan $S \overset{*}{\underset{G}{\Rightarrow}} w$. Uit (*) volgt dat $(q,w,S) \overset{*}{\underset{A}{\vdash}} (q,\lambda,\lambda)$, zodat $w \in L_\Lambda(A)$.
Derhalve $L(G) \subseteq L_\Lambda(A)$.

(2) Zij $w \in L_\Lambda(A)$, dan $(q,w,S) \overset{*}{\underset{A}{\vdash}} (q,\lambda,\lambda)$. Uit (*) volgt dat $S \overset{*}{\Rightarrow} w$, zodat $w \in L(G)$. Derhalve $L_\Lambda(A) \subseteq L(G)$.

Uit (1) en (2) volgt hetgeen te bewijzen was.
Einde bewijs

6.17. Oefening

Zij $G = (\{S,A,a,b\}, \{a,b\}, P, S)$ met de produktieregels

$S \to aAA$ $\qquad$ $A \to aS$
$A \to bS$ $\qquad$ $A \to a$

Construeer een stapelautomaat A zodanig dat $L(G) = L_\Lambda(A)$. Vergelijk voor de woorden abaaa en aabaaa de linkerafleidingen volgens G met de berekening volgens A.

6.18. Eigenschap

Als A een stapelautomaat is, dan is de taal $L_\Lambda(A)$ contextvrij.

Bewijs

Zij $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$. Definieer de contextvrije grammatica G aldus: $G = (V,\Sigma,P,S)$ met $V = \Sigma \cup \{S\} \cup \{\langle q,\gamma,p\rangle \mid q,p \in Q,\ \gamma \in \Gamma\}$ en P bevat de produktieregels

(1) $\forall q \in Q$ de regel $S \to \langle q_0,\gamma_0,q\rangle$

(2) $\forall q,q_1,\ldots,q_{m+1} \in Q\, \forall a \in \Sigma \cup \{\lambda\}\, \forall \gamma,\gamma_1,\ldots,\gamma_m \in \Gamma$ zodanig dat $(q_1,\gamma_1\ldots\gamma_m) \in \sigma(q,a,\gamma)$ de regel
$\langle q,\gamma,q_{m+1}\rangle \to a\langle q_1,\gamma_1,q_2\rangle\langle q_2,\gamma_2,q_3\rangle\ldots\langle q_m,\gamma_m,q_{m+1}\rangle$.

(3) $\forall q_1,q_2 \in Q\forall a \in \Sigma \cup \{\lambda\}\forall \gamma \in \Gamma$ zodanig dat $(q_2,\lambda) \in \sigma(q_1,a,\gamma)$ de regel
$\langle q_1,\gamma,q_2\rangle \to a$.

De dingen $\langle p,\gamma,q\rangle$ zijn hulpsymbolen, en géén rijtjes. De notatie als een rijtje is gekozen omdat daarmee eenvoudig is aan te geven, hoe de produktieregels zijn opgebouwd. Om te bewijzen dat $L(G) = L_\Lambda(A)$, wordt de volgende hulpeigenschap gebruikt:

(*) $\forall p,q \in Q\, \forall \gamma \in \Gamma\, \forall w \in \Sigma^*[\langle p,\gamma,q\rangle \overset{*}{\underset{G}{\Rightarrow}} w$ dan en slechts dan als $(p,w,\gamma) \overset{*}{\underset{A}{\vdash}} (q,\lambda,\lambda)]$.

Toon dit zelf aan met inductie naar de lengte van de afleiding volgens G respectievelijk het aantal rekenstappen van A.

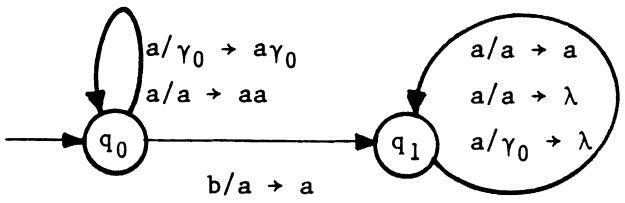
(1) Zij $w \in L(G)$, dan $\exists q \in Q[\langle q_0,\gamma_0,q\rangle \overset{*}{\underset{G}{\Rightarrow}} w]$. Uit (*) volgt dat $(q_0,w,\gamma_0) \overset{*}{\underset{A}{\vdash}} (q,\lambda,\lambda)$ zodat $w \in L_\Lambda(A)$. Derhalve is $L(G) \subseteq L_\Lambda(A)$.

(2) Zij $w \in L_\Lambda(A)$ dan $(q_0,w,\gamma_0) \overset{*}{\underset{A}{\vdash}} (q,\lambda,\lambda)$. Uit (*) volgt dat $<q_0,\gamma_0,q> \overset{*}{\underset{G}{\Rightarrow}} w$, zodat $w \in L(G)$. Derhalve is $L_\Lambda(A) \subseteq L(G)$.

Uit (1) en (2) volgt $L(G) = L_\Lambda(A)$.
Einde bewijs

6.19. Voorbeeld

Zij $A = (\{q_0,q_1\}, \{a,b\}, \{\gamma_0,a,b\}, \sigma, q_0, \gamma_0, \emptyset)$ de stapelautomaat bepaald door onderstaand transitiediagram:

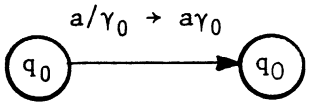


Er geldt $L_\Lambda(A) = \{a^n b a^m \mid m > n > 0\}$. In het navolgende wordt een contextvrije grammatica $G = (V,\Sigma,P,S)$ geconstrueerd zodanig dat $L(G) = L_\Lambda(A)$.
Volgens het bewijs van eigenschap 6.18. worden de volgende hulpsymbolen geintroduceerd.

$<q_0,\gamma_0,q_0>$
$<q_0,\gamma_0,q_1>$
$<q_1,\gamma_0,q_0>$
$<q_1,\gamma_0,q_1>$
$<q_0,a,q_0>$
⋮
$<q_1,b,q_1>$

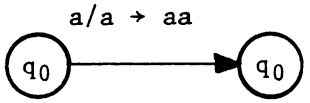
De verzameling P van produktieregels wordt als volgt verkregen:

$S \rightarrow <q_0,\gamma_0,q_0>$
$S \rightarrow <q_0,\gamma_0,q_1>$

Uit de transitie 

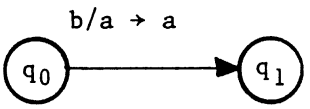
volgen de regels

$\langle q_0,\gamma_0,q_0\rangle \rightarrow a\langle q_0,a,q_0\rangle\langle q_0,\gamma_0,q_0\rangle$
$\langle q_0,\gamma_0,q_1\rangle \rightarrow a\langle q_0,a,q_0\rangle\langle q_0,\gamma_0,q_1\rangle$
$\langle q_0,\gamma_0,q_0\rangle \rightarrow a\langle q_0,a,q_1\rangle\langle q_1,\gamma_0,q_0\rangle$
$\langle q_0,\gamma_0,q_1\rangle \rightarrow a\langle q_0,a,q_1\rangle\langle q_1,\gamma_0,q_1\rangle$

Uit de transitie 

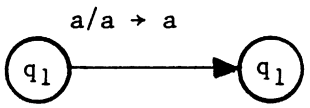
volgen de regels

$\langle q_0,a,q_0\rangle \rightarrow a\langle q_0,a,q_0\rangle\langle q_0,a,q_0\rangle$
$\langle q_0,a,q_1\rangle \rightarrow a\langle q_0,a,q_0\rangle\langle q_0,a,q_1\rangle$
$\langle q_0,a,q_0\rangle \rightarrow a\langle q_0,a,q_1\rangle\langle q_1,a,q_0\rangle$
$\langle q_0,a,q_1\rangle \rightarrow a\langle q_0,a,q_1\rangle\langle q_1,a,q_1\rangle$

Uit de transitie 

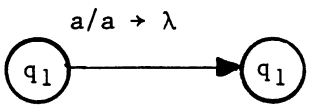
volgen de regels

$\langle q_0,a,q_0\rangle \rightarrow b\langle q_1,a,q_0\rangle$
$\langle q_0,a,q_1\rangle \rightarrow b\langle q_1,a,q_1\rangle$

Uit de transitie 

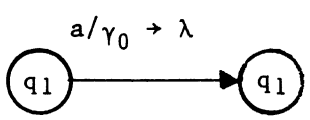
volgen de regels

$\langle q_1,a,q_0\rangle \rightarrow a\langle q_1,a,q_0\rangle$
$\langle q_1,a,q_1\rangle \rightarrow a\langle q_1,a,q_1\rangle$

Uit de transitie 

volgt de regel

$\langle q_1,a,q_1\rangle \rightarrow a$

Uit de transitie 

volgt de regel

$\langle q_1,\gamma_0,q_1\rangle \rightarrow a$

Het woord abaa wordt door A als volgt geaccepteerd:

$$(q_0,abaa,\gamma_0)\vdash(q_0,baa,a\gamma_0)\vdash(q_1,aa,a\gamma_0)\vdash(q_1,a,\gamma_0)\vdash(q_1,\lambda,\lambda)$$

en door G als volgt voortgebracht:

$$\begin{aligned} S \Rightarrow &<q_0,\gamma_0,q_1> \Rightarrow a<q_0,a,q_1><q_1,\gamma_0,q_1> \Rightarrow \\ &ab<q_1,a,q_1><q_1,\gamma_0,q_1> \Rightarrow \\ &aba<q_1,\gamma_0,q_1> \Rightarrow abaa \end{aligned}$$ •

6.20. Oefening

Beschouw de in voorbeeld 6.19. geconstrueerde grammatica G. Het merendeel van de geintroduceerde hulpsymbolen en productieregels is overbodig. Construeer een met G equivalente grammatica G' met minder hulpsymbolen en minder productieregels.

Uit het voorgaande is al gebleken dat een stapelautomaat in het algemeen een niet-deterministische automaat is. In de volgende definitie worden de voorwaarden aangegeven opdat een stapelautomaat deterministisch is.

6.21. Definitie: Een stapelautomaat $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ is deterministisch als aan de volgende voorwaarden is voldaan:

(1) $\forall q \in Q \forall \gamma \in \Gamma[\underline{\text{als}}\ \sigma(q,\lambda,\gamma) \neq \emptyset\ \underline{\text{dan}} \forall a \in \Sigma[\sigma(q,a,\gamma) = \emptyset]]$
(2) $\forall q \in Q \forall \gamma \in \Gamma \forall a \in \Sigma \cup \{\lambda\}[\text{card}(\sigma(q,a,\gamma)) \leq 1]$

6.22. Voorbeeld

De stapelautomaat A uit voorbeeld 6.12. is een niet-deterministische stapelautomaat aangezien niet aan voorwaarde (1) van definitie 6.21. is voldaan. Toch bestaat er voor een geaccepteerd woord precies één, eenduidig bepaalde accepterende berekening. Er bestaan echter vele niet-accepterende berekeningen. De stapelautomaat uit voorbeeld 6.11. daarentegen is een deterministische stapelautomaat. •

In hoofdstuk 4 is aangetoond dat deterministische en niet-deterministische eindige automaten hetzelfde herkennend vermogen hebben. Voor stapelautomaten is dit niet het geval. Er zijn talen die wel door een niet-deterministische stapelautomaat kunnen worden geaccepteerd, maar niet door een deterministische stapelautomaat. Een voorbeeld van een dergelijke taal is de taal $\{ww^s \mid w \in \{a,b\}^+\}$.

Er zijn verschillende technieken om aan te tonen dat talen niet door deterministische stapelautomaten kunnen worden geaccepteerd. Er is bijvoorbeeld een pompstelling voor talen die door deterministische stapelautomaten kunnen worden geaccepteerd. Ook onderscheidt deze klasse van talen zich door zijn afsluitingseigenschappen.

We zullen dit illustreren voor de operatie MIN.

6.23. Definitie: Voor elke taal L over een alfabet Σ wordt de taal MIN(L) gedefinieerd door:
$MIN(L) = \{w \mid w \in L \text{ en } \neg(\exists u \in \Sigma^* \exists v \in \Sigma^+[u \in L \text{ en } uv = w])\}$.

De taal MIN(L) bestaat dan uit die woorden uit L die geen echte prefix hebben die eveneens tot L behoort. Dus $MIN(\{a^nb^m \mid m \geq n \geq 0\}) = \{\lambda\}$ en $MIN(\{a^nb^m \mid m \geq n \geq 1\}) = \{a^nb^n \mid n \geq 1\}$.

6.24. Eigenschap

Voor elke deterministische stapelautomaat $A=(Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ bestaat er een deterministische stapelautomaat A_M zo dat $L_F(A_M)=MIN(L_F(A))$.

Bewijs
Definieer $A_M = (Q,\Sigma,\Gamma,\sigma_M,q_0,\gamma_b,F)$ aldus:

$\forall q \in Q \forall a \in \Sigma \cup \{\lambda\} \forall \gamma \in \Gamma[\ \sigma_M(q,a,\gamma) = \underline{\text{als}}\ q \in F\ \underline{\text{dan}}\ \emptyset\ \underline{\text{anders}}\ \sigma(q,a,\gamma)\ ]$

Dan geldt $MIN(L_F(A)) = L_F(A_M)$.

- Zij $w \in MIN(L_F(A))$.
  Dan is er een berekening $(q_0,w,\gamma_0) \vdash (q_1,w_1,\alpha_1) \vdash \cdots \vdash (q_r,\lambda,\alpha_r)$, $r \geq 0$ met $q_r \in F$ en $q_0,\ldots,q_{r-1} \notin F$. Deze berekening is dus

ook een berekening van A_M zodat $w \in L_F(A_M)$ en dus $MIN(L_F(A)) \subseteq L_F(A_M)$. Als $w = \lambda$ dan is $r = 0$ en heeft de berekening lengte 0. Maar ook deze berekening is een berekening van A_M.

- Zij $w \in L_F(A_M)$.
  Dan is er een berekening $(q_0,w,\gamma_0) \vdash (q_1,w_1,\alpha_1) \vdash \cdots \vdash (q_r,\lambda,\alpha_r)$ met $q_r \in F$ en $q_0,\ldots,q_{r-1} \notin F$.
  Omdat $\forall q \in Q \forall a \in \Sigma \cup \{\lambda\} \forall \gamma \in \Gamma[\sigma_M(q,a,\gamma) \subseteq \sigma(q,a,\gamma)]$ is dit ook een berekening van A, zodat $w \in L_F(A)$. Daar A deterministisch is volgt dat elke berekening die begint met (q_0,w,γ_0) bovenstaande berekening als beginstuk heeft (Er kunnen nog λ-stappen volgen). Dus kan w geen echte prefix hebben, die ook tot $L_F(A)$ behoort, zodat $w \in MIN(L_F(A))$ en dus $L_F(A_M) \subseteq MIN(L_F(A))$.

Einde bewijs

De klasse van talen die geaccepteerd kunnen worden door deterministische stapelautomaten, is dus gesloten met betrekking tot de bewerking MIN.

We zullen nu aantonen dat $MIN(\{ww^s \mid w \in \{a,b\}^+\})$ niet contextvrij is. Daaruit volgt:

(1) De klasse $\mathcal{L}_2 = \mathcal{L}_{NPDA}$ is niet gesloten met betrekking tot MIN. Immers $\{ww^s \mid w \in \{a,b\}^+\} \in \mathcal{L}_{NPDA}$ (zie voorbeeld 6.12.) terwijl $MIN(\{ww^s \mid w \in \{a,b\}^+\})$ niet contextvrij is en dus niet behoort tot $\mathcal{L}_{NPDA}$.

(2) $\mathcal{L}_{DPDA} \subset \mathcal{L}_{NPDA}$ en $\{ww^s \mid w \in \{a,b\}^+\}$ is een taal die behoort tot $\mathcal{L}_{NPDA} \setminus \mathcal{L}_{DPDA}$. Immers als $\{ww^s \mid w \in \{a,b\}^+\}$ tot $\mathcal{L}_{DPDA}$ behoort, dan kan ook $MIN(\{ww^s \mid w \in \{a,b\}^+\})$ door een deterministische stapelautomaat worden geaccepteerd en is dus contextvrij. Dus $\{ww^s \mid w \in \{a,b\}^+\} \in \mathcal{L}_{NPDA} \setminus \mathcal{L}_{DPDA}$.

Om te bewijzen dat $MIN(\{ww^s \mid w \in \{a,b\}^+\})$ niet contextvrij is, maken we gebruik van de volgende eigenschap:

"Voor elke $L \in \mathcal{L}_2$ en elke $R \in \mathcal{L}_3$ geldt $L \cap R \in \mathcal{L}_2$"
(De eigenschap wordt bewezen in hoofdstuk 7).

Zij R de reguliere taal $\{(ab)^p(ba)^q(ab)^r(ba)^s \mid p,q,r,s \geq 1\}$.

$$MIN(\{ww^s \mid w \in \{a,b\}^+\}) \cap R = \{(ab)^n(ba)^m(ab)^m(ba)^n \mid n > m > 0\}.$$

Toon nu zelf aan dat $\{(ab)^n(ba)^m(ab)^m(ba)^n \mid n > m > 0\}$ niet contextvrij is (met behulp van de pompstelling voor contextvrije talen).

6.25. Oefeningen

(1) Ontwerp een stapelautomaat A zodanig dat $L_\Lambda(A) = \{w \in \{a,b\}^* \mid \#(a,w) = 2 \times \#(b,w)\}$. Is de door u ontworpen automaat deterministisch?

(2) Construeer een stapelautomaat A zodanig dat $L_F(A)=\{a^ib^jc^k \mid i \neq j \text{ of } j \neq k\}$. Is de door u geconstrueerde automaat deterministisch?

(3) Toon aan dat de taal $L = \{a^nb^n \mid n \geq 1\} \cup \{a^nb^{2n} \mid n \geq 1\}$ wel door een niet-deterministische en niet door een deterministische stapelautomaat kan worden geaccepteerd.

Een aantal resultaten wordt nu samengevat en enigszins uitgebreid in de volgende

6.26. Eigenschap

(1) $\mathcal{L}_3 \subset \mathcal{L}_{DPDA} \subset \mathcal{L}_{NPDA} = \mathcal{L}_2$

(2) $\mathcal{L}_3 \subset \mathcal{L}_{lin} \subset \mathcal{L}_{NPDA} = \mathcal{L}_2$

(3) De klasse $\mathcal{L}_{lin}$ en $\mathcal{L}_{DPDA}$ zijn onvergelijkbaar: er bestaan talen L_1 en L_2 zodanig dat

$$L_1 \in \mathcal{L}_{lin} \setminus \mathcal{L}_{DPDA}, \text{ en}$$
$$L_2 \in \mathcal{L}_{DPDA} \setminus \mathcal{L}_{lin}.$$

Bewijs

Uit de eigenschappen 6.20. en 6.22. volgt de gelijkheid $\mathcal{L}_{NPDA} = \mathcal{L}_2$, terwijl de inclusies $\mathcal{L}_3 \subseteq \mathcal{L}_{DPDA} \subseteq \mathcal{L}_{NPDA}$ en $\mathcal{L}_3 \subseteq \mathcal{L}_{lin} \subseteq \mathcal{L}_2$ een (bijna) onmiddellijk gevolg zijn van de definities van deze klassen.

