Formális nyelvek és fordítóprogramok

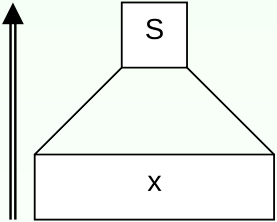
11–12. Alulról-felfelé haladó elemzés. $LR(1)$ és $LALR(1)$ elemzések

Bodó Zalán

Babeş–Bolyai Tudományegyetem
Matematika és Informatika Kar
Magyar Matematika és Informatika Intézet

$LR(k)$ elemzésben használatos fogalmak

$LR(k)$ és $LR(1)$ elemzések: **alulról-felfelé** haladó elemzés: a terminálisokból (a programból) kiindulva próbáljuk felépíteni a szintaxisfát, azaz eljutni az S szimbólumhoz **úgy**, hogy mindig megkeressük a mondatforma **nyelét** (= legbaloldalibb egyszerű részmondat) és azt helyettesítjük a szabály bal oldalával.



Def. Legjobboldalibb helyettesítés

Adott $\beta A x$ mondatforma, $\beta \in (N \cup T)^*$, $x \in T^*$, és $A \to \alpha$ szabály esetén a $\beta\alpha x$-t legjobboldalibb helyettesítésnek nevezzük.

Def. Legjobboldalibb levezetés

Ha egy levezetésben minden helyettesítés **legjobboldalibb**, akkor a levezetést **legjobboldalibb levezetésnek** nevezzük és az alábbi módon jelöljük:

$$S \underset{\text{legjobb}}{\overset{*}{\Rightarrow}} x$$

legjobboldalibb levezetés = alulról-felfelé haladó elemzés inverze

1. Példa

Tekintsük a következő grammatikát:

$$S \quad \to \quad S + S \mid S - S \mid (S) \mid 0 \mid 1$$

és a $w = 1 - (0 - 1)$ szimbólumsorozatot.
Adjuk meg w legjobboldalibb levezetését. A nyél segítségével ez visszafelé a következő módon írható fel:
$\underline{1} - (0 - 1) \Leftarrow S - (\underline{0} - 1) \Leftarrow S - (S - \underline{1}) \Leftarrow S - (\underline{S - S}) \Leftarrow S - \underline{(S)} \Leftarrow \underline{S - S} \Leftarrow S$
A nyelet aláhúzással jelöltük.

$LR(k)$ nyelvtanok

- $\#$ jelölje a szimbólumsorozat végét
- $S' \to S$ új szabály bevitele $\Rightarrow$ kiegészített nyelvtan:

$$G' = (N \cup \{S'\}, T, P \cup \{S' \to S\}, S')$$

Def. $LR(k)$ nyelvtan

Egy kiegészített nyelvtan $LR(k)$ nyelvtan ($k \geq 0$), ha

$$\begin{aligned} S' &\overset{*}{\Rightarrow} \alpha A w \Rightarrow \alpha\beta w \\ S' &\overset{*}{\Rightarrow} \gamma B x \Rightarrow \gamma\delta x = \alpha\beta y \end{aligned}$$

levezetésekre

$$FIRST_k(w) = FIRST_k(y)$$

esetén $\alpha = \gamma, \quad A = B, \quad x = y.$

Informálisan: $LR(k)$ nyelvtan, ha az $\alpha\beta w$ mondatformában a w első szimbólumától kezdve előreolvasha k db. szimbólumot, egyértelműen meghatározható, hogy β a nyél, vagyis a redukálást az $A \to \beta$ szabállyal kell végezni.

Def. Nyél (**ismétlés**)

Egy mondatforma legbaloldalibb egyszerű részmondatát a mondatforma **nyelének** nevezzük.

2. Példa

Tekintsük az alábbi G_1 grammatikát:

$$\begin{aligned} S' &\to S \\ S &\to aS \mid bA \mid b \\ A &\to bA \mid b \end{aligned}$$

$(L(G_1) = \{a^m b^n \mid m \geq 0, n \geq 1\})$
Ez $LR(1)$ grammatika; a nyél mindig a mondatforma végén fog megjelenni; egy szimbólum előreolvasásával eldönthető, hogy *léptetés* vagy *redukció* műveletet kell végrehajtani.
Vagyis, ha egy $a^i b^j$ mondatot elemzünk, akkor mindaddig *léptetnünk* kell, amíg az előreolvasási szimbólum $\#$ nem lesz.

