3 De stelling van Kleene

Definitie 3.1 Een formele taal heet *regulier* als hij wordt herkend door een deterministische eindige automaat. Talen van de vorm $L(r)$ met r een reguliere expressie noemen we *representeerbaar*.

Uit de resultaten van de vorige week volgt dus dat een taal L regulier is precies als er een NEA $\mathbb{A}$ is zdd $L = L(\mathbb{A})$. De volgende stelling, die aan de basis ligt van de gehele theorie van eindige automaten en reguliere expressies, zegt dat de klasse van reguliere talen precies overeenkomt met die van de representeerbare talen.

Stelling 3.2 (Kleene) *Een formele taal is regulier dan en slechts dan als hij representeerbaar is.*

In dit hoofdstuk zullen we laten zien dat we in feite uit een willekeurige eindige automaat $\mathbb{A}$ een reguliere expressie $r_{\mathbb{A}}$ kunnen *'destilleren'* waarvoor geldt $L(r_{\mathbb{A}}) = L(\mathbb{A})$, terwijl we ook omgekeerd, van een reguliere expressie r, een automaat $\mathbb{A}_r$ kunnen *construeren* waarvoor geldt dat $L(\mathbb{A}_r) = L(r)$.

3.1 Reguliere expressies uit eindige automaten

We beginnen met de moeilijkste transformatie: die van eindige automaten tot reguliere expressies. Het idee hier is dat we gebruik maken van tussenstappen in de vorm van *stelsels van vergelijkingen*. Kort samengevat is het procedé als volgt:

1. Vertaal de automaat naar een stelsel van vergelijkingen (zie Definitie 3.8 en Voorbeeld 3.6).
2. Los deze vergelijkingen op (zie Stelling 3.14 en Voorbeeld 3.16).

We ontwikkelen nu eerst wat terminologie over het soort vergelijkingen waar we mee te maken krijgen.

Definitie 3.3 Bij een gegeven verzameling X van *variabelen*, kun je de verzameling van *termen over* X heel kort als volgt definiëren:

$$t \ ::= \ x \in X \mid a \in \Sigma \mid \epsilon \mid \varnothing \mid t + t \mid tt \mid t^*. \tag{8}$$

Zoals gebruikelijk noemen we een term *gesloten* als er geen variabelen in voorkomen; de gesloten termen zijn dus precies de reguliere expressies.

Als elke variabele staat voor een formele taal over het alfabet Σ, dan kunnen we ook alle termen interpreteren als formele talen over Σ.

Definitie 3.4 Een *vergelijking* over een verzameling variabelen X is niets anders dan een paar (s, t) van termen. Standaard wordt zo'n vergelijking altijd genoteerd als $s = t$. Een *stelsel* van vergelijkingen is niets anders dan een verzameling van vergelijkingen.

Een *oplossing* voor een (stelsel) vergelijking(en) is een functie die aan elke variabele een formele taal toebedeelt, op zo'n manier dat alle vergelijkingen van het stelsel waar worden (dat wil zeggen, dat de linker- en de rechterkant van die vergelijking als dezelfde formele taal worden geïnterpreteerd).

Als er geen misverstand kan ontstaan over welke taal bij welke variabele hoort, zullen we gewoon zeggen dat een collectie formele talen (in plaats van een functie van variabelen naar formele talen) een oplossing is van een stelsel vergelijkingen.

Voorbeeld 3.5 Vergelijkingen kunnen nul, één of meer oplossingen hebben.

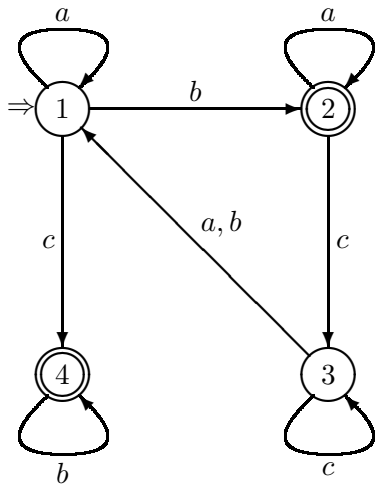
De vergelijking $x = xx$ bijvoorbeeld heeft meerdere oplossingen: als voorbeeld noemen we de talen $\{\epsilon\}$ (omdat $\{\epsilon\} \circ \{\epsilon\} = \{\epsilon\}$), en Σ^* (omdat $\Sigma^* \circ \Sigma^* = \Sigma^*$).