Blijft nog over aan te tonen dat de verschillende inclusies echt zijn. We geven hier de voorbeelden waaruit die echtheid blijkt

$\{a^n b^n \mid n > 1\} \in \mathcal{L}_{DPDA} \setminus \mathcal{L}_3$
$\{ww^s \mid w \in \{a,b\}^+\} \in \mathcal{L}_2 \setminus \mathcal{L}_{DPDA}$
$\{a^n b^n \mid n > 1\} \in \mathcal{L}_{lin} \setminus \mathcal{L}_3$
$\{a^n b^n a^m b^m \mid n,m > 1\} \in \mathcal{L}_2 \setminus \mathcal{L}_{lin}$

Dat de klassen $\mathcal{L}_{lin}$ en $\mathcal{L}_{DPDA}$ onvergelijkbaar zijn tenslotte, volgt uit de voorbeelden:

$L_1 = \{ww^s \mid w \in \{a,b\}^+\} \in \mathcal{L}_{lin} \setminus \mathcal{L}_{DPDA}$
$L_2 = \{a^n b^n a^m b^m \mid n,m > 1\} \in \mathcal{L}_{DPDA} \setminus \mathcal{L}_{lin}$

Einde bewijs

Om een algoritme en vervolgens een programma, te maken dat voor zekere $L \in \mathcal{L}_2$ de vragen "is $w \in L$?" beantwoordt, kan ook gebruik gemaakt worden van stapelautomaten. Bij L hoort een stapelautomaat A. Het algoritme gaat bij gegeven w na, welke berekeningen die beginnen met (q_0,w,γ_0) mogelijk zijn en zoekt of er daaronder een is die het gehele invoerwoord w verwerkt en eindigt in een configuratie met een accepterende toestand (respectievelijk in een configuratie met lege stapel). Dit onderzoeken van alle mogelijke berekeningen moet goed georganiseerd worden. Immers, zelfs als een woord w door de stapelautomaat A wordt geaccepteerd, is het mogelijk dat A andere berekeningen kan uitvoeren die niet eindigen (en dus ook niet tot accepteren leiden). Het is mogelijk om uit A en w een bovengrens te berekenen voor de lengte van een berekening die tot accepteren van w leidt, zodat het bepalen of er een berekening is die tot accepteren leidt daadwerkelijk kan worden uitgevoerd. Het onderzoeken van al deze mogelijkheden kan echter zeer tijdrovend (exponentieel in de lengte van de berekening) zijn. In de practijk wordt dan ook alleen gebruik gemaakt van deterministische stapelautomaten zodat het hele zoekprobleem verdwijnt en in $\mathcal{O}(|w|)$ tijd kan worden bepaald of $w \in L$.

Dit is niet voor alle contextvrije talen mogelijk. We hebben al gezien dat bijvoorbeeld $\{ww^s \mid w \in \{a,b\}^+\}$ niet met een deterministische stapelautomaat kan worden herkend.

Uit het voorgaande kan de indruk ontstaan dat voor sommige contextvrije talen $L \subseteq \Sigma^*$ de oplossing van hun lidmaatschapsprobleem $[w \in \Sigma^*: w \in L]$ een hoeveelheid rekentijd vraagt die exponentieel is in de lengte van w. Dit is echter niet het geval, voor iedere contextvrije $L \subseteq \Sigma^*$ bestaat een algoritme dat het lidmaatschapsprobleem van L oplost en waarvan de rekentijd $\mathcal{O}(|w|^3)$ is.

6.27. Eigenschap

Voor elke contextvrije taal $L \subseteq \Sigma^*$ die niet het lege woord λ bevat, bestaat er een contextvrije grammatica die L voortbrengt en in Chomsky normaalvorm is, dat wil zeggen alleen productieregels van de vorm $A \to BC$ of van de vorm $A \to a$, $A,B,C \in V\setminus\Sigma$ en $a \in \Sigma$, bevat.

Bewijs
Oefening
Einde bewijs

6.28. Definitie: Zij $G = (V,\Sigma,P,S)$ een contextvrije grammatica in Chomsky normaalvorm. Definieer een produkt $\otimes$ als volgt
$X \otimes Y = \{A \mid \exists B \in X \exists C \in Y[A \to BC \in P\}$, $X,Y \in P(V\setminus\Sigma)$.

In onderstaand algoritme wordt gebruik gemaakt van dit produkt.

6.29. Algoritme voor $[w \in \Sigma^+: w \in L(G)]$ waarin $G = (V,\Sigma,P,S)$ een contextvrije grammatica in Chomsky normaalvorm is.

invoer : $x_1x_2\ldots x_n \in \Sigma^*$
uitvoer: antw $\in \{0,1\}$ {antw = als $x_1\ldots x_n \in L(G)$ dan 1 anders 0}

```
for i = 1 to n do
    m(i,i):= {A | A → x_i ∈ P}
end;
    {Definieer P(i,j) = [m(i,j+1) = {A | a ⇒* x_i..x_{i+j}}]}
    {∀x ∈ [1,n]P(x,0)}
for j = 1 to n-1 do
    {∀y ∈ [0,j-1]∀x ∈ [1,n-y] P(x,y)}
    for j = 1 to n-1 do
        {∀y ∈ [0,j-1]∀x ∈ [1,n-y] P(x,y) ∧ ∀z ∈ [1,i-1] P(z,j)}
        m(i,i+j):= ⋃_{k=0}^{j-1} m(i,i+k) ⊗ m(i+k+1,i+j);
        {∀y ∈ [0,j-1]∀x ∈ [1,n-y] P(x,y)∧∀z ∈ [1,i-1] P(z,j)∧P(i,j)}
        {∀y ∈ [0,j-1]∀x ∈ [1,n-y] P(x,y)∧∀z ∈ [1,i] P(z,j)}
    end
    {∀y ∈ [0,j-1]∀x ∈ [1,n-y] P(x,y) ∧ ∀z ∈ [1,n-j] P(z,j)}
    {∀y ∈ [0,j] ∀x ∈ [1,n-y] P(x,y)}
end;
    {∀y ∈ [0,n-1] ∀x ∈ [1,n-y] P(x,y)}
    {P(1,n-1) dus m(1,n) = {A | A ⇒* x_1...x_n}}
antw:= if S ∈ m(1,n) then 1 else 0;
```

In algoritme 6.29. geven de uitspraken tussen de scheidingstekens { en } de (in)varianten die geldig zijn. Na afloop geldt $m(1,n) = \{A \mid A \overset{*}{\Rightarrow} x_1 \ldots x_n\}$ zodat antw = 1 als en alleen als $x_1 \ldots x_n \in L(G)$ en het algoritme derhalve correct is.

Wat de hoeveelheid rekentijd betreft:

(1) De verzamelingen V_a, voor elke $a \in \Sigma$ gedefinieerd door $V_a = \{A \in V \setminus \Sigma \mid A \to a \in P\}$ zijn onafhankelijk van de invoer en kunnen dus eens voor al berekend worden.

(2) Ook van het produkt ⊗ kan een tabel worden opgesteld zodat de waarde van X ⊗ Y kan worden opgezocht en dus een constante (dat wil zeggen van n onafhankelijke) hoeveelheid tijd vraagt.

(3) Het algoritme gebruikt dus

$$c_1 \cdot n + \sum_{j=1}^{n-1} \sum_{i=1}^{n-j} c_2 \cdot j$$

tijd, met andere woorden de benodigde tijd is $\mathcal{O}(n^3)$.

Algoritme 6.29. vereist een grammatica in Chomsky normaalvorm. Het algoritme kan worden omgewerkt zodat het bruikbaar is voor willekeurige contextvrije grammatica's. Daarvan wordt hier afgezien.

Algoritme 6.29. vraagt veel geheugenruimte ($\mathcal{O}(n^2)$). Ook daarop kan flink worden bezuinigd, maar de verdere uitwerking daarvan voert te ver.

7. OPERATIES OP TALEN

Naast grammatica's, automaten en reguliere expressies kunnen ook operaties gebruikt worden om talen te definiëren: zo levert het toepassen van een operatie f van rang n op een n-tal talen $L_1,\ldots,L_n$ een (mogelijk nieuwe) taal $f(L_1,\ldots,L_n)$. Naast de gebruikelijke verzameling-theoretische operaties als vereniging, doorsnijding en verschil, worden in de formele talentheorie nog een aantal andere operaties beschouwd. Enkele van deze operaties kwamen in vorige hoofdstukken al aan de orde; met name de concatenatie van talen (definitie 5.1.), de iteratie (of Kleene *; definitie 5.3.) en de λ-vrije iteratie (of Kleene +; definitie 5.3.). Ook de volgende operatie werd al terloops in hoofdstuk 6 genoemd.

7.1. Definitie: Zij Σ een alfabet. Als $w = a_1a_2\ldots a_n$, $a_i \in \Sigma$ $(1 \leq i \leq n)$, dan wordt het spiegelbeeld van w, notatie w^s, gedefinieerd door $w^s = a_n\ldots a_2a_1$. Verder is $\lambda^s = \lambda$, en voor elke taal L over Σ wordt het spiegelbeeld L^s van L gedefinieerd door $L^s = \{w^s \mid w \in L\}$.

7.2. Voorbeeld

Voor elke taal L over Σ met $\text{card}(\Sigma) = 1$ geldt $L^s = L$. Zij nu $\Sigma=\{a,b\}$. Het spiegelbeeld van het woord $ab^2a^3 \in \Sigma^*$, is het woord a^3b^2a. Voorts is $(a(ba^2)^3)^s = (a^2b)^3a$. Als $L = \{a(ba)^n \mid n \geq 1\}$, dan is $L^s = \{(ab)^na \mid n \geq 1\}$. Merk op dat $L^s = L$. De taal L bestaat uit zogenaamde palindromen; een palindroom is een woord, dat gelijk is aan zijn eigen spiegelbeeld. •

7.3. Oefeningen

(1) Ga na dat $(w_1^s)^s = w_1$, $(w_1w_2)^s = w_2^sw_1^s$ en $(L_1L_2)^s = L_2^sL_1^s$ voor woorden $w_1,w_2 \in \Sigma^*$ en talen $L_1,L_2 \subseteq \Sigma^*$. Gelden ook de volgende gelijkheden: $(L_1^+)^s = (L_1^s)^+$, $(L_1^*)^s = (L_1^s)^*$, $(L_1 \cup L_2)^s = L_1^s \cup L_2^s$, $(L_1 \cap L_2)^s = L_1^s \cap L_2^s$, en $(L_1L_2)^s = L_1^sL_2^s$?

(2) Ga na of de spiegelingsoperatie een (halfring-)isomorfisme is
- van $(P(\Sigma^*),(\cup,\cap,\emptyset,\Sigma^*))$ in $(P(\Sigma^*),(\cup,\cap,\emptyset,\Sigma^*))$,
- van $(P(\Sigma^*),(\cup,\bullet,\emptyset,\{\lambda\}))$ in $(P(\Sigma^*),(\cup,\bullet,\emptyset,\{\lambda\}))$, waarin $\bullet$ de concatenatieoperatie is.

Bepaal voor die gevallen, waarvoor Uw antwoord bevestigend luidt, de inverse van de spiegelingsoperatie.

(3) Toon aan dat als voor een taal L over Σ geldt, dat $L = L^s$, er niet geconcludeerd mag worden, dat L zuiver uit palindromen bestaat.

7.4. Definitie: Zij Σ en Δ alfabetten. Een afbeelding $h: \Sigma^* \to \Delta^*$ is een homomorfisme als $h(\lambda) = \lambda$ en $h(xy) = h(x)h(y)$ voor alle x en y in Σ^*. Een homomorfisme $h: \Sigma^* \to \Delta^*$ heet λ-vrij als $h(\alpha) \neq \lambda$ voor alle α in Σ.
Elk homomorfisme $h: \Sigma^* \to \Delta^*$ wordt uitgebreid tot een afbeelding $h: P(\Sigma^*) \to P(\Delta^*)$ door middel van $h(L) = \{h(x) \mid x \in L\}$ voor alle $L \subseteq \Sigma^*$. De inverse h^{-1} van h is de afbeelding $h^{-1}: P(\Delta^*) \to P(\Sigma^*)$ gedefinieerd door $h^{-1}(L) = \{x \in \Sigma^* \mid h(x) \in L\}$.

7.5. Voorbeeld

(1) Zij $\Sigma = \{0,1,\ldots,9\}$ en $\Delta = \{0,1\}$. Het homomorfisme $h: \Sigma^* \to \Delta^*$ wordt gedefinieerd door
$h(0) = 0000$; $h(1) = 0001$; $h(2) = 0010$; $h(3) = 0011$; ...; $h(9) = 1001$
De afbeelding h beeldt een decimaal weergegeven getal af op de BCD-code ("Binary Coded Decimal") van dit getal. Dan is
$h(1321) = 0001001100100001$,
$h(\{03^n1^n \mid n \geq 1\}) = \{0000(0011)^n(0001)^n \mid n \geq 1\}$.
Merk op dat h λ-vrij is.

(2) Als het homomorfisme $h: \{a,b\}^* \to \{c\}^*$ gedefinieerd is door $h(a) = cc$ en $h(b) = \lambda$, dan geldt
- voor elke $w \in \{a,b\}^*$, dat $h(w) = c^{2.\#(a,w)}$

- voor elke $L_1 \subseteq \{a,b\}^*$, dat $h(L_1) = \{c^{2.\#(a,w)} \mid w \in L_1\}$

- voor elke $L_2 \subseteq \{c\}^*$, dat $h^{-1}(L_2) = \{w \in \{a,b\}^* \mid c^{2.\#(a,w)} \in L_2\}$●

Een homomorfisme $h: \Sigma^* \to \Delta^*$ vervangt in woorden over Σ de individuele symbolen door woorden over Δ. Dit kan men veralgemenen door afbeeldingen $\sigma: \Sigma^* \to P(\Delta^*)$ te beschouwen, die symbolen in woorden over Σ vervangen door talen over Δ. Een dergelijke functie heet een substitutie.

7.6. Definitie: Zij Σ en Δ alfabetten en $\mathcal{L}$ een klasse van talen. Een $\mathcal{L}$-substitutie σ is een afbeelding $\sigma: \Sigma^* \to P(\Delta^*)$, die voldoet aan:

(1) voor alle α in Σ geldt: $\sigma(\alpha)$ is een taal over Δ behorende tot de klasse $\mathcal{L}$,

(2) $\sigma(\lambda) = \{\lambda\}$,

(3) voor alle x en y in Σ^* geldt: $\sigma(xy) = \sigma(x)\sigma(y)$.

Een $\mathcal{L}$-substitutie $\sigma: \Sigma^* \to P(\Delta^*)$ heet λ-vrij als $\lambda \notin \sigma(\alpha)$ voor alle α in Σ.

Elke $\mathcal{L}$-substitutie $\sigma: \Sigma^* \to P(\Delta^*)$ wordt uitgebreid tot een afbeelding $\sigma: P(\Sigma^*) \to P(\Delta^*)$ door middel van $\sigma(L) = \cup\{\sigma(x) \mid x \in L\}$ voor alle talen L over Σ.

7.7. Voorbeeld

Als $\sigma: \Sigma^* \to P(\Delta^*)$ met $\Sigma = \{a,b\}$ en $\Delta = \{0,1\}$ een $\mathcal{L}_3$-substitutie is, gedefinieerd door $\sigma(a) = \{10^n1 \mid n \geq 1\}$ en $\sigma(b) = \{01^n0 \mid n \geq 1\}$, dan is

$\sigma(abb) = \{10^n1 \mid n \geq 1\}\{01^n0 \mid n \geq 1\}\{01^n0 \mid n \geq 1\} =$
$= \{10^i101^j001^k0 \mid i,j,k \geq 1\}.$

Zij $L = \{(ab)^n \mid n \geq 1\}$; dan is
$10^2101^3010^2101^30 \in \sigma(L)$, en
$10^2101^5010^7101^20 \in \sigma(L)$. ●

In de studie van operaties op talen stelt men in het bijzonder belang in paren bestaande uit een klasse $\mathcal{L}$ van talen en een operatie f, zodanig dat het toepassen van f op talen uit $\mathcal{L}$ weer een taal in $\mathcal{L}$ oplevert.

7.8. Definitie: Zij $\mathcal{L}$ een klasse van talen en f een operatie van rang n. $\mathcal{L}$ is gesloten onder f indien voor alle n-tallen talen $(L_1,\ldots,L_n)$ met $L_i \in \mathcal{L}$ $(1 \leqslant i \leqslant n)$ geldt, dat $f(L_1,\ldots,L_n) \in \mathcal{L}$. Een klasse van talen $\mathcal{L}$ is gesloten onder substitutie, indien $\mathcal{L}$ gesloten is onder $\mathcal{L}$-substitutie.

7.9. Oefeningen

(1) Zij h: $\{0,1,2\}^* \to \{a,b\}^*$ een homomorfisme gedefinieerd door $h(0) = abb$; $h(1) = aa$; $h(2) = bb$. Bepaal $h(1102)$, $h(\{(10)^n 2 \mid n \geqslant 0\})$ en $h^{-1}(\{a^m b^n \mid m,n \geqslant 0\})$.

(2) Zij σ: $\{a,b\}^* \to P(\{0,1\}^*)$ een substitutie gedefinieerd door $\sigma(a) = \{0^n \mid n \geqslant 1\}$ en $\sigma(b) = \{0^n \mid n \geqslant 2\} \cup \{1^n \mid n \geqslant 2\}$. Bepaal $\sigma(\{(ab)^n a \mid n \geqslant 1\})$.

(3) Toon aan dat een homomorfisme h: $P(\Sigma^*) \to P(\Sigma^*)$ een (halfring-) homomorfisme van $(P(\Sigma^*),(\cup,\cdot,\emptyset,\{\lambda\}))$ in $(P(\Delta^*),(\cup,\cdot,\emptyset,\{\lambda\}))$ is ($\cdot$ is de concatenatie-operatie).

(4) Bewijs de volgende bewering: Een functie f: $P(\Sigma^*) \to P(\Delta^*)$ is een substitutie dan en slechts dan als f een (halfring-)homomorfisme van $(P(\Sigma^*),(\cup,\cdot,\emptyset,\{\lambda\}))$ in $(P(\Delta^*),(\cup,\cdot,\emptyset,\{\lambda\}))$ is ($\cdot$ is de concatenatie-operatie).

(5) Toon aan dat $\mathcal{L}_3$ gesloten is onder substitutie (Aanwijzing: gebruik reguliere expressies).

		$\mathcal{L}_3$	$\mathcal{L}_2$	$\mathcal{L}_1$	$\mathcal{L}_0$	$\mathcal{L}_{lin}$	$\mathcal{L}_{DPDA}$	$\mathcal{L}_{DLBA}$
spiegeling		+	+	+	+	+	-	+
vereniging		+	+	+	+	+	-	+
doorsnede		+	-	+	+	-	-	+
complement		+	-	?	—	-	+	+
concatenatie		+	+	+	+	-	-	+
iteratie		+	+	+	+	-	-	+
λ-vrije iteratie		+	+	+	+	-	-	+
homomorfisme	*)	+	+	-	+	+	-	-
λ-vrij homomorfisme	*)	+	+	+	+	+	-	+
invers homomorfisme	*)	+	+	+	+	+	+	+
substitutie	*)	+	+	-	+	-	-	-
λ-vrije substitutie	*)	+	+	+	+	-	-	+
doorsnede met een reguliere taal	*)	+	+	+	+	+	+	+

Tabel 7.1.

\+ = gesloten; - = niet gesloten; ? = onbekend (open probleem).

