Fundamentele Informatica 3

voorjaar 2020

`http://www.liacs.leidenuniv.nl/~vlietrvan1/fi3/`

Rudy van Vliet

kamer 140 Snellius, tel. 071-527 2876

rvvliet(at)liacs(dot)nl

college 5, 4 maart 2020

8. Recursively Enumerable Languages

8.3. More General Grammars

Huiswerkopgave

Voor 0.4pt

Inleveren: donderdag 19 maart 2020, 16:15 uur
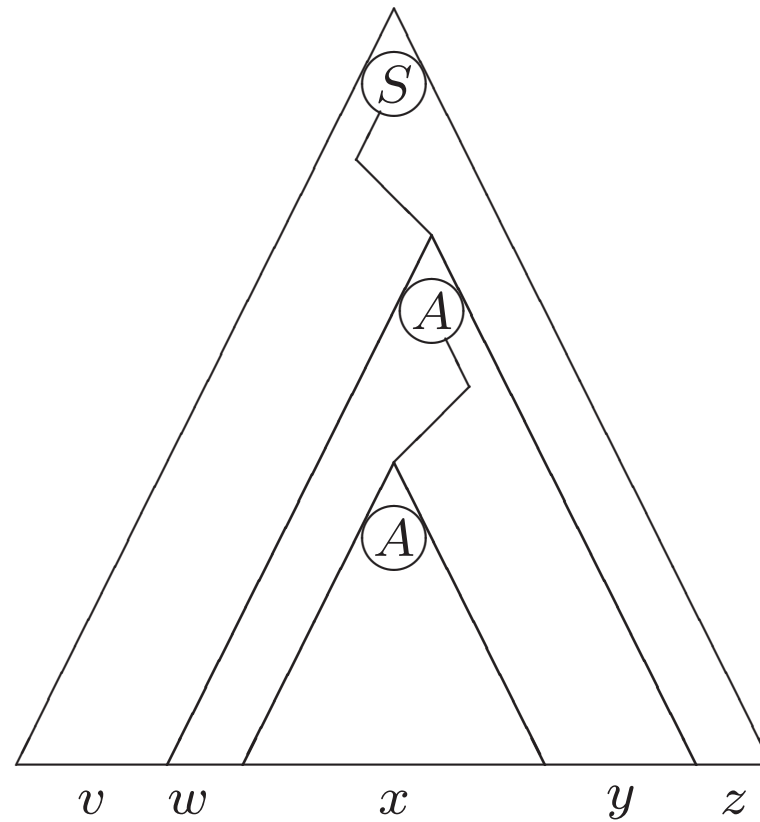
Laatste hoorcollege, 18 maart 2020

Betabanenmarkt = Science Career Event...

8.3. More General Grammars

reg. languages	FA	reg. grammar	reg. expression
determ. cf. languages	DPDA		
cf. languages	PDA	cf. grammar	
cs. languages	LBA	cs. grammar	
re. languages	TM	unrestr. grammar	

