Automaten

Informatica, UvA

Yde Venema

Inhoud

1 Formele talen en reguliere expressies

1.1 Formele talen

Definitie 1.1 Een *alfabet* is een niet-lege, eindige verzameling van objecten die we *symbolen* of *letters* zullen noemen.

Voorbeeld 1.2 Voorbeelden van alfabetten zijn:

$$\begin{aligned}\Sigma_1 &:= \{0,1,2,3,4,5,6,7,8,9\},\\ \Sigma_2 &:= \{a,b,c,\ldots,x,y,z\},\\ \Sigma_3 &:= \Sigma_1 \cup \Sigma_2 \cup \{+,-,*,=,:=\} \cup \{\texttt{;},\texttt{if},\texttt{then},\texttt{else},\texttt{fi},\texttt{while},\texttt{do},\texttt{od}\}\end{aligned}$$

De Nederlandse taal bevat oneindig veel woorden, zodat de verzameling van alle Nederlandse woorden géén voorbeeld van een alfabet is.

Definitie 1.3 Een *woord* of *string* over een alfabet Σ is een eindig rijtje symbolen uit Σ. Het *lege woord* wordt weergegeven als ϵ. De verzameling van alle woorden over Σ wordt weergegeven als Σ^*.

Voorbeeld 1.4 De verzameling $\{a,b\}^*$ bestaat uit de woorden $\epsilon, a, b, aa, ab, ba, bb, aaa, \ldots$

Definitie 1.5 Het samenvoegen van twee woorden door ze achter elkaar te zetten noemen we *concatenatie*; dezelfde term gebruiken we ook voor het product van deze operatie.

Voorbeeld 1.6 De concatenatie van de woorden *ab* en *bbac* geeft (of *is*) het woord *abbbac*, concatenatie van de woorden *ab*, *bc*, ϵ en *cd* levert het woord *abbccd* op.

Van belang zijn verder de volgende afgeleide begrippen.

Definitie 1.7 De *lengte* van een woord w, notatie: $|w|$, definiëren we als het aantal symbolen van w. De *frequentie* van een symbool in een woord is het aantal keren dat het symbool in het woord voorkomt.

Voorbeeld 1.8 Beschouw de statement "`if` $x = 3$ `then` $y := z$" als woord over het alfabet Σ_3 uit Voorbeeld 1.2. Er staan acht Σ_3-symbolen in dit woord (merk op dat de spatie geen element van Σ_3 is!). Hiermee is de lengte vastgesteld: $|\texttt{if}\ x = 3\ \texttt{then}\ y := z| = 8$. De lengte van het lege woord is nul: $|\epsilon| = 0$.

Voorbeelden van frequenties: $\#_a(aabaaac) = 5$, $\#_b(aabaaac) = 1$ en $\#_d(aabaaac) = 0$.

Definitie 1.9 Een *(formele) taal* over een alfabet Σ is een verzameling woorden over Σ.

Dat wil zeggen: elke deelverzameling van Σ^* is een formele taal over Σ.

Voorbeeld 1.10 Stel eerst dat we als alfabet de verzameling $\Sigma = \{a, b\}$ nemen. Voorbeelden van formele talen zijn:

$$\begin{aligned} L_1 &= \{w \in \Sigma^* \mid |w| = 2\}, \\ L_2 &= \{w \in \Sigma^* \mid |w| > 2\}, \\ L_3 &= \{w \in \Sigma^* \mid \#_a w = k\}. \end{aligned}$$

Oftewel: $L_1 = \{aa, ab, ba, bb\}$, L_2 bestaat uit alle Σ-woorden die minstens drie symbolen bevatten, en L_3 bestaat uit die Σ-woorden waar precies k maal het symbool a in voorkomt.

Andere voorbeelden: de Nederlandse taal, opgevat als de verzameling (correct gespelde) Nederlandse woorden, kun je zien als een formele taal over het alfabet $\{a, b, \ldots, z\}$. Programmeertalen zijn altijd formele talen over een alfabet vergelijkbaar met Σ_3 uit Voorbeeld 1.2.

Omdat formele talen gedefinieerd worden als *verzamelingen*, kun je er de gebruikelijke operaties op toepassen, zoals vereniging, doorsnede, en complement (dat wil zeggen, complement met betrekking tot de verzameling Σ^*). Daarnaast zijn er nog een aantal andere operaties op formele talen.