*) Merk op dat deze regel in de tabel niet betrekking heeft op één enkele operatie, maar op oneindig veel operaties (alle homomorfismen, alle substituties, enz.).

In tabel 7.1. wordt een overzicht gegeven van de afsluitingseigenschappen van de meest bekende taalklassen. In deze tabel wordt met een plusteken aangegeven, dat de betreffende klasse van talen gesloten is met betrekking tot de gespecificeerde operatie; een minteken geeft aan, dat de betreffende klasse van talen niet gesloten is onder deze operatie, terwijl een vraagteken een "open probleem" aanduidt.

Enkele van de in eigenschap 7.10. genoemde resultaten worden hieronder bewezen; het resterende deel van de bewijzen wordt ter oefening aan de lezer overgelaten, of valt buiten het bestek van dit college (b.v. het feit dat $\mathcal{L}_{DPDA}$ gesloten is onder de complement-operatie).

7.10. Eigenschap

De klasse $\mathcal{L}_2$ is gesloten onder (a) spiegelen, (b) vereniging, (c) concatenatie, (d) λ-vrije iteratie, (e) iteratie, (f) homomorfisme, en (g) substitutie.

Bewijs

Zij $G_1 = (V_1,\Sigma_1,P_1,S_1)$ en $G_2 = (V_2,\Sigma_2,P_2,S_2)$ contextvrije grammatica's met $(V_1 \setminus \Sigma_1) \cap (V_2 \setminus \Sigma_2) = \emptyset$ en zij S een nieuw hulpsymbool. Voor elke operatie f van rang n construeren we een contextvrije grammatica G_f, zodanig dat $L(G_f) = f(L(G_1),\ldots,L(G_n))$.

(a) Definieer $G_s = (V_1,\Sigma_1,P_s,S_1)$ met $P_s = \{A \to \phi^s \mid A \to \phi \in P_1\}$. Dan is $L(G_s) = (L(G_1))^s$.

(b) Definieer $G_\cup = (V_\cup, \Sigma_1 \cup \Sigma_2, P_\cup, S)$ met $V_\cup = V_1 \cup V_2 \cup \{S\}$ en $P_\cup = \{S \to S_1, S \to S_2\} \cup P_1 \cup P_2$. Dan geldt $L(G_\cup) = L(G_1) \cup L(G_2)$.

(c) Definieer $G_\bullet = (V_\bullet, \Sigma_1 \cup \Sigma_2, P_\bullet, S)$ met $V_\bullet = V_1 \cup V_2 \cup \{S\}$ en $P_\bullet = \{S \to S_1S_2\} \cup P_1 \cup P_2$; zodat $L(G_\bullet) = L(G_1)L(G_2)$.

(d) Definieer $G_+ = (V_+,\Sigma_1,P_+,S)$ met $V_+ = V_1 \cup \{S\}$ en $P_+ = \{S \to S_1, S \to S_1S\} \cup P_1$. Dan is $L(G_+) = (L(G_1))^+$.

(e) als (d) met $S \to \lambda$ in plaats van $S \to S_1$.

(f) Zij $h: \Sigma_1^* \to \Delta^*$ een homomorfisme. Definieer een contextvrije grammatica $G_h = (V_h,\Delta,P_h,S_1)$ met $V_h = (V_1 \setminus \Sigma_1) \cup \Delta$ en $P_h = \{A \to g(\phi) \mid A \to \phi \in P_1\}$, waarin $g: V_1^* \to V_h^*$ het homomorfisme is, bepaald door

$g(\alpha) = h(\alpha)$ voor $\alpha \in \Sigma_1$, en
$g(\alpha) = \alpha$ voor $\alpha \in V_1 \setminus \Sigma_1$.

Dan geldt $L(G_h) = h(L(G_1))$.

(g) Zij σ: $\Sigma_1^* \to P(\Delta^*)$ een $\mathcal{L}_2$-substitutie; dit houdt in, dat voor alle α in Σ_1 er een contextvrije grammatica $G_\alpha=(V_\alpha,\Delta,P_\alpha,S_\alpha)$ bestaat zodanig dat $L(G_\alpha) = \sigma(\alpha)$. Zonder verlies van algemeenheid mogen we aannemen dat de verzamelingen hulpsymbolen van alle grammatica's G_1 en G_α ($\alpha \in \Sigma_1$) onderling disjunct zijn. Definieer nu $G_\sigma = (V_\sigma,\Delta,P_\sigma,S_1)$ met

$V_\sigma = \cup\{V_\alpha \mid \alpha \in \Sigma_1\} \cup (V_1\backslash\Sigma_1)$ en
$P_\sigma = \cup\{P_\alpha \mid \alpha \in \Sigma_1\} \cup \{A \to h(\psi) \mid A \to \psi \in P_1\}$

waarin $h: V_1^* \to V_\sigma^*$ het homomorfisme is, gedefinieerd door

$h(\alpha) = S_\alpha$ voor alle α in Σ_1
$h(\alpha) = \alpha$ voor alle α in $V_1\backslash\Sigma_1$.

Dan is $L(G_\sigma) = \sigma(L(G_1))$. (Ga dit na)

Einde bewijs

Om de overeenkomstige eigenschap voor de klassen $\mathcal{L}_0$ en $\mathcal{L}_1$ (In dit laatste geval moeten de homomorfismen en substituties λ-vrij zijn) te bewijzen, zijn gecompliceerdere constructies nodig. De grammatica's zoals in bovenstaand bewijs gedefinieerd, voldoen dan niet langer.

7.11. Eigenschap

De klasse $\mathcal{L}_2$ is niet gesloten onder doorsnede en complement.

Bewijs

Beschouw $L_1 = \{a^k b^k a^n \mid n,k \geq 1\}$ en $L_2 = \{a^k b^n a^n \mid n,k \geq 1\}$; deze talen zijn contextvrij (Ga dit na). Maar $L_1 \cap L_2 = \{a^m b^m a^m \mid m \geq 1\}$ is niet contextvrij (zie bewijs van eigenschap 3.34.). Dus is $\mathcal{L}_2$ niet gesloten onder doorsnede.

Veronderstel, dat $\mathcal{L}_2$ gesloten is onder complement, dan is - vanwege het feit dat $\mathcal{L}_2$ gesloten is onder vereniging (zie eigenschap 7.10.(b)) - de taal $L_1 \cap L_2 = \overline{(\overline{L_1} \cup \overline{L_2})}$ ook contextvrij; hetgeen niet waar is. Dus is de onderstelling, dat $\mathcal{L}_2$ gesloten is onder complement, onjuist.

Einde bewijs

7.12. Oefening

Bewijs de volgende beweringen door de bewijzen van de eigenschappen 7.10. en 7.11. op de juiste wijze aan te passen.

(1) $\mathcal{L}_3$ is gesloten onder spiegeling, vereniging, concatenatie, (λ-vrije) iteratie en homomorfisme.
(2) $\mathcal{L}_{lin}$ is gesloten onder spiegeling, vereniging en homomorfisme.
(3) $\mathcal{L}_{lin}$ is niet gesloten onder doorsnede en complement.

Bewijs bovendien dat
(4) $\mathcal{L}_{lin}$ niet gesloten is onder concatenatie (zie eigenschap 3.16.).

7.13. Eigenschap

De klasse $\mathcal{L}_3$ is gesloten onder complement en doorsnede.

Bewijs
Zij L een type 3 taal. Dan bestaat er een deterministische eindige automaat $A = (Q,\Sigma,\sigma,q_0,F)$, die L accepteert. Definieer nu $\overline{A} = (Q,\Sigma,\sigma,q_0,\overline{F})$ met $\overline{F} = Q\backslash F$. Dan geldt $L(\overline{A}) = \Sigma^*\backslash L(A)$ (Toon dit aan). Dus is $\mathcal{L}_3$ gesloten onder complement.

Zij L_1 en L_2 type 3 talen. Dan geldt $L_1 \cap L_2 = \overline{(\overline{L_1} \cup \overline{L_2})}$ waarin $\overline{L}$ als gebruikelijk een afkorting is voor $\Sigma^*\backslash L$. Daar $\mathcal{L}_3$ gesloten is onder vereniging en complement, geldt $L_1 \cap L_2 \in \mathcal{L}_3$; dus is $\mathcal{L}_3$ gesloten onder doorsnede.

Van deze laatste bewering geven we nu nog een alternatief, constructief bewijs: zij L_1 en L_2 type 3 talen. Dan bestaan er deterministische eindige automaten $A_1 = (Q_1,\Sigma_1,\sigma_1,q_{01},F_1)$ en $A_2 = (Q_2,\Sigma_2,\sigma_2,q_{02},F_2)$, die respectievelijk L_1 en L_2 accepteren. Definieer $A_0 = (Q_1 \times Q_2, \Sigma_1 \cap \Sigma_2, \sigma_0, (q_{01},q_{02}), F_1 \times F_2)$ met $\sigma_0: (Q_1 \times Q_2) \times (\Sigma_1 \cap \Sigma_2) \to (Q_1 \times Q_2)$

$\sigma_0((q_1,q_2),a) = (\sigma_1(q_1,a), \sigma_2(q_2,a))$ voor alle $q_1 \in Q_1$, $q_2 \in Q_2$ en $a \in \Sigma_1 \cap \Sigma_2$.

Er geldt dan:

$((q_1,q_2),xy) \overset{*}{\underset{A_0}{\vdash}} ((q_1',q_2'),y)$ dan en slechts dan als

$$(q_1,xy) \overset{*}{\underset{A_1}{\vdash}} (q_1',y) \text{ en } (q_2,xy) \overset{*}{\underset{A_2}{\vdash}} (q_2',y)$$

(Toon dit aan). Met behulp van deze hulpeigenschappen bewijst men vervolgens: $L(A_0) = L(A_1) \cap L(A_2) = L_1 \cap L_2$, waaruit volgt dat $\mathcal{L}_3$ gesloten is onder doorsnede.

Einde bewijs

7.14. Eigenschap

De klasse $\mathcal{L}_2$ is gesloten onder (a) doorsnede met een reguliere taal, en (b) invers homomorfisme.

Bewijs

(a) Zij L een contextvrije taal en R een reguliere taal. Dan bestaan er een stapelautomaat $A_L = (Q_L,\Sigma_L,\Gamma,\sigma_L,q_{0L},\gamma_0,F_L)$ en een deterministische eindige automaat $A_R = (Q_R,\Sigma_R,\sigma_R,q_{0R},F_R)$ die L en respectievelijk R accepteren. Definieer een stapelautomaat $A = (Q_L \times Q_R,\ \Sigma_L \cap \Sigma_R,\ \Gamma,\ \sigma,\ (q_{0L},q_{0R}),\ \gamma_0,\ F_L \times F_R)$ waarin σ de afbeelding van $(Q_L \times Q_R) \times ((\Sigma_L \cap \Sigma_R) \cup \{\lambda\}) \times \Gamma$ in de eindige deelverzamelingen van $(Q_L \times Q_R) \times \Gamma^*$ is zodanig dat voor alle $q_L,q_L' \in Q_L$; $q_R,q_R' \in Q_R$; $a \in (\Sigma_L \cap \Sigma_R)$; $\gamma \in \Gamma$; en $\xi \in \Gamma^*$ geldt:

$((q_L',q_R'),\xi) \in \sigma((q_L,q_R),a,\gamma)$ dan en slechts dan als $(q_L',\xi) \in \sigma_L(q_L,a,\gamma)$ en $\sigma_R(q_R,a) = q_R'$,

en $((q_L',q_R'),\xi) \in \sigma((q_L,q_R),\lambda,\gamma)$ dan en slechts dan als $(q_L',\xi) \in \sigma_L(q_L,\lambda,\gamma)$ en $q_R' = q_R$.

Met inductie kan men bewijzen dat

$$((q_{0L},q_{0R}),w,\gamma_0) \overset{n}{\underset{A}{\vdash}} ((q_L,q_R),\lambda,\xi)$$

dan en slechts dan geldt als

$$(q_{0L},w,\gamma_0) \overset{n}{\underset{A_L}{\vdash}} (q_L,\lambda,\xi) \text{ en } (q_{0R},w) \overset{|w|}{\underset{A_R}{\vdash}} (q_R,\lambda).$$

Uitgaande van deze hulpeigenschap kan men aantonen, dat $L(A) = L(A_L) \cap L(A_R) = L \cap R$; dus is $\mathcal{L}_2$ gesloten onder doorsnede met een reguliere taal.

(b) Zij L een contextvrije taal, en $A = (Q,\Sigma,\Gamma,\sigma,q_0,\gamma_0,F)$ een stapelautomaat, die L accepteert. Zij $h: \Delta^* \to \Sigma^*$ een homomorfisme. We construeren een stapelautomaat $A' = (Q',\Delta,\Gamma,\sigma',q_0',\gamma_0,F')$ zodanig dat $L(A') = h^{-1}(L(A)) = h^{-1}(L)$. Zij $k = \max\{|h(\alpha)| \mid \alpha \in \Delta\}$ en $\Sigma^{\leq k} = \{\lambda\} \cup \Sigma \cup \Sigma^2 \cup \ldots \cup \Sigma^k$. Definieer nu $Q' = Q \times \Sigma^{\leq k}$, $q_0' = (q_0,\lambda)$, $F' = F \times \{\lambda\}$, terwijl σ' als volgt bepaald is: voor alle $q \in Q$; $ax \in \Sigma^{\leq k}$; $a \in \Sigma \cup \{\lambda\}$; $b \in \Delta$; en $\gamma \in \Gamma$:

$$\sigma'((q,\lambda),b,\gamma) = \{((q,h(b)),\gamma)\}$$
$$\sigma'((q,ax),\lambda,\gamma) = \{((q',x),\omega) \mid (q',\omega) \in \sigma(q,a,\gamma)\}.$$

Het basis idee van de constructie is vrij eenvoudig. De tweede component van de toestand dient als buffer en bevat steeds een woord ter lengte $\leq k$. Aan de hand van een invoersymbool $b \in \Delta$ wordt de buffer gevuld met h(b). A' simuleert dan de berekening van A (A gebruikt de inhoud van de buffer als invoer) tot de buffer leeg is. Na eventueel enkele λ-stappen wordt er weer een symbool van de invoer van A' gelezen, de buffer gevuld en de simulatie van A voortgezet. Accepteren van de invoer vindt plaats als een eindtoestand van A bereikt is, en de buffer leeg is.

Het bewijs, dat $L(A') = h^{-1}(L(A))$, wordt aan de lezer overgelaten.
Einde bewijs

7.15. Oefening

Bewijs dat $\mathcal{L}_3$ gesloten is onder invers homomorfisme (Aanwijzing: gebruik transitiesystemen).

Naast het definiëren van nieuwe talen door middel van operaties, kunnen (verzamelingen van) operaties ook gebruikt worden om klassen van talen te kenmerken. We besluiten dit hoofdstuk met een voorbeeld.

7.16. Eigenschap (Stelling van Kleene)

$\mathcal{L}_3$ is de kleinste klasse van talen, die alle eindige talen bevat, en gesloten is onder vereniging, concatenatie en iteratie.

Bewijs

Zij $\mathcal{K}$ de kleinste klasse van talen, die alle eindige talen bevat en gesloten is onder $\cup$, $\cdot$ en *. Daar $\mathcal{L}_3$ alle eindige talen bevat en gesloten is onder $\cup$, $\cdot$ en * (zie tabel 7.1.; oefening 7.12. of het bewijs van eigenschap 5.14.) geldt $\mathcal{K} \subseteq \mathcal{L}_3$.

Omgekeerd volgt uit het bewijs van eigenschap 5.16. dat elke type 3 taal te verkrijgen is door de operaties $\cup$, $\cdot$ en * een eindig aantal keren toe te passen op eindige talen (Toon dit aan). Daaruit volgt $\mathcal{L}_3 \subseteq \mathcal{K}$.
Samengevat levert dit $\mathcal{L}_3 = \mathcal{K}$.
Einde bewijs

8. UITLEIDING

In dit hoofdstuk zal een aantal richtingen worden aangegeven, waarin wordt voortgebouwd op de in de vorige hoofdstukken behandelde stof.

8.1. Programmeertaal-ontleders en -vertalers

In hoofdstuk 6 is het volgende probleem al ter sprake gekomen: bepaal van een gegeven tekst of deze al dan niet een syntactisch correct Pascal-programma is. Ontleders en vertalers voor programmeertalen dienen echter meer informatie af te leveren dan alleen de uitkomst van een dergelijke beslissing. Indien de gegeven tekst geen (syntactisch correct) Pascal is, moet de vertaler voor de gebruiker zinvolle foutmeldingen voortbrengen; anderzijds dient van een aangeboden (syntactisch correcte) Pascal-tekst een ontleding gegeven te worden. De syntactische structuur is namelijk bepalend voor de betekenis van de tekst en dus voor het vertalen ervan in bijvoorbeeld PDP-11-assembler.

Voor het ontleden van een Pascal-tekst kan te werk worden gegaan op de wijze, die de stapelautomaat van 6.20. toepast. Daar worden op systematische wijze alle afleidingen geprobeerd, die consistent zijn met de invoersymboolrij. De methode werkt van boven naar beneden ("top-down") in de zin dat met het start-symbool van de grammatica wordt begonnen en vanaf daar naar een zin van L(G) wordt toegewerkt. Het is ook mogelijk andersom te werk te gaan. Dan zoekt men voor een produktieregel $A \rightarrow \alpha$ naar een voorkomen van het deelwoord α. Vervolgens wordt α tot A "gereduceerd": α wordt vervangen door A. Zo werkt men vanaf de gegeven symboolrij naar het startsymbool toe. Beide methoden zijn in het algemeen niet erg efficient. Dit wordt veroorzaakt door al het terugsporen dat nodig is. Er wordt daarom gezocht naar ontledingsalgoritmen, die voor bepaalde soorten contextvrije grammatica's zeer efficiënt werken. Verder wil men bij een gegeven soort ontledingsalgoritme de klasse grammatica's kenmerken, die efficiënte ontledingsalgoritmen van de betreffende soort bezitten.

Algoritmen, die als uitgangspunt voor een vertaler kunnen dienen, verkrijgt men door de in dit dictaat gebruikte automaten te voorzien van

een uitvoer-mogelijkheid. In termen van de fysische interpretatie van automaten gebeurt dit door een uitvoerband met schrijfkop toe te voegen. De schrijfkop kan alleen naar rechts bewegen en de uitvoerband kan worden beschreven maar niet gelezen. Uitgaande van een eindige automaat, bijvoorbeeld, maakt men een "eindige vertaler" door een uitvoerband toe te voegen en te definiëren dat de transitiefunctie bij een paar (toestand, invoersymbool) nu een paar (toestand, rij van uitvoersymbolen) geeft, of een eindige verzameling van zulke paren in het niet-deterministische geval. Op een overeenkomstige manier kunnen (niet-)deterministische stapelvertalers worden gedefinieerd. Kernpunt van onderzoek is de vraag: gegeven een klasse $\mathcal{L}$ van talen en een klasse $\mathcal{T}$ van vertalers, welke klasse van talen vormt $\mathcal{T}(\mathcal{L})$, de verzameling van alle vertalingen van de talen uit $\mathcal{L}$.