FI2: Pumping Lemma for CFLs

FI2: Pumping Lemma for CFLs
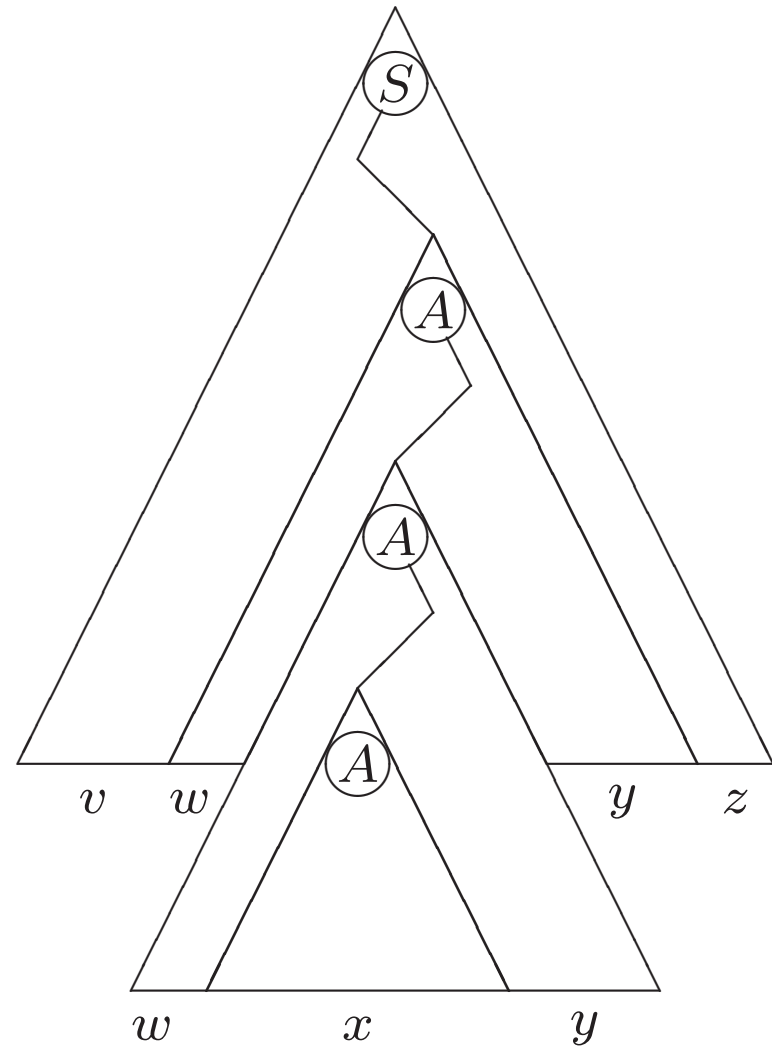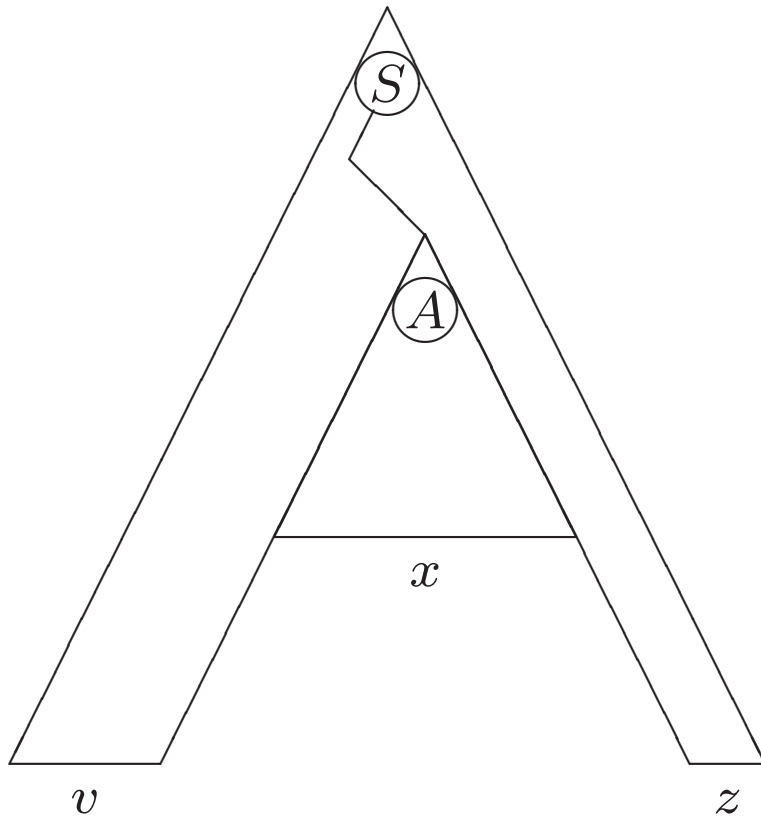
Definition 8.10. Unrestricted grammars

An unrestricted grammar is a 4-tuple $G = (V, \Sigma, S, P)$, where V and Σ are disjoint sets of variables and terminals, respectively, S is an element of V called the start symbol, and P is a set of productions of the form

$$\alpha \to \beta$$

where $\alpha, \beta \in (V \cup \Sigma)^*$ and α contains at least one variable.

Notation as for CFGs:

$$\alpha \Rightarrow_G^* \beta$$

$$L(G) = \{x \in \Sigma^* \mid S \Rightarrow_G^* x\}$$

but. . .