Definitie 1.11 Gegeven formele talen L over een alfabet Σ, definiëren we de volgende nieuwe formele talen:

$$\begin{aligned} L \circ L' &= \{ww' \mid w \in L, w' \in L'\} \\ L^0 &= \{\epsilon\} \\ L^{n+1} &= L \circ L^n \\ L^* &= \bigcup_{n \geq 0} L^n \end{aligned}$$

Voorbeeld 1.12 Beschouw het alfabet $\Sigma = \{a, b, c\}$, en definieer, voor elk natuurlijk getal k:

$$L_k := \{w \in \Sigma^* \mid \#_a w = k\}.$$

Dat wil zeggen: L_k bestaat uit die woorden over Σ waar precies k maal de letter a in voorkomt. Dan geldt bijvoorbeeld:

$$\begin{aligned} L_3 \circ L_5 &= L_8, \\ (L_2)^0 &= \{\epsilon\}, \\ (L_2)^n &= L_{2n}, \\ (L_2)^* &= \{w \in \Sigma^* \mid \#_a w \text{ is even}\}. \end{aligned}$$

1.2 Reguliere expressies

Definitie 1.13 Gegeven een alfabet Σ, definiëren we de collectie van *reguliere expressies over* Σ door middel van de volgende Backus-Naur vorm (BNF):

$$r \quad ::= \quad a \mid \epsilon \mid \varnothing \mid r + r \mid rr \mid r^*. \tag{1}$$

Hierbij staat a voor een willekeurig symbool uit Σ.

Voor wie niet bekend is met deze manier van definiëren: u dient (1) te zien als een verkorte notatie van de volgende inductieve definitie:

- alle symbolen $a \in \Sigma$, tesamen met ϵ en $\varnothing$, vormen de *atomaire* reguliere expressies;
- als r en s reguliere expressies zijn, dan ook $r + s$, rs en r^*.

Voorbeeld 1.14 Als voorbeelden van reguliere expressies over het alfabet $\Sigma = \{a, b\}$ noemen we: a^*, a^*b^*, ϵ, en $(a^*\varnothing + \epsilon b)^*$.

Reguliere expressies over een alfabet Σ kunnen worden gezien als *condities* waar een woord uit Σ^* al dan niet aan kan voldoen. In het geval een woord w voldoet aan de expressie r, spreken we van *matching*, notatie: $w \rhd r$. Om precies te bepalen *wanneer* een woord matcht met een expressie, gebruiken we de volgende definitie.

Definitie 1.15 Gegeven is een alfabet Σ. Met inductie naar de complexiteit van de reguliere expressie r bepalen we of een woord w met deze expressie matcht:

$$\begin{aligned}
w \rhd a &\quad \text{als} \quad w = a, \\
w \rhd \epsilon &\quad \text{als} \quad w = \epsilon, \\
w \rhd \varnothing &\quad \text{nooit} \\
w \rhd r + s &\quad \text{als} \quad w \rhd r \text{ of } w \rhd s, \\
w \rhd rs &\quad \text{als} \quad w \text{ kan worden gesplitst als } w = uv \text{ zó dat } u \rhd r \text{ en } v \rhd s, \\
w \rhd r^* &\quad \text{als} \quad w = \epsilon \text{ of } w \rhd r \text{ of } w \text{ kan worden gesplitst als } w = w_1 \dots w_k \\
&\qquad\qquad\qquad\qquad \text{zó dat } w_i \rhd r \text{ voor elke } i.
\end{aligned}$$

In woorden drukt de laatste clausule uit dat een woord w matcht met een expressie van de vorm r^* als $w = \epsilon$ of w matcht met r, of w kan worden geschreven als de concatenatie van een aantal strings die allemaal met r matchen.

Voorbeeld 1.16 Laat $\Sigma = \{a, b, c\}$.

Dan geldt dat een Σ-woord w matcht met de expressie $(b + c)^*$ dan en slechts dan als er in w géén a's voorkomen (ga na).

Verder geldt bijvoorbeeld dat het woord $w = bcaac$ matcht met de expressie $r = (b + c)^*a(b + c)^*a(b + c)^*$. We kunnen namelijk w opsplitsen in de woorden bc, a, ϵ, a en c, en voor deze woorden geldt:

- $bc \rhd (b+c)^*$ omdat $b \rhd b+c$ en $c \rhd b+c$,
- $a \rhd a$.
- $\epsilon \rhd (b+c)^*$,
- $a \rhd a$,
- $c \rhd (b+c)^*$ omdat $c \rhd b+c$.

In het algemeen geldt dat een woord w matcht met $r = (b+c)^*a(b+c)^*a(b+c)^*$ als w precies tweemaal het symbool a bevat, en dus geldt dat $w \rhd r^*$ als w een *even* aantal a's bevat.

De link tussen reguliere expressies en formele talen komt in de volgende definitie naar voren.