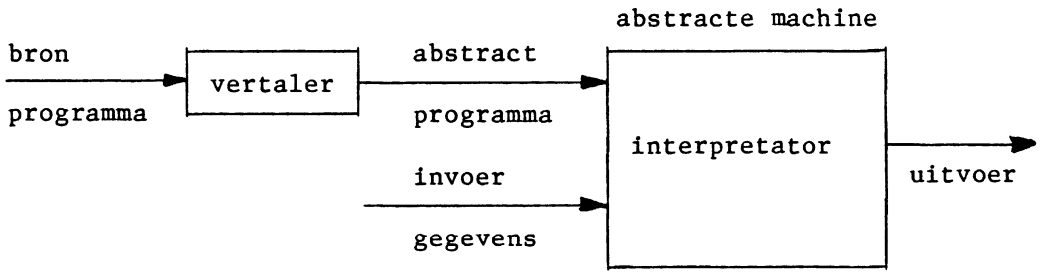
8.2. Definiëren van syntaxis en semantiek

In Pascal wordt de verzameling legale programma's vastgelegd met behulp van een contextvrije grammatica aangevuld met beperkingen in het Engels (zoals: men mag alleen identifiers gebruiken, die reeds gedeclareerd zijn). De betekenis van de diverse taalconstructies wordt ook in het Engels gegeven. Het is niet nodig hiervoor een natuurlijke taal te gebruiken, daar al het noodzakelijke kan worden gespecificeerd door middel van een type 0 grammatica. Dit resulteert echter niet steeds in een heldere en voor de hand liggende beschrijving. Er worden daarom alternatieven onderzocht.

Dit kunnen modificaties zijn van herschrijfsystemen. Voor de definitie van ALGOL-68, bijvoorbeeld, is een nieuw soort herschrijfsysteem gebruikt. Dit is een twee-traps systeem: de herschrijfregels, die mogen worden gebruikt, worden zelf weer voortgebracht door een herschrijfsysteem.

Er zijn ook machine-gerichte alternatieven. Een van de oudste voorstellen voor een volledige definitie van een programmeertaal is de suggestie daarvoor een werkelijke implementatie te nemen. Twee bezwaren van deze methode zijn, dat het onvermijdelijk is dat de hardware van de betreffende machine de taal binnensluipt en dat de definitie slechts beperkt

toegankelijk is. Om aan deze bezwaren tegemoet te komen, ontwikkelde het IBM laboratorium in Wenen het idee om voor de implementatie een hypothetische machine te gebruiken. Dit leidde tot VDL, de Vienna Definition Language. In VDL wordt de betekenis van een programma gedefinieerd als de rij toestandsveranderingen van een abstracte machine, die het programma uitvoert. Hoe het programma wordt uitgevoerd, is gedefinieerd door een algoritme; de interpretator.



Om onderscheid te maken tussen statische en dynamische eigenschappen wordt het bronprogramma eerst vertaald in een abstract programma door een tweede algoritme, de vertaler.

Om nu een programmeertaal te definiëren moeten we aangeven wat voor deze definitie de toestandsverzameling van de abstracte machine moet zijn en hoe de vertaler en de interpretator werken. Dit alles dient in de precies gedefinieerde notatie van VDL te geschieden. Deze methode is gebruikt voor een formele definitie van de programmeertaal PL/1.

8.3. Complexiteitstheorie

In dit dictaat zijn twee typen automaten besproken: de eindige automaat en de stapelautomaat. Zoals in hoofdstuk 6 aangegeven werd, zijn deze op te vatten als verbijzonderingen van een meer algemeen type automaat: de Turing-machine. De nadruk in dit dictaat ligt vooral op problemen als "welke talen kunnen worden geaccepteerd door stapelautomaten?" zonder rekening te houden met de hoeveelheid geheugencellen, die wordt gebruikt en zonder te letten op het aantal stappen in een berekening (de "rekentijd" van de stapelautomaat). In de complexiteitstheorie ligt het zwaartepunt juist op deze kostenaspecten. Gevraagd wordt bijvoorbeeld, welke

talen L over Σ geaccepteerd kunnen worden door een (niet-)deterministische Turing-machine, zodanig dat voor de berekening, die bepaalt of $w \in L$ ten hoogste $f(|w|)$ geheugencellen nodig zijn, waarbij $f: N \to N$ een of andere gegeven functie is (bijvoorbeeld: $f(n) = \lceil \log_2 n \rceil$ of $f(n) = n$). Een belangrijke vraag is ook wat de kracht van niet-determinisme is. Een taal $L \subseteq \Sigma^*$ behoort tot de klasse NP (respectievelijk, P) als er een niet-deterministische (deterministische) Turing-machine en een polynoom $p: N \to N$ bestaan, zodanig dat voor elke $w \in \Sigma^*$ er een berekening van ten hoogste $p(|w|)$ stappen bestaat, die bepaalt of $w \in L$. In concreto betekent de vraag naar de kracht van niet-determinisme, bijvoorbeeld, of P = NP geldt. Deze vraag is één van de prominente onopgeloste problemen van de informatica. In het college al74 Theoretische Informatica II wordt nader op de complexiteitstheorie ingegaan.

8.4. Alternatieve definities

De relatie $\overset{*}{\Longrightarrow}$ van een herschrijfsysteem of grammatica werd in hoofdstuk 1 zodanig gedefinieerd dat het mogen toepassen van een herschrijfregel op een zinsvorm onafhankelijk is van de vraag hoe men aan deze zinsvorm is gekomen: alles wat kan, dat mag ook. Het herschrijven kan op verschillende manieren worden bestuurd. Neem bijvoorbeeld de grammatica $G = (\{S,A,B,a,b\}, \{a,b\}, P, S)$ met productieregels

0: S → ABS
1: A → aA
2: B → bB
3: S → aS
4: A → a
5: B → b
6: S → a

De productieregels zijn van een etiket voorzien om bij een afleiding een woord te bepalen, dat deze afleiding beschrijft. Bij de afleiding $S \overset{0}{\Longrightarrow} ABS \overset{0}{\Longrightarrow} ABABS \overset{4}{\Longrightarrow} ABaBS$ hoort het woordt 004. Het herschrijfproces kan nu worden bestuurd door alleen die herschrijvingen toe te laten, waarvan het bijbehorende woord behoort tot een gegeven besturingstaal C over het alfabet $\{0,1,2,\ldots,6\}$. Neem bijvoorbeeld $C = \{0(123)^n456 \mid n > 0\}$, dan is $L(G,C) = \{a^n b^n a^n \mid n > 1\}$. Deze taal is niet contextvrij terwijl L(G) contextvrij is en C regulier is.

De relatie $\Rightarrow$ van een herschrijfsysteem werd in hoofdstuk 1 zodanig gedefinieerd, dat op één plaats een deelwoord wordt herschreven. Het herschrijven vindt dus sequentieel plaats. Men kan, althans uitgaande van contextvrije productieregels, ook parallel herschrijven (zie hoofdstuk 1). Neem bijvoorbeeld een herschrijfsysteem met één regel: $a \rightarrow a^2$. Dan geldt $a \Rightarrow aa \Rightarrow aaaa \Rightarrow a^8 \Rightarrow a^{16} \Rightarrow \ldots$ indien men steeds alle a's parallel herschrijft. Dit soort parallelle herschrijfsystemen vindt men in de literatuur onder de naam "L systems" (van "Lindenmayer systems", ook wel "developmental systems"). Ze zijn ontstaan in de theoretische biologie als modellen voor het groeien van bepaalde, eenvoudige organismen.

8.5. Herschrijven van gecompliceerde objecten

Ook objecten van hogere dimensie kan men herschrijven. Bijvoorbeeld, in het 2-dimensionale geval leveren de regels

$X \rightarrow H$

$$X \rightarrow \begin{array}{c} X \\ | \\ X - C - OH \\ | \\ X \end{array} \qquad\qquad X \rightarrow \begin{array}{c} X \\ | \\ X - C - H \\ | \\ X \end{array}$$

(X is het enige hulpsymbool; C, H, O, — en | zijn eindsymbolen) uitgaande van

$$\begin{array}{c} X \\ | \\ X - C - OH \\ | \\ X \end{array}$$

de structuurformules van alle alkanolen en polyhydroxyalkanolen. Voor dergelijke herschrijfsystemen kan het herschrijven ook sequentieel of parallel plaatsvinden, en al dan niet onder een of andere vorm van besturing. De theorie van dergelijke meer-dimensionale herschrijfsystemen

is aanzienlijk ingewikkelder en tot op heden minder diepgaand uitgewerkt dan het één-dimensionale geval.

8.6. Algebraïsche benadering

Hoofdstuk 7 eindigde met een voorbeeld van het karakteriseren van een klasse talen door middel van een verzameling talen en een verzameling operaties. Een, in de literatuur, in deze context veel voorkomende (oneindige) verzameling operaties bestaat uit vereniging, (λ-vrij) homomorfisme, invers homomorfisme, doorsnede met een reguliere taal, eventueel aangevuld met concatenatie en iteratie. Een klasse van talen, die gesloten is onder deze operaties wordt een AFL ("Abstract Family of Languages") genoemd. Voor deze AFL's en soortgelijke structuren zijn dergelijke karakteriseringsstellingen op een abstract niveau bewezen.

9. LITERATUUR

De onderstaande literatuurlijst beperkt zich tot een aantal studieboeken. Voor verwijzingen naar oorspronkelijke literatuur raadplege men de in deze lijst genoemde standaardwerken.

1. A.V. Aho & J.D. Ullman: "The Theory of Parsing, Translation and Compiling" Volume I (1972) & II (1973) Prentice-Hall, Englewood Cliffs, N.J.

2. F.S. Beckman: "Mathematical Foundations of Programming" (1980) Addison-Wesley, Reading, Mass.

3. J. Berstel: "Transductions and Context-Free Languages" (1979) Teubner Studienbücher, B.G. Teubner, Stuttgart.

4. J.C. Cleaveland & R.C. Uzgalis: "Grammars for Programming Languages" (1977) North-Holland, Amsterdam [twee-traps grammatica's].

5. P.J. Denning, J.B. Dennis & J.E. Qualitz: "Machines, Languages, and Computation" (1978) Prentice-Hall, Englewood Cliffs, N.J.

6. S. Eilenberg: "Automata, Languages and Machines" Volume A (1974) Academic Press, N.Y.

7. S. Ginsburg: "The Mathematical Theory of Context-Free Languages" (1966) McGraw-Hill, N.Y.

8. S. Ginsburg: "Algebraic and Automata-Theoretic Properties of Formal Languages" (1975) North-Holland, Amsterdam [AFL-theorie].

9. M.A. Harrison: "Introduction to Formal Language Theory" (1978) Addison-Wesley, Reading, Mass.

10. F.C. Hennie: "Introduction to Computability" (1977) Addison-Wesley, Reading, Mass. [Hoofdstuk 2].

11. G.T. Herman & G. Rozenberg: "Developmental Systems and Languages" (1975) North-Holland, Amsterdam [parallel herschrijven].

12. J.E. Hopcroft & J.D. Ullman: "Formal Languages and Their Relation to Automata" (1969) Addison-Wesley, Reading, Mass.

13. J.E. Hopcroft & J.D. Ullman: "Introduction to Automata Theory, Languages, and Computation" (1979) Addison-Wesley, Reading, Mass.

14. H.R. Lewis & C.H. Papadimitriou: "Elements of the Theory of Computation" (1981) Prentice-Hall, Englewood Cliffs, N.J.

15. R. McNaughton: "Elementary Computability, Formal Languages, and Automata" (1982) Prentice-Hall, Englewood Cliffs, N.J.

16. G. Révész: "Introduction to Formal Languages" (1983) McGraw-Hill, N.Y.

17. H. Rogers Jr.: "Theory of Recursive functions and Effective Computability" (1967) McGraw-Hill, N.Y. [Hoofdstuk 2].

18. G. Rozenberg & A. Salomaa: "The Mathematical Theory of L Systems" (1980) Academic Press, N.Y. [parallel herschrijven].

19. A. Salomaa: "Theory of Automata" (1969) Pergamon, Oxford.

20. A. Salomaa: "Formal Languages" (1973) Academic Press, N.Y.

21. A. Salomaa: "Jewels of Formal Language Theory" (1981) Pitman, London.

22. W.J. Savitch: "Abstract Machines and Grammars" (1982) Little Brown, Boston, Mass.

23. D. Wood: "Grammar and L Forms: An Introduction" (1980) Lecture Notes in Computer Science 91, Springer, New York/Heidelberg/Berlin.

De in deze lijst genoemde boeken over de theorie van formele talen bevatten vrijwel zonder uitzondering één of meer paragrafen over berekenbaarheid, beslisbaarheid en dergelijke. De boeken over berekenbaarheid gaan natuurlijk veel verder dan in het kader van dit college nodig is. De inleidende hoofdstukken geven veelal een goed informeel overzicht (in het bijzonder [10, 17]).

Wat betreft de formele talen theorie, zijn [1, 5, 9, 12, 13, 16, 20, 22] uitstekende standaardwerken. De in deze boeken behandelde stof omvat onder andere ook de onderwerpen, die in dit dictaat aan de orde komen. Andere boeken zijn weer meer geschikt voor voortgezette studie [3, 6, 8, 21, 23] en/of behandelen bijzondere onderwerpen genoemd in hoofdstuk 8 [4, 8, 11, 18, 23].

10. VRAAGSTUKKEN

1.1. Construeer contextvrije grammatica's voor elk van de volgende talen:

$L_1 = \{a^{3i}b^i \mid i \geq 1\}$,
$L_2 = \{a^ib^j \mid i \geq j \geq 0\}$,
$L_3 = \{x \mid x \text{ in } \{a,b\}^*, \text{ en } \#(a,x) = \#(b,x)\}$, en
$L_4 = (L_1L_2 \cup L_2)^*$

Bewijs dat uw constructies correct zijn.

1.2. Verander de definitie van grammatica zodanig, dat in plaats van een startsymbool er nu een eindige verzameling I van startwoorden (over V) is. L(G) wordt dan: $L(G) = \{x \mid x \text{ in } \Sigma^*, \text{ en er is een i in I met } i \overset{*}{\Longrightarrow} x\}$. De vier typen restricties op de vorm van de regels blijven dezelfde. Bewijs dat de families $\mathcal{L}_i$ (i = 0, 1, 2, 3) ook dezelfde blijven onder deze verandering.

1.3. Verander de definitie van type 1 grammatica zodanig, dat naast $S \to \lambda$ nu ook regels $A \to \lambda$ voor elke A uit $V \setminus \Sigma$ zijn toegelaten. Bewijs dat elke type 0 taal nu door zo'n veranderde type 1 grammatica kan worden voortgebracht.

1.4. Zij $G = (V,\Sigma,P,S)$ een type 0 grammatica met $\Sigma = \{a\}$, $V \setminus \Sigma = \{S,A,B,C,M,N,T,X\}$ en $P = \{S \to MTC,\ C \to TC,\ C \to M,\ TM \to XA,\ TX \to XTB,\ MX \to MN,\ NT \to TN,\ NA \to Ma,\ BT \to TB,\ BA \to Aa,\ MM \to \lambda\}$. Bewijs, dat $L(G) = \{a^{t(n)} \mid n \geq 1;\ t(n) = 1 + 2 + \ldots + n\}$.

1.5. Construeer een type 0 grammatica G zodanig dat $L(G) = \{a^nb^{n^2} \mid n \geq 1\}$ Toon aan dat uw constructie correct is.

1.6. Construeer een contextgevoelige grammatica die de taal $\{ww \mid w \text{ in } \{0,1\}^*\}$ voortbrengt. Bewijs dat uw constructie correct is.

1.7. Zij $G = (V, \Sigma, P, S)$ met $V\Sigma = \{S,A,B\}$, $\Sigma = \{0,1\}$ en $P = \{S \to 0AB, A1 \to SB1, A0 \to S0B, B \to SA, B \to 01, 1B \to 0\}$. Bewijs dat $L(G) = \emptyset$.

1.8. Construeer een herschrijfsysteem dat een woord $¢a^rb^s\$$ (met $r \geq 0$ en $s \geq 1$) omzet in een woord $a^rb^sc^td^u$ zodanig dat $r = t.s + u$ en $0 \leq u \leq s-1$. Bewijs dat uw constructie correct is.

1.9. Construeer een herschrijfsysteem dat een gegeven woord $¢aPbq\$$ $(p,q \geq 0)$ omzet in een woord $aPbqa^r$ met $r = pq$. Bewijs dat uw constructie correct is.

1.10. Construeer een herschrijfsysteem dat controleert of een gegeven woord a^n $(n \geq 2)$ samengesteld (d.i. niet-priem) is. Bewijs dat uw constructie correct is.

1.11. Construeer een contextgevoelige grammatica $G = (V, \Sigma, P, S)$ met $L(G) = \{a^n\$a^m \mid m = 2^n, n \geq 0\}$ en $\Sigma = \{a,\$\}$. Bewijs dat uw constructie correct is.

1.12. Bewijs dat voor elke contextvrije grammatica $G = (V,\Sigma,P,S)$, de taal $\{\alpha \mid \alpha \text{ in } V^* \text{ en } S \overset{*}{\Longrightarrow} \alpha\}$ contextvrij is.

2.1. Bewijs dat als $\text{card}(V) = 1$ het PCP over V algoritmisch oplosbaar is.

2.2. Zij Σ een alfabet met tenminste twee letters en $\underline{x} = (x_1,\ldots,x_n)$, $\underline{y} = (y_1,\ldots,y_n)$, twee lijsten van niet-lege woorden over Σ. Ga na of het Post correspondentieprobleem algoritmisch oplosbaar is als

(1) $n = 1$

(2) $n \geq 1$ en voor elke i $(1 \leq i \leq n)$ geldt: $|x_i| = |y_i|$.

Motiveer steeds uw antwoord.

2.3. Ga na of het volgende probleem algoritmisch onoplosbaar is. Gegeven een alfabet Σ met tenminste twee letters en $\underline{x} = (x_1,\ldots,x_n)$, $\underline{y} = (y_1,\ldots,y_n)$, twee lijsten van niet-lege woorden over Σ; bestaan er getallen $k \geqslant 1$, $m \geqslant 1$, $i_1,\ldots,i_k$, $j_1,\ldots,j_m$ met $1 \leqslant i_p, j_q \leqslant n$ $(1 \leqslant p \leqslant k, 1 \leqslant q \leqslant m)$ zodanig dat

$$x_{i_1}\ldots x_{i_k} = y_{j_1}\ldots y_{j_m}?$$

2.4. Ga na of het volgende probleem algoritmisch onoplosbaar is. Gegeven een alfabet Σ met tenminste twee letters en $\underline{x} = (x_1,\ldots,x_n)$, $\underline{y} = (y_1,\ldots,y_n)$, twee lijsten van niet-lege woorden over Σ; bestaan er getallen $k \geqslant 1$, $i_1,..,i_k$, $j_1,..,j_k$ met $1 \leqslant i_p, j_q \leqslant n$ $(1\leqslant p,q\leqslant k)$ zodanig dat

$$x_{i_1}\ldots x_{i_k} = y_{j_1}\ldots y_{j_k}?$$

2.5. Zij $G = (V,\Sigma,P,S)$ een type 0 grammatica. Toon aan dat $L(G)$ beslisbaar is dan en slechts dan als er een berekenbare functie $f: \Sigma^* \to N$ bestaat, zodanig dat $w \in L(G)$ dan en slechts dan als $\exists k \leqslant f(w)\,[S \overset{k}{\Rightarrow} w]$.