3. Példa

Tekintsük az alábbi G_2 grammatikát:

$$\begin{aligned} S' &\rightarrow S \\ S &\rightarrow S - S \mid a \end{aligned}$$

$(L(G_2) = a - a - \ldots - a)$
Ez nem $LR(k)$ grammatika, mert pl. az $S - S - S$ kifejezést nem tudjuk egyértelműen redukálni. Ha az elemzéssel az $S - S. - S$ pontban tartunk, nem tudjuk eldönteni, hogy redukáljunk az $S \rightarrow S - S$ szabály szerint, vagy léptessünk.

Tétel

Minden $LR(k)$ nyelvtanhoz létezik vele ekvivalens $LR(1)$ nyelvtan.

- $\Rightarrow$ elegendő csak az $LR(1)$ nyelvekkel foglalkozni
- nem adunk algoritmust, amely tetszőleges $LR(k)$-t $LR(1)$-gyé alakít; nekünk kell $LR(1)$ grammatikát találnunk *adott problémára*
- adunk egy elemző módszert az $LR(1)$ grammatikákhoz; ezzel azt is el tudjuk hatékonyan dönteni, hogy adott grammatika $LR(1)$ grammatika-e (a definíció *nehezen* használható fel e célra)

$LR(1)$ kanonikus halmazok

Def. Járható prefix

Legyen az $\alpha\beta x$, $(\alpha, \beta \in (N \cup T)^*$, $x \in T^*)$ mondatforma nyele β. Ekkor az $\alpha\beta$ szimbólumsorozat prefixeit az $\alpha\beta x$ **járható prefix**einek nevezzük.

4. Példa

Tekintsük az alábbi grammatikát:

$$\begin{array}{rcl} S' & \to & S \\ S & \to & S+S \mid S-S \mid (S) \mid 0 \mid 1 \end{array}$$

Legyen továbbá $S-(0+1)$ egy mondatforma. A mondatforma nyele a 0, a járható prefixek pedig: ε, S, $S-$, $S-($, $S-(0$.

- nyél keresése = leghosszabb járható prefix keresése
- ez az alapötlete az $LR(1)$-elemzésnek
- a leghosszabb járható prefix megtalálásához egy automatát fogunk rendelni

Def. $LR(1)$-elem

Egy

$$[A \to \alpha.\beta, a]$$

elemet egy grammatika **$LR(1)$-elemének** nevezünk, ha $A \to \alpha.\beta$ a grammatika egy helyettesítési szabálya, $a \in T \cup \{\#\}$.
Az $A \to \alpha.\beta$ az $LR(1)$-elem magja, a pedig az $LR(1)$-elem előreolvasási szimbóluma.

- az elemzésben az előreolvasási szimbólumnak csak akkor lesz szerepe, ha az $LR(1)$-elem redukciót ír elő, azaz alakja $[A \to \alpha., a]$; ekkor redukciót csak abban az esetben lehet végrehajtani, ha az α-t, azaz a mondat nyelét a követi

Def.

Egy grammatika $[A \to \alpha.\beta, a]$ $LR(1)$-eleme **érvényes** a $\gamma\alpha$ járható prefixre nézve, ha

$$S' \overset{*}{\Rightarrow} \gamma A x \Rightarrow \gamma\alpha\beta x,$$

$\gamma \in (N \cup T)^*$, $x \in T^*$, és az a az x első szimbóluma, vagy $x = \varepsilon$ esetén $a = \#$.