De vergelijking $ax = by + c$ daarentegen heeft geen enkele oplossing. Stel namelijk dat de talen K en L oplossingen zouden zijn voor respectievelijk x en y.. Dan zou moeten gelden, wegens $c \in L(b) \circ L \cup \{c\}$, dat $c \in L(a) \circ K$. Met andere woorden, we zouden het woord c moeten kunnen splitsen als $c = uv$ op zo'n manier dat $u \in L(a)$, dat wil zeggen: $u = a$, en $v \in L$. Dat is natuurlijk onmogelijk.

Ten slotte: de vergelijking $x = ax + \epsilon$ heeft precies één oplossing: de taal $L(a^*)$. Dat deze taal inderdaad een oplossing van de vergelijking is, is niet moeilijk in te zien. Dat het de *enige* oplossing is, volgt uit Stelling 3.14.

Laten we nu eerst eens kijken wat voor vergelijkingen het zijn die we kunnen destilleren uit eindige automaten. Eerst een voorbeeld.

Voorbeeld 3.6 Als voorbeeld: de automaat links levert het stelsel vergelijkingen rechts:



$$\begin{cases} x_1 &= ax_1 + bx_2 + cx_4 \\ x_2 &= ax_2 + cx_3 + \epsilon \\ x_3 &= ax_1 + bx_1 + cx_3 \\ x_4 &= bx_4 + \epsilon \end{cases}$$

Voordat we de algemene definitie geven, introduceren we eerst wat notatie.

Definitie 3.7 Gegeven een verzameling $T = \{t_1, \ldots, t_n\}$ van termen, noteren we $\sum T := t_1 + \cdots t_n$. In het speciale geval dat T de lege verzameling is, noteren we $\sum T := \varnothing$.

Om verwarring te voorkomen, zullen we als notatie voor het alfabet het symbool Ω hanteren zolang we het sommatieteken $\sum$ gebruiken.

Definitie 3.8 Gegeven is een niet-deterministische automaat $\mathbb{A} = (Q, q_I, \Omega, \Delta, F)$. Met deze automaat associëren we het volgende stelsel vergelijkingen over de verzameling variabelen $X_{\mathbb{A}} = \{x_q \mid q \in Q\}$ (dat wil zeggen: elke toestand uit Q heeft zijn eigen variabele).

Voor elke toestand $q \in Q$/variabele $x_q \in X$ introduceren we de volgende vergelijking:

$$\begin{array}{rcll} & x_q & = & \sum_{a\in\Omega} \sum_{p\in\Delta(q,a)} ax_p + \epsilon \quad \text{in het geval } q \in F, \\ \text{en} & x_q & = & \sum_{a\in\Omega} \sum_{p\in\Delta(q,a)} ax_p \qquad \text{in het geval } q \notin F, \end{array}$$

Anders geformuleerd, stel dat $q \xrightarrow{a_1} q_1, \ldots, q \xrightarrow{a_n} q_n$ *alle* transities vanuit toestand q zijn. De bijbehorende vergelijking voor q is dan

$$\begin{array}{rcll} & x_q & = & a_1q_1 + \cdots + a_nq_n + \epsilon \quad \text{in het geval } q \in F, \\ \text{en} & x_q & = & a_1q_1 + \cdots + a_nq_n \qquad \text{in het geval } q \notin F, \end{array}$$

Wat hebben deze vergelijkingen nu met de automaat te maken? Het antwoord op deze vraag wordt gegeven door de volgende stelling.

Stelling 3.9 *Gegeven is een eindige automaat $\mathbb{A} = (Q, q_I, \Omega, \Delta, F)$. Stel nu dat de collectie $\{L_q \mid q \in Q\}$ een oplossing is voor het met $\mathbb{A}$ geassocieerde stelsel vergelijkingen. Dan geldt $L(\mathbb{A}) = L(q_I)$.*

Bewijs. (schets) Laat, voor een gegeven toestand $q \in Q$, $\mathbb{A}_q = (Q, q, \Omega, \Delta, F)$ de *q-variant* van $\mathbb{A}$ zijn, dat wil zeggen: $\mathbb{A}_q$ is precies als $\mathbb{A}$, behalve dat q de begintoestand van $\mathbb{A}_q$ is. Stel nu dat de collectie $\{L_q \mid q \in Q\}$ een oplossing is voor het met $\mathbb{A}$ geassocieerde stelsel vergelijkingen.

We kunnen dan laten zien dat voor elk woord w, en elke toestand $q \in Q$ geldt:

$$w \in L(\mathbb{A}_q) \iff w \in L_q. \tag{9}$$

De details van het (inductieve) bewijs van (9) laten we achterwege. QED

Het punt is, dat *als* het stelsel oplossingen heeft, dan is de oplossing voor x_{q_I} precies de taal $L(\mathbb{A})$. De hamvraag is nu dus *of* de stelsels vergelijkingen die we met eindige automaten associëren, altijd oplossingen hebben, en of we die oplossing kunnen geven in de vorm van reguliere expressies. Om deze vraag te beantwoorden, introduceren we het concept van *bewakers* en van *bewaakte* vergelijkingen.