Definitie 1.17 Een reguliere expressie r *beschrijft* een taal $L(r)$, gegeven door de verzameling woorden die aan r voldoen. Preciezer:

$$L(r) := \{w \in \Sigma^* \mid w \rhd r\}.$$

Voorbeeld 1.18 Hier zijn een aantal eenvoudige voorbeelden van talen over $\{a,b,c\}^*$:

$$\begin{aligned} L(a+b+\epsilon) &= \{a, b, \epsilon\}, \\ L(a+b+\varnothing) &= \{a, b\}, \\ L(a+b^*) &= \{a, \epsilon, b, bb, bbb, \ldots\}, \\ L((a+b)^*) &= \{\epsilon, a, b, aa, ab, ba, bb, aaa, \ldots\}, \\ L(a(a+b+c)^*) &= \{w \in \{a,b,c\}^* \mid w \text{ begint met een } a\ \}, \end{aligned}$$

Aan het einde van Voorbeeld 1.16 zagen we al dat

$$L\Big(((b+c)^*a(b+c)^*a(b+c)^*)^*\Big) = \{w \in \{a,b,c\}^* \mid \#_a w \text{ is even}\}.$$

Als laatste nog een wat moeilijker voorbeeld: de taal bestaande uit alle woorden in $\{a,b,c\}^*$ die *niet* met de string ab beginnen, wordt beschreven door de expressie $\epsilon + a + (b+c)(a+b+c)^* + a(a+c)(a+b+c)^*$.

Voorbeeld 1.19 Als een uitgewerkt voorbeeld laten we zien, voor het geval dat $\Sigma = \{a,b\}$, dat

$$L(b^*(a(\epsilon + bbb^*))^*(\epsilon + b)) = \{w \in \Sigma^* \mid w \text{ bevat } aba \text{ niet als deelstring }\}.$$

We behandelen eerst de inclusie $\subseteq$. Stel dus dat $w \in \{a,b\}^*$ matcht met de expressie $b^*(a(\epsilon + bbb^*))^*(\epsilon + b)$. Dan is w dus van de vorm $w = w_1w_2w_3$, zó dat

- $w_1 \rhd b^*$, dus w_1 bevat geen a's;
- $w_2 \rhd (a(\epsilon + bbb^*))^*$, dus w_2 kan gesplitst worden in een aantal woorden die elk matchen met de expressie $a(\epsilon + bbb^*)$;

- $w_3 \rhd (\epsilon + b)$, dus w_3 bevat ook geen a's.

Kan w het woord aba als deelstring bevatten? De enige mogelijkheid is dat aba in zijn geheel een deelstring van w_2 vormt, dus laten we nog wat beter kijken naar w_2. Wegens de bovenstaande analyse moet w_2, als het überhaupt meer dan één a bevat, van de vorm

$$\underbrace{ab\cdots b} \cdots \underbrace{ab\cdots b}$$

zijn, waarbij het aantal b's tussen opeénvolgende a's steeds óf nul (ϵ), óf minimaal twee (bbb^*) moet zijn. Het kan dus niet voorkomen, dat twee a's worden gescheiden door precies één b. Maar dan kan w_2 het woord aba niet als deelstring bevatten, en w zelf dus ook niet. Dit bewijst de inclusie $\subseteq$.

Voor de andere inclusie nemen we aan dat de string aba niet voorkomt in w. Dat betekent dat er tussen elke twee opeénvolgende a's in w, ofwel géén, ofwel minimaal twee b's voorkomen. Het woord w moet dan wel van de volgende vorm zijn:

- eerst een aantal (nul of meer) b's;
- dan een aantal keren (afhankelijk van de frequentie van a in w) een woord van de vorm ab^n, waarbij $n = 0$ of $n \geq 2$;
- het woord kan eventueel worden afgesloten door één enkele b.

Met andere woorden, w kan worden gesplitst in drie woorden die respectievelijk moeten matchen met de expressies b^*, $(a(\epsilon + bbb^*))^*$ en $\epsilon + b$. Hieruit volgt onmiddellijk dat w moet matchen met de expressie $b^*(a(\epsilon + bbb^*))^*(\epsilon + b)$.

De volgende propositie maakt het verband met de operaties van Definitie 1.11 duidelijk.

Propositie 1.20 Gegeven is een alfabet Σ. Er geldt, voor elk symbool $a \in \Sigma$, en voor alle reguliere expressies r en s over Σ:

$$\begin{aligned} L(\varnothing) &= \varnothing \\ L(\epsilon) &= \{\epsilon\} \\ L(a) &= \{a\} \\ L(r+s) &= L(r) \cup L(s) \\ L(rs) &= L(r) \circ L(s) \\ L(r^*) &= (L(r))^* \end{aligned}$$

1.3 Equivalentie van reguliere expressies

Voorbeeld 1.21 De expressies $bc(bc)^*$ en $(bc)^*bc$ beschrijven precies dezelfde taal, namelijk die bestaande uit de woorden $bc, bcbc, bcbcbc, \ldots$ Meer in het algemeen kun je laten zien dat voor elke expressie r geldt dat $L(rr^*) = L(r^*r)$. De expressies rr^* en r^*r heten daarom *equivalent*.