5. Példa

Legyen a következő grammatika:

$$\begin{aligned} S' &\rightarrow S \\ S &\rightarrow aSb \mid A \\ A &\rightarrow aA \mid \varepsilon \end{aligned}$$

Tekintsük továbbá két mondatforma levezetését:

$$\begin{aligned} S' &\Rightarrow S \Rightarrow aSb \Rightarrow aaSbb \\ S' &\Rightarrow S \Rightarrow aAb \Rightarrow aaAb \end{aligned}$$

Az első esetben a mondatforma nyele aSb, ezért aa egy járható prefix; erre a járható prefixre nézve érvényes az $[S \rightarrow a.Sb, b]$ $LR(1)$-elem.

A második esetben a mondatforma nyele aA, ezért aa ez esetben is egy járható prefix; erre a járható prefixre nézve érvényes az $[A \rightarrow a.A, b]$ $LR(1)$-elem.

Fontos: Ha a $\gamma\alpha$ járható prefixre nézve az $[A \to \alpha.B\beta, a]$ egy érvényes $LR(1)$-elem, és $\exists\ B \to \delta$ helyettesítési szabály a grammatikában, akkor ugyanerre a járható prefixre nézve a $[B \to .\delta, b]$ is egy érvényes $LR(1)$-elem lesz, ahol $b \in FIRST(\beta a)$.

Meg kell határoznunk a grammatika $LR(1)$-elemhalmazát; ehhez definiáljuk a *closure* és *read* függvényeket:

ALG 1 *closure*(H)

1: a H halmaz minden eleme legyen eleme a *closure*(H) halmaznak is;
2: ha $[A \to \alpha.B\beta, a] \in closure(H)$ és $B \to \gamma$ a grammatika egy szabálya, akkor legyen $[B \to .\gamma, b] \in closure(H)$, $\forall b \in FIRST(\beta a)$-ra;
3: a *closure*(H) halmazt az előző lépéssel addig kell bővíteni, amíg az lehetséges.

ALG 2 *read*(H, X), $X \in (N \cup T)$

1: ha $[A \to \alpha.X\beta, a] \in H$, akkor a *closure*$([A \to \alpha X.\beta, a])$ minden eleme legyen a *read*(H, X) eleme;
2: a *read*(H, X) halmazt az előző lépéssel addig kell bővíteni, amíg az lehetséges.

Megjegyzések:

1. Ha H a γ járható prefixre érvényes $LR(1)$-eleme(ke)t tartalmaz, akkor a *closure*(H) kibővíti ezt az összes többi, ugyancsak a γ-ra érvényes $LR(1)$-elemmel.
2. Ha H a γ járható prefixre érvényes $LR(1)$-elemeket tartalmazza, akkor a *read*(H, X) a γX-re érvényes elemeket fogja tartalmazni.
3. A rövidebb jelölés érdekében, ha van két $LR(1)$-elem, melyek magjai azonosak és csak az előreolvasási szimbólumban különböznek, akkor azokat összevonjuk a következő módon:

$$[A \to \alpha.X\beta, a/b]$$

6. Példa

Tekintsük a következő grammatikát (5. Példa):

$$\begin{aligned} S' &\rightarrow S \\ S &\rightarrow aSb \mid A \\ A &\rightarrow aA \mid \varepsilon \end{aligned}$$

A grammatika egy $LR(1)$-eleme az $[S' \rightarrow .S, \#]$.
Ekkor legyen $H_0 = closure([S' \rightarrow .S, \#])$, vagyis

$$\begin{aligned} H_0 = & \{[S' \rightarrow .S, \#], [S \rightarrow .aSb, \#], [S \rightarrow .A, \#], [A \rightarrow .aA, \#], \\ & [A \rightarrow ., \#]\} \end{aligned}$$

Ezután számítsuk ki a $read(H_0, a)$ halmazt:

$$\begin{aligned} read(H_0, a) = & closure(\{[S \rightarrow a.Sb, \#], [A \rightarrow a.A, \#]\}) = \\ & \{[S \rightarrow a.Sb, \#], [S \rightarrow .aSb, b], [S \rightarrow .A, b], \\ & [A \rightarrow .aA, b/\#], [A \rightarrow ., b/\#], [A \rightarrow a.A, \#]\} \end{aligned}$$

- megépítjük egy grammatika $LR(1)$-elemeinek kanonikus halmazát: $H_0, H_1, \ldots, H_m$
- ezek fogják az automata állapotait jelölni
- az automata végállapotai a nyél megtalálását jelentik