2.6. Zij $\Sigma = \{a,b,c\}$. Definieer voor elke $j \geqslant 1$, $h(j) = ba^j$, en voor elk geordend n-tal $\underline{w} = (w_1,..,w_n)$ van woorden w_i in $\{a,b\}^+$ $(1\leqslant i\leqslant n)$
$H(\underline{w}) = \{ c w_{i_1} w_{i_2} \ldots w_{i_k} c\, h(i_k)\ldots h(i_1) \mid k \geqslant 1;$
$1 \leqslant i_1, \ldots, i_k \leqslant n\}$.
Bewijs de volgende beweringen:

(1) Voor elke $\underline{w}$ geldt dat $H(\underline{w})$ een contextvrije taal over Σ is.

(2) Als $\underline{x} = (x_1,\ldots,x_n)$ en $\underline{y} = (y_1,\ldots,y_n)$ geordende n-tallen van woorden in $\{a,b\}^+$ zijn, dan is $H(\underline{x})/H(\underline{y})$ gelijk aan of $\{\lambda\}$ of $\emptyset$ al naar gelang er al dan niet een oplossing van het Post correspondentie probleem voor $\underline{x}$ en $\underline{y}$ bestaat (N.B. de operatie L_1/L_2 wordt gedefinieerd door $L_1/L_2 = \{u \mid uv \text{ in } L_1 \text{ en } v \text{ in } L_2\}$).

3.1. Zij $G = (V, \Sigma, P, S)$ een contextvrije grammatica waarvan de taal $L(G)$ oneindig is. Bewijs dat er een A in $V\setminus\Sigma$ is en woorden x_1, x_2, w_1, w_2 en w in Σ^* bestaan met $S \overset{*}{\Longrightarrow} x_1Ax_2$, $A \overset{*}{\Longrightarrow} w_1Aw_2$, $A \overset{*}{\Longrightarrow} w$ en $w_1w_2 \neq \lambda$.

3.2. Bewijs dat $\{a^ib^ja^k \mid i,j,k \geq 1\} \setminus \{a^nb^na^n \mid n \geq 1\}$ een contextvrije taal is.

3.3. Toon aan dat Pascal niet contextvrij is.

3.4. Construeer een contextvrije grammatica, die de taal $\{w \mid w \text{ in } \{a,b\}^*, \text{ en } \#(a,w) = \#(b,w)\} \setminus \{a^nb^n \mid n \geq 0\}$ voortbrengt. Bewijs dat uw constructie correct is.

3.5. Beschouw het herschrijfsysteem $H = (V,P)$ met $V = \{),(,[,]\}$ en $P = \{[\,] \to \lambda,\ (\,) \to \lambda\}$. Definieer nu een taal D over V door middel van $D = \{x \mid x \text{ in } V^* \text{ en } x \overset{*}{\underset{H}{\Longrightarrow}} \lambda\}$. Geef een intuïtieve beschrijving van de elementen van D. Construeer vervolgens een contextvrije grammatica voor D. Bewijs dat uw constructie correct is.

3.6. Construeer een contextvrije grammatica die de taal $\{a^nb^m \mid 1 \leq n \leq m \leq 3n\}$ voortbrengt. Bewijs dat uw constructie correct is.

3.7. Construeer een contextvrije grammatica die de taal $\{x\$y \mid x,y \text{ in } \{a,b\}^*, \text{ en } \#(a,x) = \#(a,y)\}$ voortbrengt. Bewijs dat uw constructie correct is.

3.8. Bewijs dat $\{x \$ x \mid x \text{ in } \{a,b\}^*\}$ niet contextvrij is.

3.9. Bewijs dat $\{a^nb^m \mid 0 \leq m \leq n^2;\ n \geq 1\}$ niet contextvrij is.

3.10. Bewijs dat $\{a^n b^n x \mid n > 1, |x| = n;\ x \text{ in } \{b,c\}^*\}$ niet contextvrij is.

3.11. Bewijs dat $\{x \mid x \text{ in } \{a,b\}^*, \text{ en } \#(a,x) = \#(b,x)\}$ niet regulier is.

3.12. Bewijs dat $\{xx^s \mid x \text{ in } \{a,b\}^+\}$ niet regulier is.

3.13. Bewijs dat $\{a^n b^m \mid 1 \leq m \leq n\}$ niet regulier is.

3.14. Beschouw de taal $L = \bigcup \{L_1.0^n.L_2.1^n.L_3 \mid n > 1\}$ waarbij L_1, L_2 en L_3 niet-lege talen over een of ander alfabet Σ zijn. Kan men op één of andere manier Σ, L_1, L_2 en L_3 zodanig kiezen dat L regulier is? Motiveer uw antwoord.

3.15. Bewijs dat voor elke reguliere grammatica G met eindalfabet Σ, het probleem $[w_1, w_2 \in \Sigma^*: \Sigma^* \backslash L(G) = \{w_1, w_2\}]$ algoritmisch oplosbaar is.

3.16. Bewijs dat voor elk alfabet Σ het probleem $[G \in \mathcal{G}_3: \Sigma^* \backslash L(G)$ is oneindig$]$ algoritmisch oplosbaar is.

3.17. Bewijs dat het probleem $[G_1, G_2 \in \mathcal{G}_3: L(G_1) \cap L(G_2)$ is oneindig$]$ algoritmisch oplosbaar is.

3.18. Bewijs dat het probleem $[G \in \mathcal{G}_2, \alpha \in R_\Sigma: L(G) \subseteq \rho(\alpha)]$ algoritmisch oplosbaar is.

4.1. Construeer deterministische eindige automaten A_i (i = 1, 2, 3) met $L(A_i) = L_i$ waarbij $\Sigma = \{a,b\}$ het invoeralfabet is en

$L_1 = \{x \mid x \text{ in } \Sigma^*, \text{ en } |x| \text{ is deelbaar door } 3\}$,
$L_2 = \{a\ x\ b \mid x \text{ in } \Sigma^*\}$,
$L_3 = \{x \mid x \text{ in } \Sigma^*, \text{ en } \#(aaa, x) = 0\}$.

4.2. Construeer voor de grammatica $G = (V, \Sigma, P, S)$ waarbij $\Sigma = \{a,b\}$ en $P = \{S \to aA, S \to aaA, S \to a, S \to b, A \to bbS, A \to b\}$ een niet-deterministische automaat A_1 zodanig dat $L(A_1) = L(G)$. Construeer vervolgens een deterministische automaat A_2 met $L(A_2) = L(A_1)$. Bewijs dat uw constructies correct zijn.

4.3. Construeer voor de grammatica $G = (V, \Sigma, P, S)$ waarbij $\Sigma = \{a,b,c\}$ en $P = \{S \to abS, S \to ba, S \to aA, S \to abB, A \to bbS, B \to bB, B \to c\}$ een niet-deterministische automaat A_1 zodanig dat $L(A_1) = L(G)$. Construeer vervolgens een deterministische automaat A_2 met $L(A_2) = L(A_1)$. Bewijs dat uw constructies correct zijn.

4.4. Zij E een relatie op een verzameling V, d.w.z. E is een deelverzameling van V×V. Zoals gebruikelijk schrijven we xEy in plaats van: (x,y) in E. E heet een <u>equivalentierelatie</u> als voor alle x,y en z uit V geldt:

(i) xEx,
(ii) xEy d.e.s.d. als yEx,
(iii) als xEy en yEz dan geldt ook xEz.

Een equivalentierelatie E op V deelt V op in een aantal onderling disjuncte "equivalentie-klassen" $[x] = \{y \mid yEx\}$; dit aantal wordt de <u>index</u> i(E) van E genoemd. We zeggen dat E_1 een <u>verfijning</u> van E_2 is indien E_1 een deelverzameling van E_2 is (d.w.z. xE_1y impliceert dat xE_2y).

(1) Bewijs dat voor equivalentierelaties E_1 en E_2 op V geldt dat, als E_1 een verfijning van E_2 is, dan $i(E_1) \geq i(E_2)$.

Een equivalentierelatie C op Σ^* heet een rechter-congruentierelatie als xCy impliceert dat voor alle z in Σ^* ook (xz)C(yz) geldt. Zij nu L een taal over Σ. Dan is C een verfijning van L d.e.s.d. als xCy impliceert dat: x in L d.e.s.d. als y in L (D.w.z. als xCy geldt, dan zijn of zowel x als y in L, of zowel x als y niet in L).

(2) Bewijs dat C een verfijning van L is, d.e.s.d. als L de vereniging van een zeker aantal equivalentieklassen van C is.

Zij L een taal over Σ. Definieer de equivalentierelatie C_L geïnduceerd door L d.m.v. xC_Ly d.e.s.d. als voor alle z in Σ^*: xz in L, d.e.s.d. als yz in L.

(1) Bewijs de volgende stelling.

Stelling (a) C_L is een rechter-congruentierelatie.

(b) C_L is een verfijning van L.

(c) Als C één of andere rechter-congruentierelatie is die een verfijning van L is, dan geldt dat C een deelverzameling van C_L is.

(2) Bewijs tenslotte de volgende stelling:

Stelling Zij L een taal over Σ. Dan zijn de volgende beweringen gelijkwaardig:

(a) L is regulier.

(b) Er bestaat een rechter-congruentierelatie op Σ^* die een verfijning van L is.

(c) De index $i(C_L)$ van C_L is eindig.

4.5. Zij $A = (\{q_1,q_2,q_3,q_4\}, \{0,1\}, \sigma, q_1, \{q_3,q_4\})$ een eindige automaat waarvan de transitiefunctie σ weergegeven is in bijgaande tabel.

	0	1
q_1	q_2	q_3
q_2	q_2	q_4
q_3	q_1	q_4
q_4	q_1	q_4

(1) Teken het transitiediagram van A.

(2) Construeer een type 3 grammatica G zodanig dat L(G) = L(A). Bewijs dat uw constructie correct is.

4.6. Zij L de taal bestaande uit alle woorden over $\{0,1\}$ waarin de woorden 10100, 10110 en 010111 elk ten minste één keer als deelwoord voorkomen.
(1) Construeer een niet-deterministische eindige automaat A met L(A)=L.
(2) Construeer een deterministische eindige automaat B met L(B) = L.
Bewijs dat uw constructie correct is.

4.7. Zij $A = (\{q_1,q_2,q_3,q_4\}, \{0,1\}, \sigma, \{q_1\}, \{q_2,q_3\})$ een eindige automaat waarvan de transitiefunctie σ weergegeven is in bijgaande tabel.
(1) Teken het transitiediagram van A.
(2) Bepaal L(A).
(3) Construeer een deterministische eindige automaat die L(A) accepteert. Toon aan dat uw constructie correct is.

	0	1
q_1	$\{q_2,q_3\}$	$\emptyset$
q_2	$\emptyset$	$\{q_1,q_3\}$
q_3	$\{q_4\}$	$\{q_4\}$
q_4	$\emptyset$	$\{q_2,q_4\}$

4.8. Zij L de taal bestaande uit alle woorden over $\{0,1\}$ waarin 1000, 1011 en 1010 elk tenminste één keer als deelwoord voorkomen.
(1) Construeer een deterministische eindige automaat die de taal $\{0,1\}^* \setminus L$ accepteert.
(2) Construeer een deterministische eindige automaat die de taal $(\{0,1\}^* \setminus L)^s$ accepteert.
Bewijs dat uw constructies correct zijn.

4.9. Zij $A = (\{q_1,q_2,q_3\}, \{0,1\}, \sigma, \{q_1\}, \{q_2\})$ een niet-deterministische eindige automaat waarvan de transitiefunctie weergegeven is in bijgaande de tabel.
(1) Teken het transitiediagram van A.
(2) Construeer een type 3 grammatica die de taal $\{0,1\}^* \setminus L(A)$ voortbrengt.
(3) Construeer een type 3 grammatica die de taal $(\{0,1\}^* \setminus L(A))^s$ voortbrengt.
Toon aan dat uw constructies correct zijn.

	0	1
q_1	$\{q_2\}$	$\{q_1,q_3\}$
q_2	$\{q_3\}$	$\{q_1\}$
q_3	$\{q_1\}$	$\{q_3\}$

4.10. Zij A = $(\{q_1,q_2,q_3,q_4\}, \{0,1\}, \sigma, \{q_1,q_2\}, \{q_4\})$ een niet-deterministische eindige automaat waarvan de transitiefunctie σ weergegeven is bijgaande tabel.

	0	1
q_1	$\{q_2,q_3\}$	$\{q_3,q_4\}$
q_2	$\emptyset$	$\{q_4\}$
q_3	$\{q_1,q_2\}$	$\{q_3\}$
q_4	$\emptyset$	$\{q_1,q_3\}$

(1) Teken het transitiediagram van A.

(2) Construeer een type 3 grammatica die de taal $\{0,1\}^* \setminus L(A)$ voortbrengt.

(3) Construeer een type 3 grammatica die de taal $(\{0,1\}^* \setminus L(A))^s$ voortbrengt.

Toon aan dat uw constructies correct zijn.

4.11. Beschouw de operatie Θ: $\{a,b,c\}^2 \to \{a,b,c\}$ gegeven door de tabel:

	a	b	c
a	a	a	c
b	c	a	b
c	b	c	a

D.w.z. dat bijvoorbeeld $a \Theta c = c$ en $c \Theta a = b$. Merk op dat in het algemeen de gelijkheden $x\Theta y = y\Theta x$ en $x \Theta(y \Theta z) = (x \Theta y) \Theta z$ niet gelden. Construeer nu een niet-deterministische eindige automaat, die de taal $\{x_1x_2\ldots x_n \mid n \geqslant 1;\ x_i \text{ in } \{a,b,c\} \text{ voor } 1 \leqslant i \leqslant n;$

$x_1\Theta(x_2\Theta(x_3\Theta\ldots\ldots\ldots\Theta(x_{n-1}\Theta x_n)\ldots\ldots\ldots)) =$
$((\ldots\ldots(x_1\Theta x_2)\Theta\ldots\ldots \quad \Theta x_{n-2})\Theta x_{n-1})\Theta x_n\}$ accepteert. Bewijs dat uw constructie correct is.

4.12. Zij G de grammatica $G = (V,\Sigma,P,S)$ met $\Sigma = \{a,b\}$, $V \setminus \Sigma = \{S,A,B,C\}$ en $P = \{S \to aS,\ S \to bS,\ S \to aA,\ S \to bC,\ A \to bB,\ C \to aB,\ B \to aB,\ B \to bB,\ B \to \lambda\}$. Bepaal L(G) en toon aan dat G dubbelzinnig is. Construeer vervolgens een eindige automaat die L(G) accepteert. Bewijs dat uw constructie correct is.

4.13. Zij A een gegeven eindige automaat met n toestanden. Bewijs dat L(A) oneindig is d.e.s.d. als er een woord w in L(A) bestaat met $n \leqslant |w| \leqslant 2n$.

5.1. Bewijs de volgende gelijkwaardigheden waarin α en β reguliere expressies en a, b en c symbolen zijn:
(1) $(\alpha+\beta)^* \equiv (\alpha^*+\beta)^*$,
(2) $\alpha^* \equiv \Lambda + \alpha + \alpha^2 + \alpha^3\alpha^*$,
(3) $((a+b+c)^*b+c)^*c \equiv (\Lambda+(a+b+c)^*b)cc^*$.

5.2. Construeer een type 3 grammatica voor de taal gedefiniëerd door de reguliere expressie $(aa + bb^*a + abb^*a)^*$.

5.3. Construeer een reguliere expressie die de taal definiëert, die voortgebracht wordt door de grammatica $G = (V, \Sigma, P, S)$ met $\Sigma = \{a,b\}$, $V\setminus\Sigma = \{S, A, B\}$ en $P = \{S \to abA, S \to aB, A \to aaS, A \to bA, B \to aA, B \to b\}$.

5.4. Zij $\Sigma = \{0,1\}$ en $A = (\{q_0, q_1, q_2\}, \Sigma, \sigma, \{q_0\}, \{q_0\})$ een niet-deterministische eindige automaat. De transitiefunctie is weergegeven in bijgaande tabel.

	0	1
q_0	$\{q_1\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\{q_2,q_0\}$
q_2	$\{q_2\}$	$\{q_0\}$

(1) Teken het transitiediagram van A.
(2) Construeer een reguliere expressie die L(A) definieert. Bewijs dat uw constructie correct is.

5.5. Construeer een rechts-lineaire grammatica voor de taal gedefinieerd door de reguliere expressie $(ab)^* + (bc + a)^*b$. Toon aan dat uw constructie correct is.

5.6. Construeer een deterministische eindige automaat die de taal accepteert, gedefinieerd door de reguliere expressie $(11+0)^*(00+1)^*$. Toon aan dat uw constructie correct is.

5.7. Construeer een deterministische eindige automaat, die de taal accepteert, gedefinieerd door de reguliere expressie $10+(0+11)0^*1$. Toon aan dat uw constructie correct is.

5.8. Zij $A = (\{q_1,q_2,q_3\},\{0,1\},\sigma,q_1,\{q_1\})$ een eindige automaat waarvan de transitiefunctie σ weergegeven is in bijgaande tabel.

σ	0	1
q_1	q_1	q_2
q_2	q_3	q_2
q_3	q_1	q_2

(1) Teken het transitiediagram van A.

(2) Construeer een reguliere expressie α zodanig dat $\rho(\alpha) = L(A)$.

(3) Construeer een reguliere expressie β zodanig dat $\rho(\beta) = \{0,1\}^* \setminus L(A)$.

Toon aan dat uw constructies correct zijn.

5.9. Beschouw de vergelijking $X = AX + B$ in één onbekende X. Hierin zijn A en B talen over een alfabet Σ, terwijl + dezelfde betekenis heeft als in de definitie van reguliere expressie.

(1) Bewijs dat A^*B een oplossing van deze vergelijking is.

(2) Bewijs dat voor elke oplossing S geldt: A^*B is een deelverzameling van S.

(3) Bewijs dat deze vergelijking meer dan één oplossing heeft dan en slechts dan als A λ bevat.

5.10. <u>Een-dimensionaal solitaire spel</u>. Het speelbord is een oneindig lange lat waarin op regelmatige afstanden gaten zijn geboord. In de beginsituatie zijn er eindig veel pennen geplaatst in verder willekeurig gekozen gaten. Voorbeeld:

. . . 0 0 0 1 0 0 1 1 0 1 0 0 0 . . .

Daar er maar eindig veel pennen zijn, bestaan er buitenste pennen, d.w.z. pennen links (resp. rechts) waarvan geen andere pennen meer zijn. Elke spelsituatie kan worden beschreven als een eindige symboolrij over het alfabet $\{0,1\}$, door een gat als 0 en een pen als 1 te noteren en de gaten links (resp. rechts) van de buitenste pennen niet op te schrijven. De hierboven geschetste spelsituatie noteren we dus als 1001101. Een zet bestaat uit het laten springen van een pen over een buurpen naar een daarnaast gelegen vrije plaats. De buurpen verdwijnt daarbij

Voor de zet:	Na de zet:
1 1 0	0 0 1
0 1 1	1 0 0

Het spel wordt gespeeld door opeenvolgend mogelijke zetten te doen totdat er geen zetten meer kunnen worden gedaan. Dan is er een eindsituatie bereikt. Het spel is gewonnen als de bereikte eindsituatie bestaat uit een bord waarin nog precies één pen staat. We noemen een woord w over $\{0,1\}$ winnend als w begint en eindigt met een 1 en als het spel vanuit de door w bepaalde beginsituatie kan worden gewonnen. Zij R de verzameling van alle winnende woorden. Construeer een reguliere expressie die R definiëert. Bewijs dat uw constructie correct is.