Definitie 3.10 Atomaire reguliere expressies van de vorm a of $\varnothing$ zijn altijd bewakers, maar de expressie ϵ is geen bewaker. Een expressie van de vorm $r + s$ is een bewaker als r en s *beide* bewakers zijn, terwijl een expressie van de vorm rs een bewaker is als ten minste één van de expressies r en s een bewaker is. Expressies van de vorm r^* zijn nooit bewakers.

Voorbeeld 3.11 De expressies $(a+b)(\epsilon+c)$, a^*b en $\varnothing\varnothing^*$ zijn bewakers, maar de expressies $a + b + \epsilon$, $(a + b)^*$ en $\varnothing^*$ niet.

Het idee achter dit begrip komt naar voren in het volgende lemma.

Lemma 3.12 *Voor elke expressie r geldt:*

$$r \text{ is een bewaker} \iff \epsilon \notin L(r). \tag{10}$$

Bewijs. Deze stelling bewijzen we met inductie naar de complexiteit van r.

basisstap De basisstap beslaat het geval dat r een atomaire expressie is. Er zijn dus drie mogelijkheden:

$r = a$ voor zeker symbool a. In dit geval is r een bewaker, en ook geldt $\epsilon \notin L(r) = \{a\}$.

$r = \varnothing$**.** In dit geval is r een bewaker, en ook geldt $\epsilon \notin L(r) = \varnothing$.

$r = \epsilon$**.** In dit geval is r géén bewaker, maar nu geldt $\epsilon \in L(r) = \{\epsilon\}$.

In alle gevallen geldt dus inderdaad (10).

inductiestap In de inductiestap bekijken we de gevallen waarin r een complexe expressie is; daarbij mogen we aannemen dat (10) waar is voor de deelexpressies waaruit r is opgebouwd. We onderscheiden de volgende mogelijkheden:

r **is van de vorm** $r_1 + r_2$**.** Om (10) te bewijzen, moeten we twee implicaties aantonen.
Stel eerst dat r een bewaker is. Per definitie zijn r_1 en r_2 dan allebei bewakers. Volgens de inductiehypothese geldt dus dat $\epsilon \notin L(r_1)$ en $\epsilon \notin L(r_2)$. Maar dan geldt $\epsilon \notin L(r_1) \cup L(r_2) = L(r)$.
Stel nu omgekeerd dat $\epsilon \notin L(r)$; omdat $L(r) = L(r_1) \cup L(r_2)$ vinden we dat ϵ dus noch in $L(r_1)$, noch in $L(r_2)$ kan zitten. Op grond van de inductiehypothese zien we dus dat r_1 en r_2 allebei bewakers zijn. Maar dan is r zelf ook een bewaker.

r **is van de vorm** $r_1 r_2$**.** Om (10) te bewijzen is het natuurlijk voldoende om te laten zien dat

$$r \text{ is géén bewaker} \iff \epsilon \in L(r). \tag{11}$$

We bewijzen beide implicaties van (11) weer apart.
Stel eerst dat r géén bewaker is. Per definitie zijn dan r_1 en r_2 allebei geen bewaker, zodat uit de inductiehypothese volgt dat $\epsilon \in L(r_1)$ en $\epsilon \in L(r_2)$. Maar dan geldt $\epsilon = \epsilon\epsilon \in L(r_1) \circ L(r_2) = L(r_1 r_2) = L(r)$.
Omgekeerd, als $\epsilon \in L(r)$ moeten we het woord ϵ kunnen splitsen als $\epsilon = u_1 u_2$ waarbij $u_1 \in L(r_1)$ en $u_2 \in L(r_2)$. Nu is er maar één mogelijkheid om het lege woord te splitsen, dus $u_1 = \epsilon$ en $u_2 = \epsilon$. We zien dus dat ϵ een element is van zowel van $L(r_1)$ als van $L(r_2)$. We mogen dan uit de inductiehypothese concluderen dat r_1 en r_2 beide geen bewaker zijn; per definitie is r het dan ook niet.
Dit bewijst (11) en daarmee (10).

r **is van de vorm** r_1^***.** Dit geval is makkelijk: per definitie geldt dat r géén bewaker is, en ook per definitie geldt dat $\epsilon \in L(r_1^*)$.