Example 8.12. A Grammar Generating $\{a^n b^n c^n \mid n \geq 1\}$

Example 8.12. A Grammar Generating $\{a^n b^n c^n \mid n \geq 1\}$

$$S \rightarrow SABC \mid LABC$$

$$BA \rightarrow AB \quad CB \rightarrow BC \quad CA \rightarrow AC$$

$$LA \rightarrow a \quad aA \rightarrow aa \quad aB \rightarrow ab \quad bB \rightarrow bb \quad bC \rightarrow bc \quad cC \rightarrow cc$$

Example 8.11. A Grammar Generating $\{a^{2^k} \mid k \in \mathbb{N}\}$

$$\{a, a^2, a^4, a^8, a^{16}, \ldots\} = \{a, aa, aaaa, aaaaaaaa, aaaaaaaaaaaaaaaa, \ldots\}$$

Example 8.11. A Grammar Generating $\{a^{2^k} \mid k \in \mathbb{N}\}$

$$\{a, a^2, a^4, a^8, a^{16}, \ldots\} = \{a, aa, aaaa, aaaaaaaa, aaaaaaaaaaaaaaaa, \ldots\}$$

$$S \to LaR$$

$$L \to LD \quad Da \to aaD \quad DR \to R$$

$$L \to \wedge \quad R \to \wedge$$

Example.

An Unrestricted Grammar Generating $XX = \{xx \mid x \in \{a, b\}^*\}$

First a CFG for $Pal = \{x \in \{a, b\}^* \mid x = x^r\}$:

$$S \rightarrow aSa \quad | \quad bSb \quad | \quad a \quad | \quad b \quad | \quad \Lambda$$

Example.

An Unrestricted Grammar Generating $XX = \{xx \mid x \in \{a, b\}^*\}$

$$S \rightarrow aAS \mid bBS \mid M$$

$$Aa \rightarrow aA \quad Ab \rightarrow bA \quad Ba \rightarrow aB \quad Bb \rightarrow bB$$

$$AM \rightarrow Ma \quad BM \rightarrow Mb \quad M \rightarrow \Lambda$$

Theorem 8.13.

For every unrestricted grammar G, there is a Turing machine T with $L(T) = L(G)$.

Proof.

1. Move past input
2. Simulate derivation in G on the tape of a Turing machine
3. Equal

Theorem 8.13.

For every unrestricted grammar G, there is a Turing machine T with $L(T) = L(G)$.

Proof.

1. Move past input
2. Simulate derivation in G on the tape of a Turing machine:

   Write S on tape

   Repeat

   a. Select production $\alpha \to \beta$

   b. Select occurrence of α (if there is one)

   c. Replace occurrence of α by β

   until b. fails (caused by . . .)
3. Equal

Theorem 7.31.

For every nondeterministic TM $T = (Q, \Sigma, \Gamma, q_0, \delta)$,
there is an ordinary (deterministic) TM $T_1 = (Q_1, \Sigma, \Gamma_1, q_1, \delta_1)$
with $L(T_1) = L(T)$.

Moreover, if there is no input on which T can loop forever,
then T_1 also halts on every input.

The proof of this result does not have to be known for the exam.

17

Example.

(The second part of) the construction from Theorem 8.13 to obtain a TM simulating a derivation in the unrestricted grammar with productions

$$S \rightarrow aBS \mid \wedge \quad aB \rightarrow Ba \quad Ba \rightarrow aB \quad B \rightarrow b$$