5.11. Zij $A = (\{q_1,q_2,q_3\}, \{0,1\}, \sigma, \{q_1\}, \{q_2\})$ een niet-deterministische eindige automaat waarvan de transitiefunctie weergegeven is in bijgaande tabel.

	0	1
q_1	$\{q_2\}$	$\{q_1,q_3\}$
q_2	$\{q_3\}$	$\{q_1\}$
q_3	$\{q_1\}$	$\{q_3\}$

(1) Teken het transitiediagram van A.

(2) Construeer een type 3 grammatica die de doorsnede van $\rho(\alpha)$ en L(A) voortbrengt, waarbij α de reguliere expressie $11(0+1)^*00$ is.

(3) Construeer een type 3 grammatica die de doorsnede van $\rho(\beta)$ en $(L(A))^s$ voortbrengt, waarbij β de reguliere expressie $010(0+1)^*11$ is.

Toon aan dat uw constructies correct zijn.

5.12. Zij $G = (V,\Sigma,P,S)$ de grammatica gegeven door $\Sigma = \{a,b\}$, $V\setminus\Sigma = \{S,A,B,C,D\}$ en $P = \{S \to A,\ A \to Ba,\ A \to Ca,\ A \to Cb,\ B \to Da,\ B \to Cb,\ B \to Aa,\ C \to Ca,\ D \to Ca,\ D \to Cb,\ D \to Bb,\ B \to a,\ C \to a,\ D \to b\}$.

(1) Construeer een (niet-deterministische) eindige automaat die de taal $\Sigma^*\setminus L(G)$ accepteert.

(2) Construeer een (niet-deterministische) eindige automaat die de doorsnede van $\rho(\alpha)$ en $(L(G))^s$ accepteert, waarbij α de reguliere expressie $aa(a+b)^*$ is.

Bewijs dat uw constructies correct zijn.

6.1. Construeer een stapelautomaat A met invoeralfabet $\{a,b,c\}$, zodanig dat $L_F(A) = \{x \mid x \text{ in } a^*b^*c^*, \text{ en } \#(a,x) = \#(b,x) + \#(c,x)\}$. Toon aan dat uw constructie correct is. Kan A deterministisch gekozen worden? Wat is de minimale benodigde grootte van het stapelalfabet? Motiveer uw antwoorden. Beantwoord nu dezelfde vragen voor A' met $L_F(A') = \{x \mid x \text{ in } \{a,b,c\}^*, \text{ en } \#(a,x) = \#(b,x) + \#(c,x)\}$.

6.2. Construeer een stapelautomaat A die de taal gegenereerd door de contextvrije grammatica $(\{S, A, a, b\}, \{a,b\}, \{S \to aAA, A \to bS, A \to aS, A \to a\}, S)$ accepteert (d.m.v. accepterende toestanden). Bewijs dat uw constructie correct is. Kan A deterministisch gekozen worden? Wat is het minimale aantal toestanden van A? Motiveer uw antwoorden.

6.3. Construeer een contextvrije grammatica voor de taal $L_\Lambda(A)$ met $A = (\{q_0, q_1\}, \{0,1\}, \{Z_0,X\}, \sigma, q_0, Z_0, \emptyset)$ waarbij

$\sigma(q_0, 1, Z_0) = \{(q_0, XZ_0)\}$ $\qquad$ $\sigma(q_0, \lambda, Z_0) = \{(q_0, \lambda)\}$
$\sigma(q_0, 1, X) = \{(q_0, XX)\}$ $\qquad$ $\sigma(q_1, 1, X) = \{(q_1, \lambda)\}$
$\sigma(q_0, 0, X) = \{(q_1, X)\}$ $\qquad$ $\sigma(q_1, 0, Z_0) = \{(q_0, Z_0)\}$

Bewijs dat uw constructie correct is.

6.4. Gegeven een willekeurige stapelautomaat A. Bewijs dat er een stapelautomaat A_1 met slechts één toestand bestaat zodanig dat $L_\Lambda(A) = L_\Lambda(A_1)$.

6.5. Gegeven een willekeurige stapelautomaat A. Bewijs dat er een stapelautomaat A_2 met slechts twee toestanden bestaat zodanig dat $L_F(A) = L_F(A_2)$. Onder welke voorwaarden bestaat er een stapelautomaat A_1 met één toestand zodanig dat $L_F(A) = L_F(A_1)$?

6.6. Gegeven een stapelautomaat A met de eigenschap, dat er een natuurlijk getal k is zodanig dat op elk tijdstip van elke tot accepteren leidende berekening hoogstends k symbolen op de stapel staan. Bewijs dat $L_F(A)$ regulier is.

6.7. Zij G = ($\{A, B, a, b\}$, $\{a,b\}$, P, A) met P = $\{ A \to Ba, Aa \to Bb, B \to bA, Ab \to \lambda, B \to BB, B \to b, A \to a\}$ een grammatica. Bepaal L(G). Construeer een stapelautomaat die L(G) accepteert. Bewijs dat uw constructie correct is. Is L(G) regulier?

6.8. Zij L een contextvrije taal over een alfabet Σ. Definieer:

$$L_1 = \{a_1a_3a_5\cdots a_{2k+1} \mid a_1a_2a_3a_4\cdots a_{2k+2} \text{ in L; voor zekere } a_2,a_4,\ldots,a_{2k+2};\ a_i \text{ in } \Sigma,\ 1\leq i\leq 2k+1\}$$

Bewijs dat L_1 contextvrij is.

6.9. Bewijs dat de taal $\{w_1cw_2c..cw_mccw_i^s \mid 1\leq i\leq m;\ w_j \text{ in } \{0,1\}^*,\ 1 \leq j \leq m\}$ contextvrij is.

6.10. Zij L de taal over $\{[,],(,)\}$ gedefinieerd door:

(i) λ in L,
(ii) als w in L is, dan zijn ook [w] en (w) in L,
(iii) als x en y in L zijn, dan is ook xy in L,
(iv) niets anders zit in L.

Construeer een stapelautomaat die L accepteert. Bewijs dat uw constructie correct is.

6.11. Construeer een deterministische stapelautomaat A zodanig dat $L_F(A) = \{xcycx^s \mid x,y \text{ in } \{a,b\}^*\} \cup \{xcydy^s \mid x,y \text{ in } \{a,b\}^*\}$. Bewijs dat uw constructie correct is.

6.12. Construeer een deterministische stapelautomaat die de taal $\{a^ib^jca^m \mid i,j \geq 1;\ i+j \equiv 1 \pmod 2;\ m = i+j\} \cup \{a^ib^jda^n \mid i,j \geq 1;\ i+j \equiv 0 \pmod 2;\ n = \frac{1}{2}(i+j)\}$ accepteert. Bewijs dat uw constructie correct is.

6.13. Construeer een deterministische stapelautomaat die de taal $\{0^k\$w \mid w \text{ in } \{0,1\}^*;\ k = \#(0,w)\} \cup \{0^p\$\$w \mid w \text{ in } \{0,1\}^*;\ p = \#(1,w)\}$ accepteert. Bewijs dat uw constructie correct is.

7.1. Zij A,B en C willekeurige talen over een alfabet Σ. Welke van de volgende gelijkheden zijn (on)juist? Geef of een bewijs of een tegenvoorbeeld.

(1) $(AB)^* = A^*B^*$
(2) $A^*(B \cap C)^* = (AB \cap AC)^*$
(3) $(A^*B^*)^* = (A \cup B)^*$

7.2. Een taal L is een <u>palindroom taal</u> als voor alle w uit L geldt $w = w^s$.

(1) Bewijs dat als L een palindroom taal is dan is ook L^s een palindroom taal.
(2) Ga na of de volgende beweringen (on)juist zijn. Geef steeds of een bewijs of een tegenvoorbeeld.
  (2.1) "Als $L = L^s$ dan is L een palindroom taal".
  (2.2) Zij $L' = L \cap L^s$ voor zekere taal L. "Als L een palindroom taal is, dan is L' dat ook".
  (2.3) "Als L' een palindroom taal is, dan is L dat ook".
  (2.4) "Als $w = w^s$ dan is de taal w^* een palindroom taal".

7.3. Welke van de volgende gelijkheden zijn geldig voor alle talen L_1, L_2, L_3 en alle homomorfismen h? Motiveer uw antwoord.

(a) $L_1(L_2 \cup L_3) = L_1L_2 \cup L_1L_3$
(b) $(L_1 \cap L_2)^* = L_1^* \cap L_2^*$
(c) $(L_1L_2)^* L_1 = L_1(L_2L_1)^*$
(d) $h^{-1}(L_1 \cap L_2) = h^{-1}(L_1) \cap h^{-1}(L_2)$

waarbij voor iedere taal L, $h^{-1}(L)$ gedefiniëerd is als $h^{-1}(L) = \{x \mid h(x)$ in $L\}$. Indien een gelijkheid ongeldig is, probeer dan voorwaarden te vinden, waaraan L_1, L_2, L_3 en/of h dienen te voldoen opdat dan de gelijkheid geldt.

7.4. Verander de definitie van type 2 grammatica zodanig, dat in plaats van en startsymbool er nu een type 2 taal I van startwoorden (over V) is. Voor zo'n grammatica $G = (V, \Sigma, P, I)$ met I in $\mathcal{L}_2$, blijven de type 2 restricties op de vorm van de regels dezelfde, terwijl L(G) nu wordt gedefiniëerd door $L(G) = \{x \mid x \text{ in } \Sigma^*$, en er is een i in I met $i \overset{*}{\Rightarrow} x\}$. Bewijs dat de familie $\mathcal{L}_2$ dezelfde blijft onder deze verandering.

7.5. Twee talen L_1 en L_2 heten <u>bijna gelijk</u> indien $L_1 \setminus L_2$ en $L_2 \setminus L_1$ eindig zijn. Bewijs dat een taal L regulier is dan en slechts dan als L bijna gelijk is aan een reguliere taal.

7.6. Bewijs de (on)juistheid van de volgende beweringen:

(1) $h^{-1}(h(L)) = L$, en
(2) $h(h^{-1}(L)) = L$.

waarin L een willekeurige taal, h een willekeurig homomorfisme is, en $h^{-1}(S)$ gedefinieerd wordt door $h^{-1}(S) = \{x \mid h(x) \text{ in } S\}$.

7.7. Definieer voor elke taal L over Σ, $SUB(L) = \{w \mid xwy$ in L; voor zekere x,y in $\Sigma^*\}$. Bewijs dat SUB(L) regulier is, indien L regulier is.

7.8. Zij L een contextvrije taal. Bewijs, dat de taal $\{x$ in L $\mid |x|$ is oneven$\}$ ook contextvrij is.

7.9. Bewijs dat voor elke funktie f van talen naar talen die voldoet aan:

$$f(\{\lambda\}) = \{\lambda\},\ f(L_1 . L_2) = f(L_1).f(L_2),\ \text{en}\ f(\bigcup_{n=0}^{\infty} L_n) = \bigcup_{n=0}^{\infty} f(L_n)$$

een substitutie is.

7.10. Is de volgende bewering waar of onwaar? "Als de vereniging van L_1 en L_2 context-vrij is en L_1 regulier is, dan is L_2 contextvrij". Geef of een bewijs of een tegenvoorbeeld.

7.11. Wat is het verband tussen $\mathcal{L}_3$ en de kleinste klasse $\mathcal{F}$ van talen , die voldoet aan

(1) $\mathcal{F}$ bevat alle eindige talen, en
(2) $\mathcal{F}$ is gesloten onder vereniging, doorsnijding en complement.

(Een klasse $\mathcal{F}$ is gesloten onder complement, indien voor alle L in $\mathcal{F}$ met L over Σ ook $\Sigma^* \backslash L$ in $\mathcal{F}$ is). Is $\mathcal{L}_3$ gesloten onder (aftelbaar) oneindige vereniging? Motiveer uw antwoorden.

11. TENTAMENS MET UITWERKINGEN

TECHNISCHE HOGESCHOOL DELFT
Onderafdeling der Wiskunde en Informatica
Julianalaan 132
2628 BL DELFT

TENTAMEN THEORETISCHE INFORMATICA (a137)
Woensdag 20 juni 1984
09.00 - 12.00 uur

In het volgende wordt met "bewijs ...." aangegeven dat een (correct en volledig) bewijs moet worden gegeven. Met "beargumenteer ...." wordt bedoeld, dat men een steekhoudende redenering dient te geven, waarin kan worden volstaan met het formuleren van beweringen en inductie-hypothesen, terwijl het inductiebewijs niet behoeft te worden uitgeschreven; evenzo kan men volstaan met het formuleren van invarianten (of invariante beweringen) en behoeft men de verificatie, dat deze beweringen inderdaad voor een gegeven herschrijfsysteem invariant zijn, niet in detail uit te werken.

1. Bewijs de volgende gelijkwaardigheid (a en b zijn symbolen):

$$\Lambda + (a+b)^*b \equiv ((a+b)^*b)^*$$

2. Zij $G = (\{S,A,B,a,b\}, \{a,b\}, P, S)$ met $P = \{S \to aaS, S \to abA, S \to aabA, A \to B, B \to aaB, B \to \lambda\}$ een grammatica.

   (2.1.) Construeer een niet-deterministische eindige automaat A_1, zodanig dat $L(A_1) = L(G)$.

   (2.2.) Zij $A_2 = (\{p_0,p_1\}, \{a,b\}\ \sigma, \{p_0\}, \{p_1\})$ een eindige automaat, waarbij σ aldus is bepaald:
   $\sigma(p_0,a) = \sigma(p_0,b) = \{p_1\}$,
   $\sigma(p_1,a) = \sigma(p_1,b) = \{p_0\}$.
   Construeer een eindige automaat A_3 zodanig dat $L(A_3) = L(G) \cap L(A_2)$.

   Beargumenteer de correctheid van de door u geconstrueerde automaten A_1 en A_3.

3. Ontwerp een contextvrije grammatica G zodanig dat $L(G) = \{a^n b^m c^k \mid n,k \geq 0,\ m = n+k\}$.
   Bewijs de correctheid van uw grammatica.

4. Bewijs dat de taal $L = \{w \in \{a,b,c\}^* \mid (\#(a,w))^2 = \#(b,w),\ \#(c,w) > 0\}$ niet contextvrij is.

5. Zij $A = (Q,\Sigma,\Gamma,\sigma,q_0,\$,F)$ een stapelautomaat met $\Sigma = \{a,b\}$, $\Gamma = \{\$,1\}$, $Q = \{q_0,q_1,q_2,q_3\}$, $F = \{q_3\}$ en

$\sigma(q_0,a,\$) = \{(q_0,1\$)\}$,
$\sigma(q_0,a,1) = \{(q_0,11)\}$,
$\sigma(q_0,\lambda,1) = \{(q_1,\lambda)\}$,
$\sigma(q_2,\lambda,\$) = \{(q_3,\lambda)\}$.
$\sigma(q_1,b,1) = \{(q_2,\lambda)\}$,
$\sigma(q_1,b,\$) = \{(q_3,\lambda)\}$,
$\sigma(q_2,\lambda,1) = \{(q_1,\lambda)\}$,

(5.1.) Teken het transitiediagram van A.
(5.2.) Bepaal $L_F(A)$. Beargumenteer uw antwoord.
(5.3.) Bepaal $L_\Lambda(A)$. Beargumenteer uw antwoord.
(5.4.) Is A deterministisch? Beargumenteer uw antwoord.
(5.5.) Construeer een contextvrije grammatica G zodanig dat $L(G) = \{x \in \Sigma^* \mid x \in L_F(A);\ |x| \text{ is een 3-voud}\}$. Beargumenteer de correctheid van uw grammatica.

6. Zij Σ en Δ alfabetten, en $HOM(\Sigma,\Delta) = \{f \mid f: \Sigma^* \to \Delta^* \text{ is een } \lambda\text{-vrij homomorfisme}\}$. Definieer voor elk paar f en g uit $HOM(\Sigma,\Delta)$ de taal E_{fg} over Σ door middel van

$$E_{fg} = \{x \in \Sigma^+ \mid f(x) = g(x)\}.$$

(6.1.) Bewijs, dat voor alle f en g uit $HOM(\Sigma,\Delta)$ geldt:
$E_{fg} = E_{gf}$, en $E_{fg}^+ = E_{fg}$.
(6.2.) Bewijs, dat het probleem $U = [f,g \in HOM(\Sigma,\Delta): E_{fg} \neq \emptyset]$ algoritmisch onoplosbaar is.
(6.3.) Bewijs, dat als $card(\Sigma) = 1$, dan is het probleem $U_1 = [f,g \in HOM(\Sigma,\Delta): E_{fg} = \emptyset]$ algoritmisch oplosbaar.
(6.4.) Bewijs, dat voor alle alfabetten Σ en Δ en voor alle f en g uit $HOM(\Sigma,\Delta)$ de taal E_{fg} beslisbaar is.

20.06.1984

1. a) Voor iedere verzameling L geldt $L^* = \bigcup_{i \geqslant 0} L^i$, zodat $L^0 \cup L = \{\lambda\} \cup L \subseteq L^*$. Dus $\rho(\Lambda + (a+b)^*b) = \rho(\Lambda) \cup \rho((a+b)^*b) \subseteq \rho(((a+b)^*b)^*)$.

b) Zij $L = \{a,b\}^*\{b\}$. Dan geldt $L^k \subseteq L$ voor elke $k \geqslant 1$. Dit volgt met inductie naar k:

$k = 1$: $L \subseteq L$.

Stel $L^k \subseteq L$ voor alle k, $1 \leqslant k \leqslant n$ (Inductiehypothese)

$k = n+1$: $L^{n+1} = L^nL \subseteq LL \subseteq L$. Omdat $\rho((a+b)^*b) = \rho((a+b)^*)\rho(b) = \{a,b\}^*\{b\}$ geldt

$$\rho(((a+b)^*b)^*) = \bigcup_{k\geqslant 0} (\rho((a+b)^*b))^k = \{\lambda\} \cup \bigcup_{k\geqslant 1} (\rho((a+b)^*b))^k$$

$$\subseteq \{\lambda\} \cup \rho((a+b)^*b)$$

Uit a en b samen volgt $\rho(\Lambda + (a+b)^*b) = \rho(((a+b)^*b)^*)$ zodat $\Lambda + (a+b)^*b \equiv ((a+b)^*b)^*$.

2.1. Grammatica is rechtslineair, kan worden omgevormd tot een equivalente reguliere grammatica en vervolgens kan een equivalente eindige automaat worden geconstrueerd (resulteert in 7 toestanden). Hier wordt L(G) bepaald en dan een automaat voor L(G) gemaakt. Schets van een afleiding volgens G

$$S \overset{*}{\Rightarrow} a^{2n}S \Rightarrow a^{2n}abA \Rightarrow a^{2n+1}bB \overset{*}{\Rightarrow} a^{2n+1}ba^{2m} \text{ of}$$

$$S \overset{*}{\Rightarrow} a^{2n}S \Rightarrow a^{2n}aabA \Rightarrow a^{2n+2}bB \overset{*}{\Rightarrow} a^{2n+2}ba^{2m}$$

Hieruit zien we dat $L(G) = \{a^{n+1}ba^{2m} \mid n,m \geqslant 0\}$.