QED

Definitie 3.13 Een vergelijking heet *bewaakt* als hij van de vorm

$$x = gx + t$$

is, waarbij g een bewaker is, en t een term waarin x niet voorkomt. Een bewaakte vergelijking heet *lineair* als hij van de vorm

$$x = \sum_i r_i x_i + s$$

is, waarbij elke r_i en s gesloten termen zijn. Een stelsel vergelijkingen heet *bewaakt* als elke vergelijking bewaakt is, en *lineair* als het bestaat uit lineaire vergelijkingen.

Een stelsel vergelijkingen heet *vol* als er voor elke variabele x die in het stelsel voorkomt, precies één vergelijking van de vorm $x = t(x_1, \ldots, x_n)$ is.

Merk op dat eindige automaten volle, bewaakte, lineaire stelsels vergelijkingen opleveren.

Stelling 3.14 *Elk vol, bewaakt en lineair stelsel van vergelijkingen heeft een unieke oplossing, die effectief uit het stelsel te verkrijgen is in de vorm van een verzameling reguliere expressies.*

Bewijs. Eén voor één elimineren we variabelen uit het stelsel, gebruik makend van de regel

$$\frac{x = gx + t}{x = g^*t} \qquad (R)$$

Deze regel mag worden toegepast mits in de vergelijking $x = gx + t$ de term g een bewaker is, en de variabele x niet voorkomt in de term t. (De correctheid van de regel wordt gegeven door Lemma 3.17.)

Dit proces termineert als we bij de laatste vergelijking zijn aanbeland. Deze vergelijking brengen we in de vorm

$$x = gx + r,$$

waarbij g en r allebei gesloten termen zijn, en g bovendien een guard. De oplossing van x is dan

$$x = g^*r,$$

oftewel $x = L(g)^* \circ L(r)$. Nu kunnen we één voor één de oplossingen voor de andere variabelen vinden door stapsgewijze invulling. QED

Voorbeelden maken het idee waarschijnlijk duidelijker.

Voorbeeld 3.15 Beschouw de vergelijking $x = ax + b$. Omdat a een bewaker is, mogen we de regel (R) toepassen. We krijgen dan als oplossing: $x = a^*b$.

Beschouw, als variant hiervan, de vergelijking $x = ax + \epsilon$. Toepassing van (R) levert als oplossing: $x = a^*\epsilon$; dit kunnen we vereenvoudigen tot $x = a^*$.

Ten slotte, we kunnen de regel (R) ook toepassen op vergelijkingen van de vorm $x = gx$, tenminste, als g een bewaker term is. Gebruik makend van de equivalentie tussen gx en $gx + \varnothing$, kunnen we de vergelijking herschrijven als $x = gx + \varnothing$; nu toepassen van (R) geeft als oplossing $x = g^*\varnothing$, en dat kunnen we weer vereenvoudigen tot $x = \varnothing$. En inderdaad, omdat $\epsilon \notin L(g)$, is de lege verzameling de enige formele taal L waarvoor geldt dat $L = L(g) \circ L$.

Voorbeeld 3.16 We zagen al dat de automaat van Voorbeeld 3.6 het volgende stelsel vergelijkingen oplevert:

$$\begin{cases} x_1 &= ax_1 + bx_2 + cx_4 \\ x_2 &= ax_2 + cx_3 + \epsilon \\ x_3 &= ax_1 + bx_1 + cx_3 \\ x_4 &= bx_4 + \epsilon \end{cases} \tag{12}$$

Het verdient aanbeveling om, in de rechterkant van deze vergelijkingen, alle termen met dezelfde variabele samen te nemen:

$$\begin{cases} x_1 &= ax_1 + bx_2 + cx_4 \\ x_2 &= ax_2 + cx_3 + \epsilon \\ x_3 &= (a+b)x_1 + cx_3 \\ x_4 &= bx_4 + \epsilon \end{cases} \tag{13}$$

We beginnen met de eliminatie van x_4; uit de laatste vergelijking volgt met de regel (R) dat

$$x_4 = b^*. \tag{14}$$

Vullen we dit in het stelsel (13) in, dan krijgen we

$$\begin{cases} x_1 &= ax_1 + bx_2 + cb^* \\ x_2 &= ax_2 + cx_3 + \epsilon \\ x_3 &= (a+b)x_1 + cx_3 \end{cases} \tag{15}$$