ALG 3 $LR(1)$-elemek kanonikus halmazainak meghatározása

1: $H_0 = closure([S' \to .S, \#])$
2: Minden lehetséges X szimbólumra, $X \in (N \cup T)$, képezzük a $read(H_0, X)$ halmazt; ha ez nem üreshalmaz és nem egyezik meg egy előbbi halmazzal, akkor legyen ez egy új, eggyel nagyobb indexű kanonikus halmaz;
3: Ismételjük meg az előző műveletet az új kanonikus halmazokra; végezzük ezt addig, amíg a kanonikus halmazok halmaza tovább már nem bővíthető.

Az így létrehozott $H_0, H_1, \ldots, H_m$ halmazokat **a grammatika $LR(1)$-kanonikus halmazai**nak nevezzük.

7. Példa

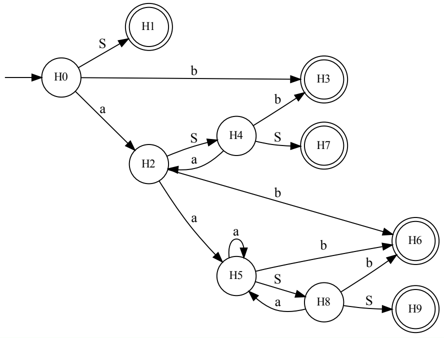
Tekintsük az alábbi grammatikát, és határozzuk meg annak $LR(1)$-kanonikus halmazait:

$$\begin{aligned} S' &\rightarrow S \\ S &\rightarrow aSS \mid b \end{aligned}$$

$$\begin{aligned}
H_0 &= closure([S' \rightarrow .S, \#]) = \{[S' \rightarrow .S, \#], [S \rightarrow .aSS, \#], [S \rightarrow .b, \#]\} \\
H_1 &= read(H_0, S) = closure([S' \rightarrow S., \#]) = \{[S' \rightarrow S., \#]\} \\
H_2 &= read(H_0, a) = closure([S \rightarrow a.SS, \#]) = \\
&= \{[S \rightarrow a.SS, \#], [S \rightarrow .aSS, a/b], [S \rightarrow .b, a/b]\} \\
H_3 &= read(H_0, b) = closure([S \rightarrow b., \#]) = \{[S \rightarrow b., \#]\} \\
H_4 &= read(H_2, S) = closure([S \rightarrow aS.S, \#]) = \\
&= \{[S \rightarrow aS.S, \#], [S \rightarrow .aSS, \#], [S \rightarrow .b, \#]\} \\
H_5 &= read(H_2, a) = closure([S \rightarrow a.SS, a/b]) = \\
&= \{[S \rightarrow a.SS, a/b], [S \rightarrow .aSS, a/b], [S \rightarrow .b, a/b]\} \\
H_6 &= read(H_2, b) = closure([S \rightarrow b., a/b]) = \{[S \rightarrow b., a/b]\} \\
H_7 &= read(H_4, S) = closure([S \rightarrow aSS., \#]) = \{[S \rightarrow aSS., \#]\} \\
& read(H_4, a) = closure([S \rightarrow a.SS, \#]) = H_2 \\
& read(H_4, b) = closure([S \rightarrow b., \#]) = H_3
\end{aligned}$$

$$
\begin{aligned}
H_8 \quad &= \quad read(H_5, S) = closure([S \to aS.S, a/b]) = \\
&= \quad \{[S \to aS.S, a/b], [S \to .aSS, a/b], [S \to .b, a/b]\} \\
&\quad\quad read(H_5, a) = closure([S \to a.SS, a/b]) = H_5 \\
&\quad\quad read(H_5, b) = closure([S \to b., a/b]) = H_6 \\
H_9 \quad &= \quad read(H_8, S) = closure([S \to aSS., a/b]) = \{[S \to aSS., a/b]\} \\
&\quad\quad read(H_8, a) = closure([S \to a.SS, a/b]) = H_5 \\
&\quad\quad read(H_8, b) = closure([S \to b., a/b]) = H_6
\end{aligned}
$$

Az eredő automata (a végállapotok a redukciót előíró kanonikus halmazok):



Az $LR(1)$ elemző

Tétel

Egy γ járható prefixre érvényes $LR(1)$-elemek halmaza az a H_k kanonikus elemhalmaz, amelyik az elemző véges determinisztikus automatájának ahhoz a H_k állapotához tartozik, amelyikbe az automata a kezdőállapotból a γ hatására kerül.