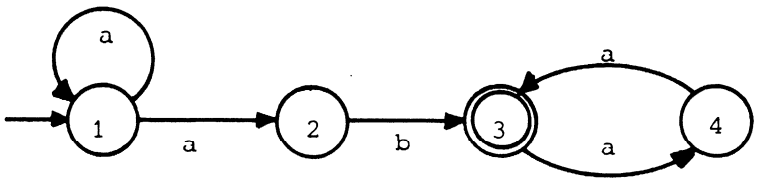
De volgende uitspraak U is een invariant voor G: $U = [\phi \in \{a^{2n} \mid n \geqslant 0\}\{S, abA, aabA, abB, aabB, ab, aab\}\{a^{2m} \mid m \geqslant 0\}]$. Als $w \in L(G)$ dan $S \overset{*}{\Rightarrow} w$. Daar U(S), geldt ook U(w) dus $w \in \{a^{2n} \mid n \geqslant 0\}\{ab, aab\}\{a^{2m} \mid m \geqslant 0\} = \{a^{n+1}ba^{2m} \mid n,m \geqslant 0\}$.

Verder geldt voor elke $n \geq 0$: $S \overset{*}{\Rightarrow} a^{2n}S$ en $B \overset{*}{\Rightarrow} a^{2n}B$ (bewijs met inductie naar n). Als $w \in \{a^{n+1}ba^{2m} \mid n \geq 0, m \geq 0\}$, zeg $w = a^{n+1}ba^{2m}$ dan

- als n+1 even is, zeg n+1=2k dan $S \overset{*}{\Rightarrow} a^{2(k-1)}S \Rightarrow a^{2k}bA \Rightarrow a^{n+1}bB \overset{*}{\Rightarrow} a^{n+1}ba^{2m}$

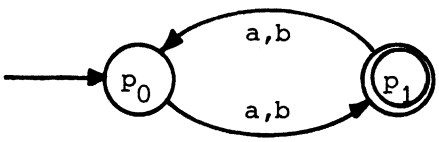
- als n+1 oneven is, zeg n+1=2k+1 dan $S \overset{*}{\Rightarrow} a^{2k}S \Rightarrow a^{2k+1}bA \Rightarrow a^{n+1}bB \overset{*}{\Rightarrow} a^{n+1}ba^{2m}$

Dus $L(G) \subseteq \{a^{n+1}ba^{2m} \mid n,m \geq 0\} \subseteq L(G)$. Deze taal wordt geaccepteerd door de automaat



Immers:
- $(1,w) \overset{*}{\vdash} (2,\lambda)$ als en alleen als $w = a^{n+1}$, voor zekere $n \geq 0$.
- $(3,w) \overset{*}{\vdash} (3,\lambda)$ als en alleen als $w = a^{2n}$, voor zekere $n \geq 0$.

2.2. De gegeven automaat A_2 heeft als transitiediagram

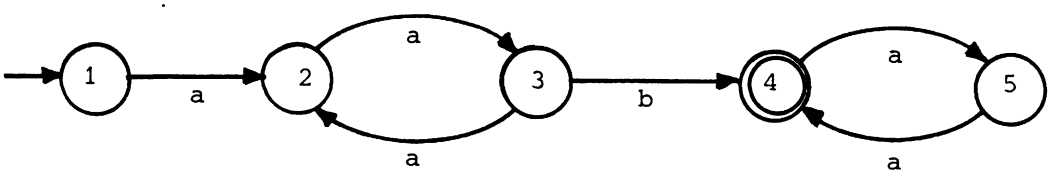


$L(A_2) = \{w \in \{a,b\}^* \mid |w| \text{ is oneven}\}$ want

$(p_x,w) \overset{*}{\vdash} (p_y,\lambda)$ als en alleen als $x + |w| \equiv y \pmod 2$.

$$\begin{aligned} \text{Dus } L(A_2) \cap L(G) &= \{a^{n+1}ba^{2m} \mid m \geq 0,\ n \geq 0,\ n+1+1+2m \text{ oneven}\} \\ &= \{a^{n+1}ba^{2m} \mid m \geq 0,\ n \geq 0,\ n \text{ oneven}\} \\ &= \{a^{2n}ba^{2m} \mid n \geq 1,\ m \geq 0\} \end{aligned}$$

Deze taal wordt geaccepteerd door de automaat met als transitiediagram



want
$(3,w) \vdash^{*} (4,\lambda)$ als en alleen als $w = ba^{2m}$ voor zekere $m \geq 0$ en
$(1,w) \vdash^{*} (3,\lambda)$ als en alleen als $w = a^{2m}$ voor zekere $m \geq 1$.

3. $G = (\{S,A,C,a,b,c\}, \{a,b,c\}, P, S)$ met als verzameling productieregels:
$P = \{S \to AC, A \to aAb, A \to \lambda, C \to bCc, C \to \lambda\}$.
De uitspraak $U(\phi) = [\phi \in \{a\}^*\{A,\lambda\}\{b\}^*\{C,\lambda\}\{C\}^*$ en $\#(a,\phi) + \#(c,\phi) = \#(b,\phi)]$ is een invariant voor het aan de grammatica ten grondslag liggende herschijfsysteem. Stel namelijk dat $U(\phi)$ geldt, en dat $\phi \Longrightarrow \psi$ door middel van de regel

- $A \to aAb$ Uit $U(\phi)$ volgt dan dat voor zekere $n,m,k \geq 0$, $\phi = a^nAb^mCc^k$ of $\phi = a^nAb^mc^k$ en $m = n+k$. Als $\phi \Longrightarrow \psi$ door middel van $A \to aAb$ volgt dus $\psi = a^{n+1}Ab^{m+1}Cc^k$ of $\psi = a^{n+1}Ab^{m+1}c^k$ met $m+1 = n+1+k$ zodat $U(\psi)$.

- $A \to \lambda$ Dan kan ϕ op dezelfde wijze worden geschreven en volgt $\psi = a^nb^mCc^k$ of $\psi = a^nb^mc^k$ met $m = n+k$. Derhalve $U(\psi)$.

- Controle op herschrijven door middel van $C \to bCc$ of $C \to \lambda$ gaat net zo.

Aangezien elke afleiding van G de vorm $S \Longrightarrow AC \overset{*}{\Longrightarrow} w$ heeft en U(AC) geldt, geldt ook U(w). Daar $w \in \{a,b,c\}^*$ is, volgt: als $w \in L(G)$ dan $w \in \{a^nb^mc^k \mid n+k = m,\ n,k,m \geq 0\}$.

Voor iedere $n \geq 0$ geldt: $A \overset{*}{\Longrightarrow} a^nAb^n$.
$n = 0$: $A \overset{*}{\Longrightarrow} A$, definitie van $\overset{*}{\Longrightarrow}$.
Stel $A \overset{*}{\Longrightarrow} a^nAb^n$ voor elke n, $0 \leq n \leq m$ (inductiehypothese)
$n = m+1$: $A \Longrightarrow aAb \overset{*}{\underset{(IH)}{\Longrightarrow}} aa^nAb^nb = a^{n+1}Ab^{n+1}$.
Evenzo volgt: voor iedere $n \geq 0$ geldt $C \overset{*}{\Longrightarrow} b^nCc^n$.

Zij nu $w \in \{a^nb^mc^k \mid n,k,m \geq 0,\ m = n+k\}$. Onderscheid vier gevallen:

$w = \lambda$ $(n=m=k=0)$ $S \Rightarrow AC \Rightarrow C \Rightarrow \lambda$ dus dan $w \in L(G)$

$w = a^n b^n$ $(k=0)$ $S \Rightarrow AC \overset{*}{\Rightarrow} a^n A b^n C \Rightarrow a^n b^n C \Rightarrow a^n b^n$, dus $w \in L(G)$.

$w = b^k c^k$ $(n=0)$ $S \Rightarrow AC \overset{*}{\Rightarrow} Ab^k Cc^k \Rightarrow b^k Cc^k \Rightarrow b^k c^k$, dus $w \in L(G)$.

$w = a^n b^{n+k} c^k$ $S \Rightarrow AC \overset{*}{\Rightarrow} a^n A b^n C \overset{*}{\Rightarrow} a^n A b^n b^k Cc^k \Rightarrow a^n b^{n+k} Cc^k \Rightarrow a^n b^{n+k} c^k$, dus $w \in L(G)$.

Dus $L(G) = \{a^n b^{n+k} c^k \mid n,k > 0\}$.

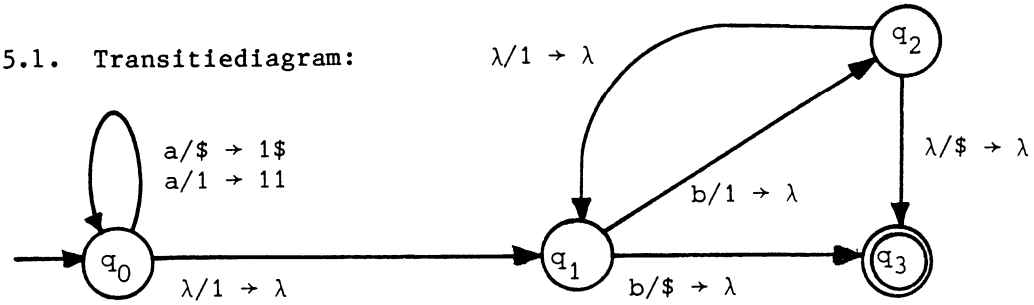
4. Zij $L_1 = \{a^n b^{n^2} \mid n > 1\}$. Deze verzameling is niet bestand tegen pompen volgens de pompstelling voor $\mathcal{L}_2$ (eigenschap 3.33.). Stel dat dit wel het geval is, en dat het getal p voldoet. Dan moet aPb^{P^2} geschreven kunnen worden als $aPb^{P^2} = xuvyz$ zodat $|uy| > 1$, $|uvy| < p$ en $xu^i vy^i z \in L$ voor $i > 0$, immers $|aPb^{P^2}| = p^2+p>p$ (want $p > 1$). u en y kunnen elk ten hoogste een soort letter bevatten omdat anders in xu^2vy^2z de alfabetische volgorde is verstoord. Bovendien kunnen u en y niet dezelfde soort letter bevatten, anders geldt $(\#(a,xvz))^2 \neq \#(b,xvz)$. Derhalve bestaat u alleen uit a's en y alleen uit b's en geldt $|u| > 1$, $|y| > 1$. $\#(a,xu^2vy^2z) = p + |u| > p+1$, zodat $\#(b,xu^2vy^2z)$ groter dan of gelijk aan $(p+1)^2$ moet zijn. Maar $\#(b,xu^2vy^2z) = p^2 + |y| < p^2 + p < p^2 + 2p + 1 = (p+1)^2$. Tegenspraak.

Omdat L_1 niet bestand is tegen pompen volgens 3.33, is $L_1 \notin \mathcal{L}_2$.

Als $L \in \mathcal{L}_2$, dan is ook $L_1 = L \cap \{a^n b^m \mid n,m > 1\}$ contextvrij, immers $\mathcal{L}_2$ is gesloten onder doorsnijding met reguliere talen (7.14) en $\{a^n b^m \mid n,m > 1\} \in \mathcal{L}_3$.

Dus $L \notin \mathcal{L}_2$ (L is niet bestand tegen pompen; dit aantonen is echter bewerkelijker).

5.1. Transitiediagram:



5.2./

5.3. De enige manier om de stapel te ledigen, is met behulp van een transitie $x/\$ \to \lambda$. Deze transities voeren allemaal tot een accepterende toestand, dus $L_\Lambda(A) \subseteq L_F(A)$. Omdat iedere transitie naar een accepterende toestand tevens de stapel ledigt, geldt ook $L_F(A) \subseteq L_\Lambda(A)$. Samen leidt dit tot $L_\Lambda(A) = L_F(A)$.

Alle accepterende berekeningen beginnen in q_0, eindigen in q_3 en gaan via toestand q_1.

$(q_0,w,\$) \vdash^* (q_0,\lambda,\alpha)$ als en alleen als $w = a^n$ en $\alpha = 1^n\$$ voor zekere $n \geq 0$

$(q_1,w,1^n\$) \vdash^* (q_1,\lambda,\$)$ als en alleen als $n = 2k$ en $w = b^k$ voor zekere $k \geq 0$

$(q_1,w,1^n\$) \vdash^* (q_2,\lambda,\$)$ als en alleen als $n = 2k+1$ en $w = b^{k+1}$ voor zekere $k \geq 0$.

Dus $w \in L_F(A)$ als en alleen als er symboolrijen $u = a^n$ en $v = b^m$ zijn, zodat

$$- (q_0,w,\$) = (q_0,a^nb^m,\$) \vdash^* (q_0,b^m,1^n\$) \vdash (q_1,b^m,1^{n-1}\$) \vdash^* (q_1,b,\$) \vdash (q_3,\lambda,\lambda)$$

Dan moet dus $n \geq 1$ en $n-1 = 2k$ en $m = k+1$ voor zekere $k \geq 0$. Of

$$- (q_0,w,\$) = (q_0,a^nb^m,S) \vdash^* (q_0,b^m,1^n\$) \vdash (q_1,b^m,1^{n-1}\$) \vdash^* (q_2,\lambda,\$) \vdash (q_3,\lambda,\lambda)$$

Dan moet dus $n \geq 1$ en $n-1 = 2k+1$ en $m = k+1$ voor zekere $k \geq 0$.
Met andere woorden:

$w \in L_F(A)$ als en alleen als $w \in \{a^nb^m \mid n \geq 1,\ m = \lfloor \frac{n-1}{2} \rfloor + 1\}$.

Samengevat:

$L_F(A) = L_\Lambda(A) = \{a^n b^m \mid n \geq 1, m = \lfloor \frac{n-1}{2} \rfloor + 1\}$.

5.4. $\sigma(q_0,a,1) \neq \emptyset$ en $\sigma(q_0,\lambda,1) \neq \emptyset$ dus A is niet deterministisch.

5.5. $L(G) = \{a^n b^m \mid n \geq 2, \lfloor \frac{n-1}{2} \rfloor + 1 = m, n+m \text{ is drievoud}\}$
Nu geldt:

indien n even is, $n = 2k$, dan $n + \lfloor \frac{n-1}{2} \rfloor + 1 = 2k + k-1 + 1 = 3k$

indien n oneven is, $n = 2k+1$, dan $n + \lfloor \frac{n-1}{2} \rfloor + 1 = 2k + 1 + k+1 = 3k + 2$

Dus $L(G) = \{a^n b^m \mid n=2k \text{ voor zekere } k\geq 1, m = \lfloor \frac{n-1}{2} \rfloor + 1\}$
$= \{a^{2k} b^k \mid k \geq 1\}$.

$G = (\{S,a,b\}, \{a,b\}, \{S \to aaSb, S \to aab\}, S)$

invariant: $U(\phi) = [\phi \in \{a^{2k} x b^k \mid k \geq 1, x \in \{S,\lambda\}\}]$.
Dus als $w \in L(G)$ dan:
- $w = aab$ en dan $w \in \{a^{2k} b^k \mid k \geq 1\}$
- $S \Longrightarrow aaSb \overset{*}{\Longrightarrow} w$. Omdat U(aaSb) geldt, geldt eveneens U(w) zodat $w \in \{a^{2k} b^k \mid k \geq 1\}$.

Derhalve $L(G) \subseteq \{a^{2k} b^k \mid k \geq 1\}$.

Voor elke $k \geq 0$ geldt $S \overset{*}{\Longrightarrow} a^{2k} S b^k$ (bewijs met inductie naar k).
Dus als $w \in \{a^{2k} b^k \mid k \geq 1\}$ dan is er een afleiding $S \overset{*}{\Longrightarrow} a^{2(k-1)} S b^{k-1} \Longrightarrow a^{2k} b^k$, dus $w \in L(G)$.
Derhalve $\{a^{2k} b^k \mid k \geq 1\} \subseteq L(G)$.

6.1. $E_{fg} = \{x \in \Sigma^+ \mid f(x) = g(x)\} = \{x \in \Sigma^+ \mid g(x) = f(x)\} = E_{gf}$.

$E^+_{fg} = \bigcup_{k \geq 1} E^k_{fg}$, dus $E_{fg} \subseteq E^+_{fg}$.

Zij $x \in E^+_{fg}$ dan is er een $k \geq 1$ zodat $x = u_1u_2\ldots u_k$ met $u_i \in E_{fg}$ voor alle i, $1 \leq i \leq k$. Dus $f(u_i) = g(u_i)$ voor alle i, $1 \leq i \leq k$. Omdat f en g homomorfismen zijn, geldt $f(x) = f(u_1)f(u_2)\ldots f(u_k) = g(u_1)g(u_2)\ldots g(u_k) = g(x)$, zodat $x \in E_{fg}$ en derhalve $E^+_{fg} \subseteq E_{fg}$.

Samengenomen: $E_{fg} = E^+_{fg}$.

6.2. Zij $(n,a_1,\ldots,a_n,b_1,\ldots,b_n)$ met $a_i,b_i \in \Delta^+$ voor alle i, $1 \leq i \leq n$ een exemplaar van PCP en zij $\Sigma = \{s_1,s_2,\ldots,s_n\}$ een alfabet van n letters.

Zij $f,g: \Sigma^* \to \Delta^*$ homomorfismen, bepaald als volgt: $f(s_i) = a_i$ en $g(s_i) = b_i$, $1 \leq i \leq n$.

Dan $E_{fg} = \{w \in \Sigma^+ \mid f(w) = g(w)\}$. $E_{fg} \neq \emptyset$ als en alleen als $\exists w \in \Sigma^+[f(w) = g(w)]$ als en alleen als er een rij $s_{i_1}s_{i_2}\ldots s_{i_k}$ bestaat zodat $f(s_{i_1}s_{i_2}\ldots s_{i_k}) = a_{i_1}a_{i_2}\ldots a_{i_k} = b_{i_1}b_{i_2}\ldots b_{i_k} = g(s_{i_1}s_{i_2}\ldots s_{i_k})$.
Met andere woorden, $E_{fg} \neq \emptyset$ als en alleen als het exemplaar van PCP "waar" oplevert. Als dus het probleem U algoritmisch oplosbaar is, is ook PCP algoritmisch oplosbaar. Derhalve is U niet algoritmisch oplosbaar.

6.3. $E_{fg} \neq \emptyset$ als en alleen als er een woord s^k in Σ^+ is, zodat $f(s^k) = (f(s))^k = g(s^k) = (g(s))^k$. Dit is het geval als en alleen als $f(s) = g(s)$. De vraag of $f(s) = g(s)$ kan daadwerkelijk worden beslist, zodat $[f,g \in HOM(\Sigma,\Delta): E_{fg} = \emptyset]$ algoritmisch oplosbaar is ingeval $card(\Sigma) = 1$.

6.4. Een algoritme, dat de karakteristieke functie van E_{fg} berekent, is:

invoer : $x = s_1 s_2 \ldots s_k \in \Sigma^+$

uitvoer: $u \in \{ja, nee\}$

methode: $y := f(s_1)f(s_2)\ldots f(s_k)$;

$z := g(s_1)g(s_2)\ldots g(s_k)$;

als $y = z$ dan $u :=$ ja anders $u :=$ nee.

Omdat f en g eindig gerepresenteerd kunnen worden, bijvoorbeeld als tabellen, is bovenstaande procedure daadwerkelijk uitvoerbaar en dus is E_{fg} beslisbaar.