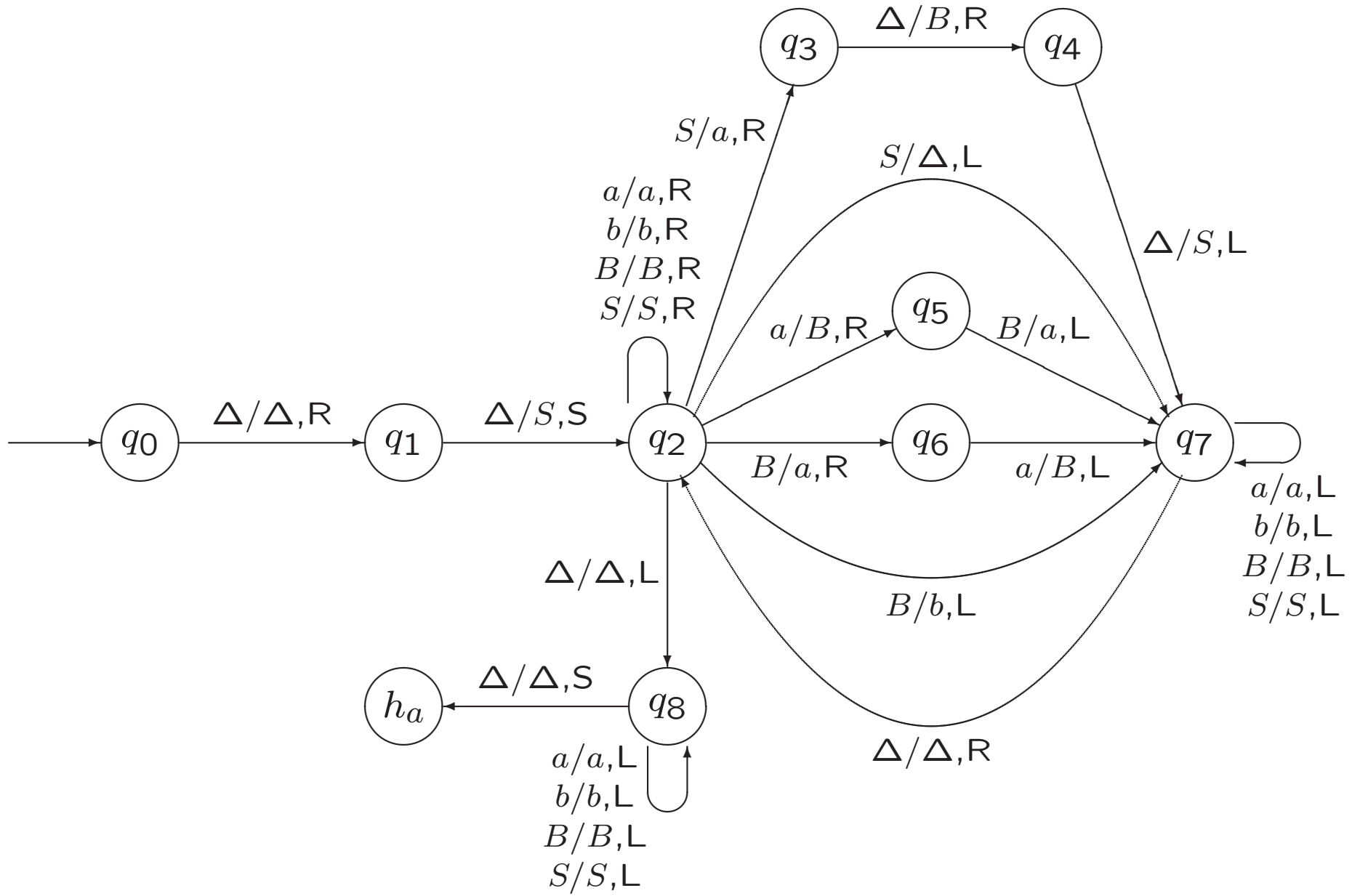See next slide

N.B.:

In next slide, we simulate application of arbitrary production by
• first moving to arbitrary position in current string (at q_2)
• only then selecting (and applying) a possible production

This implementation of the construction must be known for the exam

18

Theorem 8.14.

For every Turing machine T with input alphabet Σ,
there is an unrestricted grammar G
generating the language $L(T) \subseteq \Sigma^*$.

Proof.

1. Generate (every possible) input string for T.
2. Simulate computation of T for this input string as derivation in grammar.
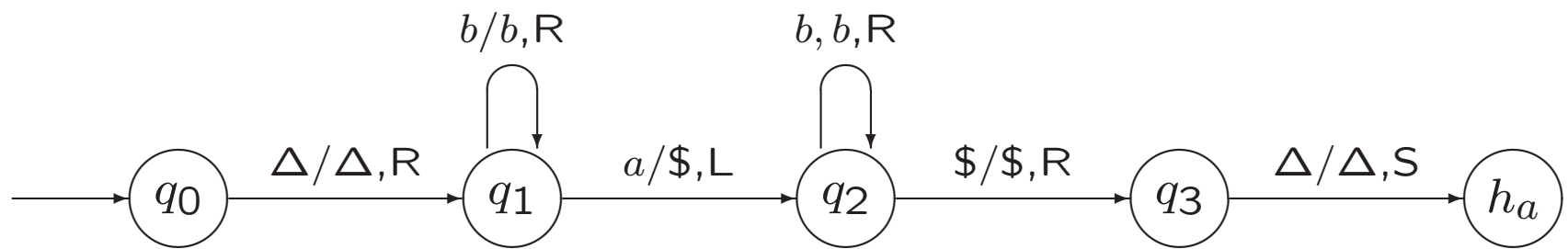3. If T reaches accept state, reconstruct original input string.