- $LR(1)$ **elemző táblázat** = a tételben szereplő DVA leírása
- si = léptetés az i. állapotba
- rj = redukálás a j. szabály alapján

Tétel

A G' kiegészített nyelvtan akkor és csak akkor $LR(1)$ grammatika, ha a nyelvtanhoz készített **kanonikus elemző táblázatok** kitöltése *konfliktusmentes*.

Konfliktus = kitöltéskor a táblázat egy mezőjébe egynél több (léptetés vagy redukció) művelet kerül.

Elemzés

- az elemző egy állapota:
  (elemzendő sorozat$\#$, verem tartalma)
- kezdőállapot: $(.w\#, 0)$
- ha az elemző aktuális állapota $(x.by\#, i_k i_{k-1} \dots i_0)$, akkor
  - ha a táblázat (i_k, b) mezője sℓ-t tartalmaz, akkor

    $$(x.by\#, i_k i_{k-1} \dots i_0) \Rightarrow (xb.y\#, \ell i_k i_{k-1} \dots i_0)$$

    azaz léptetés (*shift*) történik
  - ha a táblázat (i_k, b) mezője rℓ-t tartalmaz, akkor redukciót/redukálást végzünk az ℓ. szabály alapján:

    $$(x.by\#, i_k i_{k-1} \dots i_0) = (z\alpha.by\#, i_k i_{k-1} \dots i_j i_{j-1} \dots i_0) \Rightarrow$$

    $$\Rightarrow (z.Aby\#, i_j i_{j-1} \dots i_0)$$

    ahol a redukciót az $A \to \alpha$ szabály alapján végeztük, $|\alpha| = k - j$
  - különben: szintaktikai hiba

8. Példa

Tekintsük a 7. Példa grammatikáját és a már meghatározott $LR(1)$-kanonikus halmazokat:

$$0 : S' \to S$$
$$1 : S \to aSS$$
$$2 : S \to b$$

$$\begin{aligned}
H_0 &= \{[S' \to .S, \#], [S \to .aSS, \#], [S \to .b, \#]\} \\
H_1 &= \{[S' \to S., \#]\} \\
H_2 &= \{[S \to a.SS, \#], [S \to .aSS, a/b], [S \to .b, a/b]\} \\
H_3 &= \{[S \to b., \#]\} \\
H_4 &= \{[S \to aS.S, \#], [S \to .aSS, \#], [S \to .b, \#]\} \\
H_5 &= \{[S \to a.SS, a/b], [S \to .aSS, a/b], [S \to .b, a/b]\} \\
H_6 &= \{[S \to b., a/b]\} \\
H_7 &= \{[S \to aSS., \#]\} \\
H_8 &= \{[S \to aS.S, a/b], [S \to .aSS, a/b], [S \to .b, a/b]\} \\
H_9 &= \{[S \to aSS., a/b]\}
\end{aligned}$$

	a	b	S	$\#$
0	$s2$	$s3$	$s1$	
1				r0 / OK
2	$s5$	$s6$	$s4$	
3				$r2$
4	$s2$	$s3$	$s7$	
5	$s5$	$s6$	$s8$	
6	$r2$	$r2$		
7				$r1$
8	$s5$	$s6$	$s9$	
9	$r1$	$r1$		

Elemezzük az *aabbabb* szöveget.