Als de volgende stap elimineren we x_3 uit het stelsel; daartoe herschikken we de laatste vergelijking tot

$$x_3 = cx_3 + (a+b)x_1, \tag{16}$$

omdat we met (R) hieruit mogen concluderen dat

$$x_3 = c^*(a+b)x_1, \tag{17}$$

waarmee we x_3 als een directe functie van x_1 en x_2 hebben uitgedrukt. We kunnen x_3 nu inderdaad elimineren uit (15), namelijk door in de twee bovenste vergelijkingen de rechterterm $c^*(a+b)x_1$ van (17) voor x_3 te *substitueren*. We krijgen dan

$$\begin{cases} x_1 &= ax_1 + bx_2 + cb^* \\ x_2 &= ax_2 + cc^*(a+b)x_1 + \epsilon \end{cases} \tag{18}$$

Nu kunnen we uit dit nieuwe stelsel vergelijkingen één van de twee overgebleven variabelen elimineren, bijvoorbeeld x_2. Uit de vergelijking

$$x_2 = ax_2 + cc^*(a+b)x_1 + \epsilon \tag{19}$$

volgt namelijk, opnieuw met (R), dat

$$x_2 = a^*(cc^*(a+b)x_1 + \epsilon). \tag{20}$$

Als we vervolgens deze (deel)oplossing voor x_2 in de eerste vergelijking van (18) invullen, dan krijgen we

$$x_1 = ax_1 + ba^*(cc^*(a+b)x_1 + \epsilon) + cb^*. \tag{21}$$

Wat herschrijven geeft

$$x_1 = (a + ba^*cc^*(a+b))\, x_1 + ba^* + cb^*, \tag{22}$$

zodat we als oplossing voor x_1 vinden:

$$x_1 = (a + ba^*(cc^*(a+b)))^*\, (ba^* + cb^*). \tag{23}$$

Ten slotte, achtereenvolgens invullen van de oplossingen (20), (17) en (14) geeft de unieke oplossing van (12). Om deze oplossing wat leesbaarder te houden, herschrijven we eerst (20) tot $x_2 = a^*cc^*(a+b)x_1 + a^*$. Nu invullen van (23) geeft:

$$\begin{cases} x_1 &= (a + ba^*(cc^*(a+b)))^*\, (ba^* + cb^*) \\ x_2 &= a^*cc^*(a+b)\, (a + ba^*(cc^*(a+b)))^*\, (ba^* + cb^*) + a^* \\ x_3 &= c^*(a+b)\, (a + ba^*(cc^*(a+b)))^*\, (ba^* + cb^*) \\ x_4 &= b^*. \end{cases} \tag{24}$$

Het zal duidelijk zijn dat de regel (R) een cruciale rol speelt in het oplossen van vergelijkingen. De correctheid van deze regel volgt uit het volgende lemma.

Lemma 3.17 *Stel dat X, L en M formele talen zijn, en dat $\epsilon \notin L$. Dan geldt*

$$X = L \circ X \cup M \iff X = L^* \circ M. \tag{25}$$

Bewijs. We laten de implicatie van rechts naar links in (25) als een oefening voor de lezer, en concentreren ons op de implicatie van links naar rechts. Stel dus dat $X = L \circ X \cup M$. We zullen laten zien dat

$$X = L^* \circ M. \tag{26}$$

Eerst bewijzen we dat $L^* \circ M \subseteq X$. Neem een willekeurig woord $w \in L^* \circ M$; w kan dus worden geschreven als $w = v \in M$ of als $w = u_1 \cdots u_n v$ met elke $u_i \in L$ en $v \in M$. In het eerste geval concluderen we dat $w \in X$ omdat $M \subseteq X$. In het tweede geval beginnen we met te constateren dat $v \in X$, weer omdat $M \subseteq X$. Hieruit volgt dat $u_n v \in L \circ X \subseteq X$; achtereenvolgens kunnen we dan laten zien dat $u_{n-1}(u_n v)$, $\ldots$, $u_2(u_3 \cdots u_n v)$ en ten slotte $w = u_1(u_2 \cdots u_n v)$ in X zitten.

Voor de andere inclusie $(\subseteq)$ van (26) laten we zien dat dat voor ieder woord w geldt:

$$w \in X \;\Rightarrow\; w \in L^* \circ M. \tag{27}$$

Het bewijs van (27) gaat met inductie naar de lengte van w.

In de *basisstap* bekijken we het geval dat $|w| = 0$, dat wil zeggen: w is het lege woord. Stel dat $\epsilon \in X$, zodat uit de aanname $X = L \circ X \cup M$ volgt dat $\epsilon \in L \circ X \cup M$. Maar uit

$\epsilon \notin L$ volgt $\epsilon \notin L \circ X$, zodat we mogen concluderen dat $\epsilon \in M$. Uit $M \subseteq L^* \circ M$ volgt dan onmiddelijk dat $\epsilon \in L^* \circ M$.