TECHNISCHE HOGESCHOOL DELFT
Onderafdeling der Wiskunde & Informatica
Julianalaan 132
2628 BL DELFT

TENTAMEN THEORETISCHE INFORMATICA (a137)
Maandag 20 augustus 1984
14.00 - 17.00 uur

In het volgende wordt met "bewijs ..." aangegeven dat een (correct en volledig) bewijs moet worden gegeven. Met "beargumenteer ..." wordt bedoeld dat men een steekhoudende redenering dient te geven, waarin kan worden volstaan met het formuleren van beweringen en inductiehypothesen, terwijl het inductiebewijs niet behoeft te worden uitgeschreven; evenzo kan men volstaan met het formuleren van invarianten (of invariante beweringen) en behoeft men de verificatie, dat deze beweringen inderdaad voor een gegeven herschrijfsysteem invariant zijn, niet in detail uit te werken.

1. L_1 en L_2 zijn talen over het alfabet Σ en h: $\Delta^* \to \Sigma^*$ is een homomorfisme. Onderzoek of de volgende inclusies gelden. Geef steeds een bewijs of een tegenvoorbeeld.

(1.1) $(L_1 \cup L_2)^* \subseteq L_1^* \cup L_2^*$

(1.2) $L_1^* \cup L_2^* \subseteq (L_1 \cup L_2)^*$

(1.3) $h^{-1}(L_1 \cup L_2) \subseteq h^{-1}(L_1) \cup h^{-1}(L_2)$

(1.4) $h^{-1}(L_1) \cup h^{-1}(L_2) \subseteq h^{-1}(L_1 \cup L_2)$

2. De grammatica $G = (\{S,X,a,b\},\{a,b\},P,S)$ heeft de volgende productieregels.

$P = \{S \to abSba,\ S \to X,\ S \to \lambda,\ X \to aXa,\ X \to bXb,\ X \to aXb,\ X \to bXa,\ X \to S\}$

(2.1) Bepaal L(G) en beargumenteer Uw antwoord.

(2.2) Construeer een deterministische eindige automaat A zodat L(A) = L(G). Beargumenteer de correctheid van A.

(2.3) Zij h: $\{a,b\}^* \to \{a\}^*$ het homomorfisme bepaald door h(a)=h(b)=a. Construeer een deterministische eindige automaat B zodat L(B) = h(L(G)). Beargumenteer de correctheid van automaat B.

3. Ontwerp een herschrijfsysteem (V,P) met $\{B,E,a\} \subseteq V$ waarvoor geldt $Ba^nE \overset{*}{\Longrightarrow} a^m$ dan en slechts dan als $m = \lfloor \frac{n}{3} \rfloor$. Bewijs de goede werking van het herschrijfsysteem.

4. Zij $L = \{w \in \{a,b,c\}^* \mid \#(a,w) + \#(b,w) = \#(c,w)\}$

(4.1) Construeer een stapelautomaat A zodat $L_F(A) = L_\Lambda(A) = L$. Beargumenteer de correctheid van uw ontwerp.

(4.2) Construeer een contextvrije grammatica G zodat $L(G) = L \cap \rho(a^*b^*c^*)$. Beargumenteer de correctheid van Uw grammatica.

5. Zij $L = \{b^n a^{m^2} \mid n,m \geq 1\} \cup \{a^k \mid k \geq 1\}$.

(5.1) Laat zien dat L voldoet aan de pompstelling voor reguliere talen.

(5.2) Bewijs dat L niet regulier is.

6. Bewijs dat het probleem $[G_1,G_2 \in \mathcal{G}_2 : L(G_1) \subseteq L(G_2)]$ algoritmisch onoplosbaar is. U mag er daarbij van uitgaan dat de taal $\{wcv\$ \mid w,v \in V^*, w \neq v^s\}$ contextvrij is voor elk alfabet V, zonder dat U deze uitspraak behoeft te bewijzen of te beargumenteren.

20.08.1984

1.1. Onjuist

Neem $L_1 = \{a\}$, $L_2 = \{b\}$ dan $(L_1 \cup L_2)^* = \{a,b\}^* \not\subseteq \{a\}^* \cup \{b\}^* = L_1^* \cup L_2^*$; ab, bijvoorbeeld, is in $\{a,b\}^* \setminus (\{a\}^* \cup \{b\}^*)$.

1.2. Juist

Als $L \subseteq M$ dan is $L^* \subseteq M^*$. Immers, als $w \in L^*$ en $w \neq \lambda$ dan is w te schrijven als $w = v_1 v_2 \ldots v_k$ voor zekere $k \geq 1$ en $v_i \in L$ voor alle i, $1 \leq i \leq k$. Dus ook $v_i \in M$ voor alle i, $1 \leq i \leq k$. Dus $w \in M^*$. $L_1 \subseteq L_1 \cup L_2$ en $L_2 \subseteq L_1 \cup L_2$. Dus $L_1^* \subseteq (L_1 \cup L_2)^*$ en $L_2^* \subseteq (L_1 \cup L_2)^*$. Dus ook $L_1^* \cup L_2^* \subseteq (L_1 \cup L_2)^*$.

1.3. Juist

Als $w \in h^{-1}(L_1 \cup L_2)$, dan $h(w) \in L_1 \cup L_2$. Dus $h(w) \in L_1$ of $h(w) \in L_2$. Dus $w \in h^{-1}(L_1)$ of $w \in h^{-1}(L_2)$. Dus $w \in h^{-1}(L_1) \cup h^{-1}(L_2)$. Met andere woorden: $h^{-1}(L_1 \cup L_2) \subseteq h^{-1}(L_1) \cup h^{-1}(L_2)$.

1.4. Juist

Als $w \in h^{-1}(L_1) \cup h^{-1}(L_2)$, dan $w \in h^{-1}(L_1)$ of $w \in h^{-1}(L_2)$. Dan $h(w) \in L_1$ of $h(w) \in L_2$. Dus $h(w) \in L_1 \cup L_2$ zodat $w \in h^{-1}(L_1 \cup L_2)$. Met andere woorden: $h^{-1}(L_1) \cup h^{-1}(L_2) \subseteq h^{-1}(L_1 \cup L_2)$.

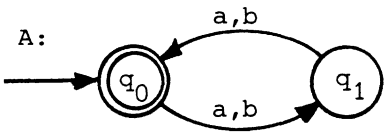
2.1. De regel $S \to abSba$ is overbodig en de symbolen S en X kunnen vereenzelvigd worden. $L(G) = \{w \in \{a,b\}^* \mid |w| \text{ is even}\}$.

invariant: $U(\phi) = [\phi \in \{uyv \mid u,v \in \{a,b\}^*, |u| = |v|, y \in \{S,X,\lambda\}\}]$.
Als $w \in L(G)$ dan $S \overset{*}{\Rightarrow} w$. Daar U(S) geldt, geldt ook U(w). Dus $|w|$ is even en $w \in \{w \in \{a,b\}^* \mid |w| \text{ is even}\}$.
Voor elke $k \geq 0$ en elk tweetal $u,v \in \{a,b\}^*$ met $|u| = |v| = k$ geldt $S \Rightarrow X \overset{k}{\Rightarrow} uXv$ (Met inductie naar k en splitsing naar de vier mogelijkheden voor de combinatie "eerste-letter-u, laatste-letter-v"). Als $w \in \{w \in \{a,b\}^* \mid |w| \text{ is even}\}$ dan zijn er $u,v \in \{a,b\}^*$ met $|u| = |v|$ en $w = uv$. Dan $S \Rightarrow X \overset{*}{\Rightarrow} uXv \Rightarrow uv = w$ zodat $w \in L(G)$.

Samen: $L(G) = \{w \in \{a,b\}^* \mid |w| \text{ is even}\}$.

De automaat met onderstaand transitiediagram voldoet

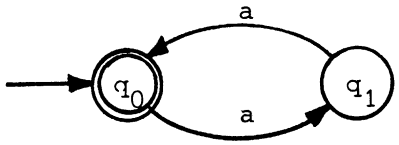
A:



$(q_x,v) \vdash^* (q_y,\lambda)$ als en alleen als $y \equiv x + |v| \pmod 2$. (Bewijs met inductie naar de lengte van v). $L(A) = \{w \mid (q_0,w) \vdash^* (q_0,\lambda)\} = \{w \in \{a,b\}^* \mid |w| \text{ is even}\}$.

2.3. $h(L(G)) = \{a^{2n} \mid n > 0\}$.

B:



$(q_x,a^n) \vdash^* (q_y,\lambda)$ als en alleen als $y \equiv x + n \pmod 2$.

3. Zij het herschrijfsysteem (V,P) bepaald door $V=\{B,a,E\}$ en $P = \{Ba^3 \to aB, BE \to \lambda, BaE \to \lambda, Ba^2E \to \lambda\}$.
Voor elke n is de uitspraak U_n een invariant

$$U_n(\phi) \overset{\text{def}}{=} [\phi = a^{\lfloor \frac{n}{3} \rfloor} \text{ of } k,m > 0\ [\phi = a^k B a^m E \text{ en } 3k + m = n]].$$

Stel dat $U_n(\phi)$ geldt en dat $\phi \Rightarrow \psi$ via de regel:

- $Ba^3 \to aB$: dan zijn er getallen k en m zodat $\phi = a^k B a^m E$ en $3k + m = n$ en $m \geq 3$. Voor ψ geldt: $\psi = a^{k+1} B a^{m-3} E$ en $3(k+1) + m - 3 = 3k + m = n$. Dus $U_n(\psi)$.

- $BaE \to \lambda$: dan zijn er getallen k en m zodat $\phi = a^k B a^m E$ en $3k + m = n$ en $m = 1$. Dus $k = \lfloor \frac{n}{3} \rfloor$. Voor ψ geldt dus $\psi = a^{\lfloor \frac{n}{3} \rfloor}$.

- $BE \to \lambda$ en $Ba^2E \to \lambda$ gaan net zo als $BaE \to \lambda$.

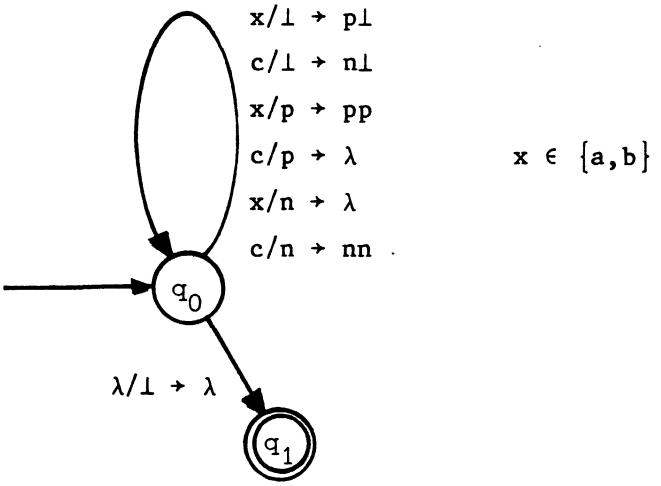
Als dus voor zekere $n \geq 0$ geldt $Ba^nE \overset{*}{\Rightarrow} a^m$ dan is $m = \lfloor \frac{n}{3} \rfloor$, immers $U_n(Ba^nE)$ geldt en dus ook $U_n(a^m)$.

Voor elke $n \geq 0$ geldt: $Ba^{3n} \overset{*}{\Rightarrow} a^nB$. Bewijs met inductie naar w:
$n = 0$: $B \overset{*}{\Rightarrow} B$ per definitie.
Stel dat voor elke n, $0 \leq n \leq m$ geldt $Ba^{3n} \overset{*}{\Rightarrow} a^nB$ (Inductiehypothese)
$n = m + 1$: $Ba^{3(m+1)} = Ba^{3m}aaa \overset{*}{\underset{IH}{\Rightarrow}} a^mBa^3 \Rightarrow a^{m+1}B$.

Voor iedere $n \geq 0$ is er een $m \geq 0$ en een r, $0 \leq r < 3$ zodat $n = 3m + r$.
Dus geldt voor elke $n \geq 0$: $Ba^nE = Ba^{3m+r}E \overset{*}{\Rightarrow} a^mBa^rE \Rightarrow a^m = a^{\lfloor \frac{n}{3} \rfloor}$. Zodat voor elke $n \geq 0$, Ba^nE herschreven kan worden tot $a^{\lfloor \frac{n}{3} \rfloor}$.

4.1. Op de stapel wordt bijgehouden het verschil $\Delta(w) = \#(a,w) + \#(b,w) - \#(c,w)$. Als het verschil positief is, staan er p's op de stapel, als het negatief is, n's. Als $\Delta(w) = 0$ bevat de stapel alleen het beginstapelsymbool $\perp$. Tenslotte wordt door middel van een λ-transitie de stapel leeg gemaakt.

Definieer A in overeenstemming met het transitiediagram.



Elke accepterende berekening van A heeft de vorm $(q_0,w,\perp) \overset{*}{\vdash} (q_0,\lambda,\perp) \vdash (q_1,\lambda,\lambda)$, zodat $L_\Lambda(A) = L_F(A)$. Tevens geldt

$(q_0,w,\perp) \overset{*}{\vdash} (q_0,\lambda,\alpha\perp)$ als en alleen als

indien $\Delta(w) = 0$ dan $\alpha = \lambda$

indien $\Delta(w) > 0$ dan $\alpha = p^{\Delta(w)}$

indien $\Delta(w) < 0$ dan $\alpha = n^{\Delta(w)}$

Dus $L_\Lambda(A) = L_F(A) = \{w \in \{a,b,c\}^* \mid \Delta(w) = 0\}$.

4.2. $L \cap \rho(a^*b^*c^*) = \{a^nb^mc^{n+m} \mid n,m \geq 0\}$. Een contextvrije grammatica die deze taal voortbrengt, is

$G = (\{S,T,a,b,c\},\{a,b,c\},P,S)$ met $P = \{S \to aSc,\ S \to T,\ T \to bTc,\ T \to \lambda\}$.
Voor elke $n \geq 0$ geldt: $S \overset{*}{\Rightarrow} a^nSc^n$ en $T \overset{*}{\Rightarrow} b^nTc^n$. Dus, voor elke $n,m \geq 0$: $S \overset{*}{\Rightarrow} a^nSc^n \Rightarrow a^nTc^n \overset{*}{\Rightarrow} a^nb^mTc^mc^n \Rightarrow a^nb^mc^{n+m}$, zodat

$L \cap \rho(a^*b^*c^*) \subseteq L(G)$.
Verder geldt de volgende invariant:

$$U(\phi) = [\phi \in \{a^nSb^n \mid n \geq 0\} \cup \{a^nb^mTc^{n+m} \mid n,m \geq 0\} \cup \{a^nb^mc^{m+n} \mid n,m \geq 0\}]$$

Zij $w \in L(G)$ dan $S \overset{*}{\Rightarrow} w$. Omdat $U(S)$ geldt, geldt ook $U(w)$. Daar $w \in \{a,b,c\}^*$ volgt $w \in \{a^nb^mc^{n+m} \mid n,m \geq 0\} = L \cap \rho(a^*b^*c^*)$.
Samengevat: $L(G) = L \cap \rho(a^*b^*c^*)$.

5.1. Kies p gelijk aan 1. Zij $w \in L$, $|w| > 1$ en zij c de eerste letter van w: $w = cv$. Dan kan men schrijven $w = xyz$ zodat voor elke $i \geq 0$ $xy^iz \in L$ door de keuze $x = \lambda$, $y = c$ en $z = v$. Dit is onmiddellijk duidelijk voor $i \geq 1$. Wat het geval $i = 0$ betreft:

als $w = ba^{r^2}$ dan $xy^0z = a^{r^2} \in \{a^k \mid k \geq 1\} \subseteq L$
als $w = bb^ta^{r^2}$ dan $xy^0z = ba^{r^2} \in \{b^na^{m^2} \mid n,m \geq 1\} \subseteq L$
als $w = aa^t$ dan $xy^0z = a^t \in \{a^k \mid k \geq 1\} \subseteq L$

5.2. De klasse $\mathcal{L}_3$ is gesloten met betrekking tot doorsnijden. Als dus $L \in \mathcal{L}_3$ dan ook $L \cap \rho(ba^*) = \{ba^{m^2} \mid m \geq 1\} \in \mathcal{L}_3$. Maar $\{ba^{m^2} \mid m \geq 1\}$ is niet bestand tegen pompen volgens de pompstelling voor reguliere talen. Stel immers dat dit wel het geval is, en dat het getal p voldoet. Beschouw $ba^{(p+1)^2} \in \{ba^{m^2} \mid m \geq 1\}$. Daar $|ba^{(p+1)^2}| > p$ moet men x, y en z kunnen kiezen zodat $xyz = ba^{(p+1)^2}$ en voor elke $i \geq 0$ $[xy^iz \in \{ba^{m^2} \mid m \geq 1\}]$ en $0 < |y| \leq p$. Dan is $\#(b,y) = 0$. Welnu $xy^0z = ba^{(p+1)^2-|y|}$.

$(p+1)^2 > (p+1)^2 - |y| \geq (p+1)^2-p = p^2 + p + 1 > p^2$, zodat $xy^0z \notin \{ba^{m^2} \mid m \geq 1\}$.

Derhalve $\{ba^{m^2} \mid m \geq 1\} \notin \mathcal{L}_3$ en dus ook $L \notin \mathcal{L}_3$.

6. Zij $(n,a_1,\ldots,a_n,b_1,\ldots,b_n)$ een exemplaar van PCP over het alfabet V. De grammatica $G_p = (\{B,S\} \cup V, V, P, B)$ zij gedefinieerd door

$$P = \{B \to S\$\} \cup \{S \to a_iSb_i^s \mid 1 \leq i \leq n\} \cup \{S \to a_icb_i^s \mid 1 \leq i \leq n\}.$$

Dan geldt

$$L(G_p) = \{a_{i_1}a_{i_2}\ldots a_{i_k}c(b_{i_1}b_{i_2}\ldots b_{i_k})^s\$ \mid k \geq 1,\ 1 \leq i_j \leq n \text{ voor alle } j,$$

$$1 \leq j \leq k\}.$$

Indien het exemplaar van PCP oplevert de uitspraak:

- waar : dan is er rij $i_1,\ldots,i_k$ zodat $a_{i_1}\ldots a_{i_k} = b_{i_1}\ldots b_{i_k}$ dus dan $L(G_p) \setminus \{wcv\$ \mid w \neq v^s\} \neq \emptyset$ en $L(G_p) \not\subseteq \{wcv\$ \mid w,v \in V^*,\ w \neq v^s\}$.
- onwaar: dan geldt voor elke rij $i_1\ldots i_k$ dat $a_{i_1}\ldots a_{i_k} \neq b_{i_1}\ldots b_{i_k}$ zodat $L(G_p) \subseteq \{wcv\$ \mid w,v \in V^*\ w \neq v^s\}$.

Zij G_0 een type 2 grammatica met $L(G_0) = \{wcv\$ \mid w,v \in V^*,\ w \neq v^s\}$. Krachtens gegeven bestaat er zo'n G_0.

Indien $[G_1,G_2 \in \mathcal{G}_2: L(G_1) \subseteq L(G_2)]$ algoritmisch oplosbaar is, dan is ook PCP algoritmisch oplosbaar. En wel door gegeven een exemplaar van PCP de bovenstaande grammatica G_p te construeren en vervolgens te bepalen of $L(G_p) \subseteq L(G_0)$. Derhalve is $[G_1,G_2 \in \mathcal{G}_2: L(G_1) \subseteq L(G_2)]$ niet algoritmisch oplosbaar.

12. REGISTER

Verwijzingen vinden plaats naar definitienummer (bijvoorbeeld: 1.23) of naar paginanummer (bijvoorbeeld: p. 123).