$(.aabbabb\#, 0) \stackrel{s2}{\Rightarrow} (a.abbabb\#, 02) \stackrel{s5}{\Rightarrow} (aa.bbabb\#, 025) \stackrel{s6}{\Rightarrow}$
$(aab.babb\#, 0256) \stackrel{r2}{\Rightarrow} (aa.Sbabb\#, 025) \stackrel{s8}{\Rightarrow} (aaS.babb\#, 0258) \stackrel{s6}{\Rightarrow}$
$(aaSb.abb\#, 02586) \stackrel{r2}{\Rightarrow} (aaS.Sabb\#, 0258) \stackrel{s9}{\Rightarrow} (aaSS.abb\#, 02589) \stackrel{r1}{\Rightarrow}$
$(a.Sabb\#, 02) \stackrel{s4}{\Rightarrow} (aS.abb\#, 024) \stackrel{s2}{\Rightarrow} (aSa.bb\#, 0242) \stackrel{s6}{\Rightarrow}$
$(aSab.b\#, 02426) \stackrel{r2}{\Rightarrow} (aSa.Sb\#, 0242) \stackrel{s4}{\Rightarrow} (aSaS.b\#, 02424) \stackrel{s3}{\Rightarrow}$
$(aSaSb.\#, 024243) \stackrel{r2}{\Rightarrow} (aSaS.S\#, 02424) \stackrel{s7}{\Rightarrow} (aSaSS.\#, 024247) \stackrel{r1}{\Rightarrow}$
$(aS.S\#, 024) \stackrel{s7}{\Rightarrow} (aSS.\#, 0247) \stackrel{r1}{\Rightarrow} (.S\#, 0) \stackrel{s1}{\Rightarrow} (S.\#, 01) \stackrel{r0}{\Rightarrow}$
OK

- **léptetés/léptetés** konfliktus nem lehetséges
- **léptetés/redukció** konfliktus megtörténhet, ha

$$read(H_i, X) = H_j$$
$$\text{és}$$
$$[A \to \alpha., X] \in H_i$$

(lásd a slide-sorozat végén levő 10-es példát)

- **redukció/redukció** konfliktus is történhet, ha

$$[A \to \alpha., X] \in H_i$$
$$\text{és}$$
$$[B \to \beta., X] \in H_i$$

(lásd az alábbi grammatikát:

$$\begin{array}{rcl} S' & \to & S \\ S & \to & a \mid aS \mid \varepsilon \end{array}$$

)

Az $LALR(1)$ elemző

- cél: az elemző állapotainak csökkentése, amivel természetesen növekszik az elemző hatékonysága
- ezt úgy kell megoldani, hogy az elemezhető nyelvek halmazának mérete *lényegesen ne csökkenjen*
- **észrevétel**: az $LR(1)$-kanonikus halmazok között vannak olyanok, ahol a magok megegyeznek, csak az előreolvasási szimbólum különbözik $\rightarrow$ egyesítsük ezeket a halmazokat:

  ha H_i és H_j egyesíthetők, akkor legyen $K_{i,j} = H_i \cup H_j$

- az így kapott halmazokat **egyesített** $LR(1)$ **kanonikus halmazok**nak nevezzük
- az elemzőt $LALR(1)$ **elemző**nek nevezzük

Konfliktusok

- a *read* függvény nem okoz problémát, mert az csak a magoktól függ, vagyis nem függ az előreolvasási szimbólumtól; azaz ha