Notation:

description of tape contents: $x\underline{\sigma}y$ or $x\underline{y}$

configuration $xqy = xqy\Delta = xqy\Delta\Delta$

initial configuration corresponding to input x: $q_0\Delta x$

States: $q_0 \xrightarrow{\Delta/\Delta,\mathsf{R}} q_1$ with self-loop $b/b,\mathsf{R}$; $q_1 \xrightarrow{a/\$,\mathsf{L}} q_2$ with self-loop b,b,R; $q_2 \xrightarrow{\$/\$,\mathsf{R}} q_3 \xrightarrow{\Delta/\Delta,\mathsf{S}} h_a$
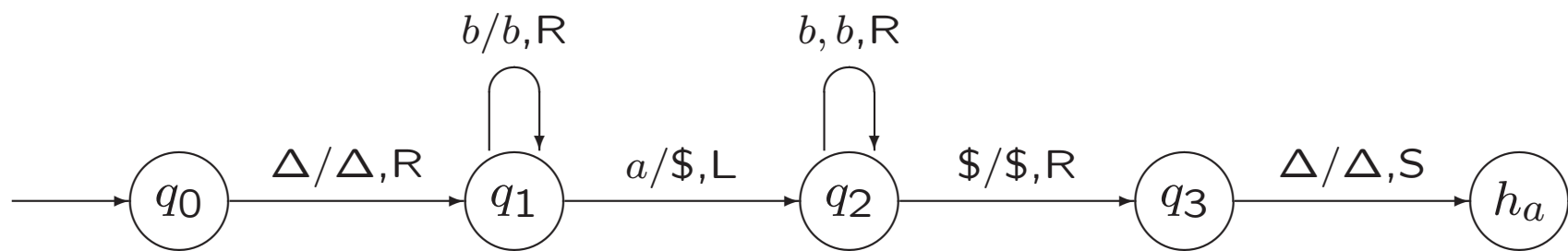
Computation for $x = ba$.

Theorem 8.14.

For every Turing machine T with input alphabet Σ,
there is an unrestricted grammar G
generating the language $L(T) \subseteq \Sigma^*$.

Proof.

1. Generate (every possible) input string for T (two copies),
with additional ($\triangle\triangle$)'s and state.
2. Simulate computation of T for this input string as derivation
in grammar (on second copy).
3. If T reaches accept state, reconstruct original input string.

q_0 $\xrightarrow{\Delta/\Delta,\mathsf{R}}$ q_1 $\xrightarrow{a/\$,\mathsf{L}}$ q_2 $\xrightarrow{\$/\$,\mathsf{R}}$ q_3 $\xrightarrow{\Delta/\Delta,\mathsf{S}}$ h_a

q_1: $b/b,\mathsf{R}$

q_2: b,b,R

3. If T reaches accept state, reconstruct original input string. . .

Theorem 8.14.

For every Turing machine T with input alphabet Σ,
there is an unrestricted grammar G
generating the language $L(T) \subseteq \Sigma^*$.

Proof.

1. Generate (every possible) input string for T (two copies),
with additional ($\triangle\triangle$)'s and state.

2. Simulate computation of T for this input string as derivation
in grammar (on second copy).

3. If T reaches accept state, reconstruct original input string.

Ad 2. Move $\delta(p, a) = (q, b, R)$ of T
      yields production $p(\sigma_1 a) \to (\sigma_1 b)q$

Ad 3. Propagate h_a all over the string
      $h_a(\sigma_1 \sigma_2) \to \sigma_1$, for $\sigma_1 \in \Sigma$
      $h_a(\triangle \sigma_2) \to \Lambda$

Theorem 8.14.
For every Turing machine T with input alphabet Σ,
there is an unrestricted grammar G
generating the language $L(T) \subseteq \Sigma^*$.

Proof.
1. Generate (every possible) input string for T (two copies),
with additional $(\triangle\triangle)$'s and state.
2. Simulate computation of T for this input string as derivation
in grammar (on second copy).
3. If T reaches accept state, reconstruct original input string.

Ad 2. Move $\delta(p, a) = (q, b, R)$ of T
     yields production $p(\sigma_1 a) \to (\sigma_1 b) q$
Ad 3. Propagate h_a all over the string (too few / many h_a's. . .)
     $h_a(\sigma_1 \sigma_2) \to \sigma_1$, for $\sigma_1 \in \Sigma$
     $h_a(\triangle \sigma_2) \to \wedge$