In de *inductiestap* bekijken we het geval dat $|w| > 0$. Stel dat w een woord in X is; wegens $X = L \circ X \cup M$ zijn er twee mogelijkheden: $w \in L \circ X$ of $w \in M$. In het tweede geval is het meteen duidelijk dat $w = \epsilon w \in L^* \circ M$. In het eerste geval weten we in eerste instantie alleen dat w gesplitst kan worden als $w = uv$, waarbij $u \in L$ en $v \in X$. Omdat $\epsilon \notin L$ moet u dus *verschillen* van het lege woord ϵ, waarmee v *korter* is dan w. We mogen dus de inductiehypothese toepassen op v, en zien dat $v \in L^* \circ M$. Maar dan vinden we dat $w = uv \in L \circ (L^* \circ M) = (L \circ L^*) \circ M \subseteq L^* \circ M$, en dus zijn we klaar met het bewijs van (27). QED

3.2 Eindige automaten uit reguliere expressies

Omgekeerd is het veel eenvoudiger om, uitgaande van een reguliere expressie r, een automaat $\mathbb{A}_r$ te construeren die precies de woorden herkent die matchen met r. Tenminste, als we een variant van de nondeterministische automaat toestaan waarin zogenaamde *stille stappen* mogelijk zijn.

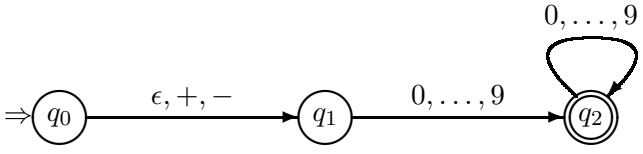
Definitie 3.18 Een *niet-deterministische eindige automaat met stille stappen*, kortweg: ϵ-NEA of *ϵ-automaat*, is een quintupel $\mathbb{A} = (Q, q_I, \Sigma, \Delta, F)$, waarbij Q, q_I, Σ en F als in een gewone niet-deterministische automaat zijn, terwijl $\Delta : Q \times (\Sigma \cup \{\epsilon\}) \to \mathcal{P}(Q)$ nu, naast de gewone transities, ook zogenaamde *ϵ-transities* of *stille stappen* toestaat. Dit zijn transities van de vorm $q \xrightarrow{\epsilon} q'$.

Veel begrippen die we kennen voor gewone niet-deterministische automaten moeten voor de variant met stille stappen enigszins worden aangepast. Dit spreekt eigenlijk altijd voor zich — we vermelden hier alleen het volgende.

Definitie 3.19 Gegeven is een ϵ-NEA $\mathbb{A} = (Q, q_I, \Sigma, \Delta, F)$. Een *succesvolle run* van $\mathbb{A}$ op een woord $w \in \Sigma^*$ is een gelabeld pad $q_I \xrightarrow{\ell_1} q_1 \xrightarrow{\ell_2} q_2 \cdots \xrightarrow{\ell k} q_k$ zó dat $q_k \in F$, terwijl w het woord is dat je krijgt als de concatenatie van alle labels $\ell_1, \ldots, \ell_k$ (waarbij deze concatenatie alle voorkomens van ϵ doet wegvallen.)

De automaat $\mathbb{A}$ *herkent* of *accepteert* een woord w als $\mathbb{A}$ minimaal één succesvolle run of w heeft.

Voorbeeld 3.20 Stel dat we de gehele getallen als volgt weergeven als woorden over het alfabet $\Sigma = \{0, \ldots, 9, +, -\}$: het getal is een optioneel $+/-$ teken, gevolgd door een rij van één of meer cijfers. Deze taal wordt herkend door de volgende ϵ-automaat:



Een sucessvolle run op het woord 305 is bijvoorbeeld: $q_0 \xrightarrow{\epsilon} q_1 \xrightarrow{3} q_2 \xrightarrow{0} q_2 \xrightarrow{5} q_2$.

Op het eerste gezicht zou je misschien kunnen denken dat ϵ-NEA's meer talen kunnen herkennen dan gewone NEA's, maar dat is niet zo. Met een lichte variatie op de subset constructie van Definitie 2.17 kun je de volgende stelling bewijzen.

Stelling 3.21 *Voor elke niet-deterministische eindige automaat met stille stappen $\mathbb{A}$ bestaat een deterministische eindige automaat $\mathbb{A}^d$ zodanig dat $L(\mathbb{A}) = L(\mathbb{A}^d)$.*

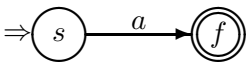
Dat betekent dat het, om de implicatie $\Leftarrow$ van Kleene's stelling te bewijzen, voldoende is om te laten zien dat we bij elke reguliere expressie een equivalente ϵ-NEA kunnen construeren.