$$K = H_1 \cup H_2 \cup \ldots \cup H_k$$

és

$$read(H_1, X) = H_1', read(H_2, X) = H_2', \ldots, read(H_k, X) = H_k'$$

és

$$K' = H_1' \cup H_2' \cup \ldots \cup H_k',$$

akkor

$$read(K, X) = K'$$

- problémák származhatnak viszont bizonyos $LR(1)$-elemek egy halmazba kerüléséből:
  1. **léptetés-léptetés konfliktusok** nem jelenhetnek meg, mivel ha
     $$[A \to \alpha.a\beta, b] \in H_i \text{ és } [B \to \gamma.a\delta, c] \in H_j,$$
     akkor az összevonás után az a szimbólumra továbbra is léptetést írunk elő
  2. **léptetés-redukálás konfliktusok** sem jelenhetnek meg, mivel ez abban az esetben történhetne meg, ha
     $$[A \to \alpha.a\beta, b] \in H_i \text{ és } [B \to \gamma., a] \in H_j,$$
     de ez nem történhet meg, mivel ez azt jelentené, hogy mindkét halmazban szerepelnie kell ugyanilyen maggal rendelkező $LR(1)$-elemnek, azaz H_j-ben lennie kell egy $[A \to \alpha.a\beta, c]$ elemnek, ami *nem lehetséges*, mert akkor nem lenne a grammatikánk $LR(1)$-es
  3. **redukálás-redukálás konfliktusok** felléphetnek; pl.:
     $$\begin{aligned} H_i &= \{[A \to c., d], [B \to c., e]\} \\ H_j &= \{[A \to c., e], [B \to c., d]\} \end{aligned}$$
     ekkor ha a bemenő szimbólum d vagy e akkor a mondat c nyele megtalálható, viszont nem eldönthető, hogy az $A \to c$ vagy a $B \to c$ szerinti reduckiót kell elvégezni

9. Példa

Tekintsük a 8. példában szereplő $LR(1)$ kanonikus halmazokat. Észrevehető, hogy a következő halmazok egyesíthetők:

$$H_2 \cup H_5$$
$$H_3 \cup H_6$$
$$H_4 \cup H_8$$
$$H_7 \cup H_9$$

A fennmaradó H_0 és H_1 halmazokat változatlanul hagyjuk.

$$
\begin{aligned}
H_0 &= \{[S' \to .S, \#], [S \to .aSS, \#], [S \to .b, \#]\} \\
H_1 &= \{[S' \to S., \#]\} \\
H_{2,5} &= \{[S \to a.SS, \#/a/b], [S \to .aSS, a/b], [S \to .b, a/b]\} \\
H_{3,6} &= \{[S \to b., \#/a/b]\} \\
H_{4,8} &= \{[S \to aS.S, \#/a/b], [S \to .aSS, \#/a/b], [S \to .b, \#/a/b]\} \\
H_{7,9} &= \{[S \to aSS., \#/a/b]\}
\end{aligned}
$$

$$read(H_0, S) = H_1, \quad read(H_0, a) = H_{2,5}$$
$$read(H_0, b) = H_{3,6}, \quad read(H_{2,5}, S) = H_{4,8}$$
$$read(H_{2,5}, a) = H_{2,5}, \quad read(H_{2,5}, b) = H_{3,6}$$
$$read(H_{4,8}, S) = H_{7,9}, \quad read(H_{4,8}, a) = H_{2,5}$$
$$read(H_{4,8}, b) = H_{3,6}$$

	a	b	S	#
0	$s2,5$	$s3,6$	$s1$	
1				$r0/OK$
2, 5	$s2,5$	$s3,6$	$s4,8$	
3, 6	$r2$	$r2$		$r2$
4, 8	$s2,5$	$s3,6$	$s7,9$	
7, 9	$r1$	$r1$		$r1$

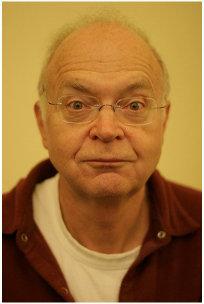
Elemezzük az *aabbabb* szöveget.

10. Példa

$$
\begin{aligned}
0 &: S' \to S \\
1 &: S \to iSeS \\
2 &: S \to iS \\
3 &: S \to a
\end{aligned}
$$

Írjuk fel az elemző táblázatot, vizsgáljuk meg, hol és miért következik be konfliktus, majd oldjuk azt fel a hagyományos módon.

Donald Knuth



Donald Ervin Knuth (1938–), amerikai informatikus, a híres „A számítógép-programozás művészete" (*The Art of Computer Programming*, TAOCP) mű szerzője. A TEX, a METAFONT megalkotója. Ő definiálta az $LR(k)$ grammatikát 1965-ben (*On the Translation of Languages from Left to Right*).

Leghíresebb művei:
Donald E. Knuth. *The Art of Computer Programming*, Volumes 1–4, Addison-Wesley Professional
Donald E. Knuth. *The TEXbook*, Reading, Massachusetts, Addison-Wesley, 1984.