Definitie 1.22 Twee reguliere expressies heten *equivalent* als ze één en dezelfde taal beschrijven.

Voorbeeld 1.23 Eerst een tegenvoorbeeld. Omdat de string *aabb* wel matcht met de expressie a^*b^*, maar niet met b^*a^*, zijn de expressies a^*b^* en b^*a^* niet equivalent.

Voorbeelden van expressies die wel equivalent zijn (wat we ook nemen voor r, s of t):

- $r + s$ en $s + r$,
- $r(s + t)$ en $rs + rt$,
- r^* en $\epsilon + rr^*$.

We bewijzen alleen het tweede voorbeeld. Dat wil zeggen: we beredeneren dat

$$L(r(s + t)) = L(rs + rt). \tag{2}$$

Stel eerst dat w een element is van $L(r(s+t))$; dat betekent dat w matcht met $r(s+t)$. Per definitie kan w dus worden gesplitst als $w = uv$ zó dat $u \rhd r$ en $v \rhd s + t$. Uit het laatste feit volgt dat $v \rhd s$ of $v \rhd t$. In het eerste geval mogen we concluderen dat $w \rhd rs$, in het tweede geval dat $w \rhd rt$; in beide gevallen volgt dat $w \rhd rs + rt$, zodat we bewezen hebben dat w een element is van $L(rs + rt)$.

De andere richting (aannemende dat $w \in L(rs + rt)$, laten zien dat w tot $L(r(s + t))$ behoort), laten we over aan de lezer.

Opgaven

Opgave 1.1 Onderzoek of de volgende woorden matchen met de expressie $(ab + aa + baa)^*$.

bbaaba, *abaabaaabaa*, *aaaabaaaa*, *baaaaabaa*, *baaaaabaaaab*.

Opgave 1.2 In UNIX kun je met het commando `ls` alle files in een directory bekijken. Maar het is ook mogelijk alleen maar bepaalde files te laten zien. Hierbij kun je gebruik van de volgende expressies maken:

$?$	$\sim$	een willekeurig element uit Σ,
$*$	$\sim$	een willekeurig element uit Σ^*,
$[n_1..n_2]$	$\sim$	een cijfer tussen n_1 en n_2 (n_1 en n_2 zelf meegerekend),
$[l_1..l_2]$	$\sim$	(een letter tussen l_1 en l_2 (l_1 en l_2 zelf meegerekend).

Het alfabet Σ is daarbij als volgt gedefineerd:

$$\Sigma := \{a, b, \ldots, z\} \cup \{A, B, \ldots, Z\} \cup \{0, \ldots, 9\} \cup \{., -\}$$

Schrijf de volgende UNIX-expressies als reguliere expressies:

(a) [2..6]?

(b) $*$

(c) $*[k..z]$

Opgave 1.3 Geef reguliere expressies voor de volgende talen in $\{a, b\}$:

(a) $\{w \mid |w| = 2\}$
(b) $\{w \mid |w| > 2\}$
(c) $\{awa \mid w \in \{a, b\}^*\}$
(d) $\{w \mid w \neq aba, w \neq bab\}$
(e) $\{w \mid \#_a(w)$ is een drievoud $\}$
(f) $\{w \mid \#_a(w)$ is geen drievoud $\}$
(g) $\{w \mid w$ eindigt niet op $ab\}$
(h) $\{w \mid w$ heeft aba niet als deelstring $\}$

Opgave 1.4 Welke talen worden door de volgende expressies weergegeven?

(a) $(1 + \epsilon)(00^*1)^*0^*$
(b) $(0^*1^*)^*000(0 + 1)^*$
(c) $(0 + 10)^*1^*$

Opgave 1.5 Gegeven is het alfabet $\Sigma = \{0, 1\}$. Welke talen worden weergegeven door de volgende expressies?

0, $\varnothing$, ϵ, 0^*, $\varnothing^*$, ϵ^*, $(0\varnothing)^*$.

Opgave 1.6 Gegeven is een alfabet Σ, en reguliere expressies r en s over Σ. Ga na of de volgende reguliere expressies paarsgewijs equivalent zijn. Verklaar uw antwoord!

(a) rr^* en r^*r
(b) $(r + s)^*$ en $r^* + s^*$
(c) $(r + s)^*$ en $(r^*s^*)^*$
(d) $(r + s)^*$ en $(rs^*)^*$
(e) de UNIX-expressies $[a..c + b..f]*$ en $[a..f]*$
(f) de UNIX-expressies $*$ en $*$?

Opgave 1.7 Gegeven is een eindige verzameling woorden E. Ontwerp een efficiënte algoritme om te bepalen of een invoerwoord al dan niet tot E^* behoort. (Met efficiënt wordt hier bedoeld dat de algoritme een rekentijd heeft die polynomiaal is in de lengte van de invoer.)