Stelling 3.22 *Voor elke reguliere expressie r is er een niet-deterministische eindige automaat $\mathbb{A}_r$ met stille stappen, zodanig dat $L(r) = L(\mathbb{A}_r)$.*

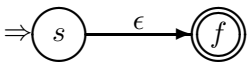
Bewijs. We gaan een bewering bewijzen die net iets sterker is dan degene in de formulering van de stelling. Noem een ϵ-NEA $\mathbb{A} = (Q, q_I, \Sigma, \Delta, F)$ *speciaal* als er precies één accepterende toestand f is, en als er bovendien uit deze f geen enkele pijl vertrekt.

We zullen laten zien dat er bij elke reguliere expressie r een speciale ϵ-NEA $\mathbb{A}_r$ is zó dat $L(r) = L(\mathbb{A}_r)$. Deze bewering bewijzen met *inductie naar de complexiteit van de reguliere expressie r*.

In de *basisstap* van het bewijs hebben we te maken met een atomaire reguliere expressie r. Er zijn dus drie mogelijkheden: $r = a$ voor een of ander symbool $a \in \Sigma$, $r = \epsilon$ of $r = \varnothing$. Afhankelijk van het type van r, definiëren we $\mathbb{A}_r$ als één van de onderstaande automaten:



$\mathbb{A}_a$



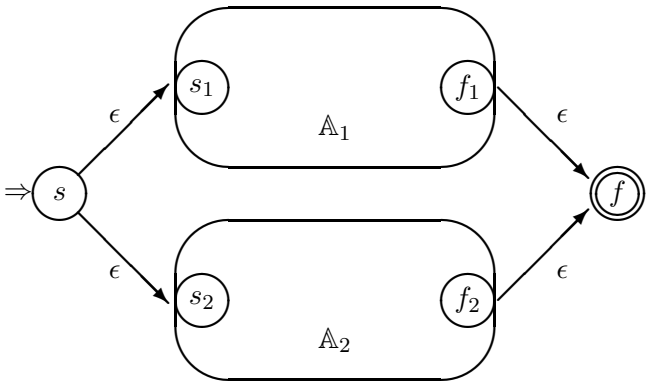
$\mathbb{A}_\epsilon$

$\mathbb{A}_\varnothing$

Merk op dat deze automaten inderdaad speciaal zijn in de zin dat ze precies één eindtoestand hebben, vanwaar geen enkele pijl vertrekt. We laten het over aan de lezer om te laten zien dat $L(\mathbb{A}_a) = \{a\} = L(a)$, etc.

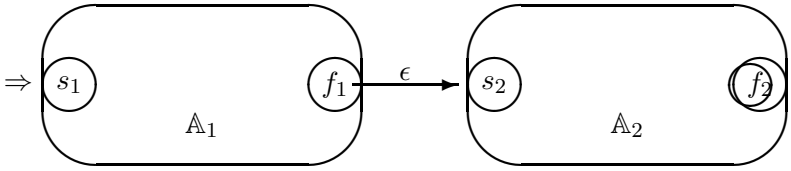
In de *inductiestap* van het bewijs hebben we te maken met een complexe reguliere expressie r. Inductief mogen we aannemen dat we al automaten hebben geconstrueerd voor alle deelexpressies van r. Onderscheid nu de volgende gevallen:

Geval $r = r_1 + r_2$**.** We mogen er dus vanuit gaan dat er speciale automaten $\mathbb{A}_1$ en $\mathbb{A}_2$ zijn zó dat $L(\mathbb{A}_1) = L(r_1)$ en $L(\mathbb{A}_2) = L(r_2)$. Op basis van deze twee automaten definiëren we nu de automaat $\mathbb{A}$ als in het onderstaande plaatje.



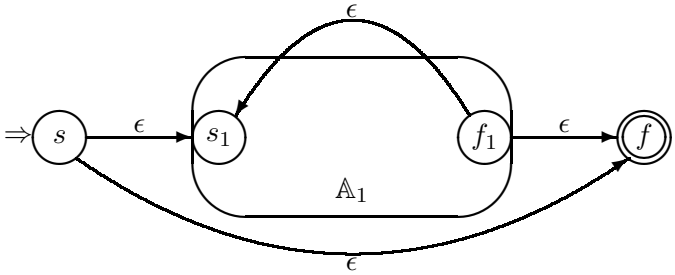
Het is niet moeilijk te zien dat $\mathbb{A}$ weer een speciale ϵ-automaat is, en dat $L(\mathbb{A}) = L(\mathbb{A}_1) \cup L(\mathbb{A}_2) = L(r_1) \cup L(r_2) = L(r_1 + r_2) = L(r)$.

Geval $r = r_1 r_2$**.** Opnieuw mogen we er vanuit gaan dat er speciale automaten $\mathbb{A}_1$ en $\mathbb{A}_2$ zijn met $L(\mathbb{A}_1) = L(r_1)$ en $L(\mathbb{A}_2) = L(r_2)$. Met behulp van deze twee automaten construeren we als volgt een nieuwe automaat $\mathbb{A}$:



We laten het opnieuw over aan de lezer om te laten zien dat $L(\mathbb{A}) = L(\mathbb{A}_1) \circ L(\mathbb{A}_2) = L(r_1) \circ L(r_2) = L(r_1 r_2) = L(r)$.

Geval $r = r_1^*$**.** Inductief mogen we aannemen dat er een speciale ϵ-automaat $\mathbb{A}_1$ is waarvoor geldt: $L(\mathbb{A}_1) = L(r_1)$. Definieer nu de automaat $\mathbb{A}$ als in het onderstaande plaatje.



(Voor een precieze definitie: $\mathbb{A}$ wordt gegeven door als verzameling toestanden te nemen $Q := Q_1 \cup \{s, f\}$, met s als starttoestand, en f als accepterende toestand. De transitiefunctie Δ van $\mathbb{A}$ is precies als de Δ_1 van $\mathbb{A}_1$, met de volgende verschillen: $\Delta(s, a) = \Delta(f, a) = \varnothing$ voor alle $a \in \Sigma$, $\Delta(s, \epsilon) := \{s_1, f\}$ en ook $\Delta(f_1, \epsilon) := \{s_1, f\}$, terwijl $\Delta(f, \epsilon) := \varnothing$.)

We claimen nu dat

$$L(\mathbb{A}) = (L(\mathbb{A}_1))^*. \tag{28}$$

We behandelen eerst de inclusie $\subseteq$. Stel dat w tot de taal $L(\mathbb{A})$ behoort; er is dus een succesvolle run van $\mathbb{A}$ op w. Als dit pad door de graaf van $\mathbb{A}$ direct van s naar f loopt,

dan is w het lege woord ϵ, en zit w dus per definitie in $(L(\mathbb{A}_1))^*$. Als het pad niet direct van s naar f loopt, dan bestaat de run dus uit de begintransitie $s \xrightarrow{\epsilon} s_1$, de eindtransitie $f_1 \xrightarrow{\epsilon} f$, met daartussen een middenstuk. Dit middenstuk bestaat uit een aantal (één of meer) paden van s_1 naar f_1, die elk corresponderen met een succesvolle run van $\mathbb{A}_1$, en steeds van elkaar worden gescheiden[1] door de stille stap $f_1 \xrightarrow{\epsilon} s_1$. Op grond hiervan is het duidelijk dat we w kunnen opsplitsen als $w = u_1 \cdots u_n$ zó dat elke u_i door $\mathbb{A}_1$ wordt herkend. Er geldt dan dus dat $w \in (L(\mathbb{A}_1))^*$.

Omgekeerd, als w behoort tot de taal $(L(\mathbb{A}_1))^*$, dan geldt dat $w = \epsilon$, of w kan worden opgesplitst als $w = u_1 \cdots u_n$ zó dat elke u_i door $\mathbb{A}_1$ wordt herkend. In het eerste geval is $s \xrightarrow{\epsilon} f$ een succesvolle run van $\mathbb{A}$ op $w = \epsilon$. In het tweede geval geeft $s \xrightarrow{\epsilon} s_1 \overset{u_1}{\twoheadrightarrow} f_1 \xrightarrow{\epsilon} s_1 \overset{u_2}{\twoheadrightarrow} \ldots \overset{u_n}{\twoheadrightarrow} f_1 \xrightarrow{\epsilon} f$ een succesvolle run. In beide gevallen geldt dus dat $\mathbb{A}$ het woord w accepteert, zodat we zeker zijn dat $w \in L(\mathbb{A})$.

QED

De constructie van Stelling 3.22 blinkt niet uit in zuinigheid. Bijvoorbeeld, de automaat die je krijgt voor de expressie $ab^* + b$ heeft maar liefst tien toestanden, terwijl er een heel simpel alternatief is me niet meer dan drie toestanden.

[1] *Hier* maken we gebruik van het feit dat $\mathbb{A}_1$ een unieke eindtoestand f_1 heeft, en dat er vanuit f_1 in $\mathbb{A}_1$ geen enkele transitie vertrekt.