The Chomsky-Schützenberger Theorem for Weighted Automata with Storage

Luisa Herrmann Heiko Vogler

Institute of Theoretical Computer Science

Faculty of Computer Science

Technische Universität Dresden

September 2, 2015

Theorem (CS-Theorem) [Chomsky, Schützenberger 63]

$$L(G) = h(D_n \cap R)$$

G context-free grammar

h homomorphism

D_n n-Dyck language

R regular language

Theorem (CS-Theorem) [Chomsky, Schützenberger 63]

$$L(G) = h(D_n \cap R)$$

Alternative CS-Theorem [Harrison 78]

$$L(G) = h(g^{-1}(D_2) \cap R)$$

Theorem (CS-Theorem) [Chomsky, Schützenberger 63]

$$L(G) = h(D_n \cap R)$$

Alternative CS-Theorem [Harrison 78]

$$L(G) = h(g^{-1}(D_2) \cap R)$$

[Weir 88]:

CS-Theorem for string languages generated by tree-adjoining grammars

Theorem (CS-Theorem) [Chomsky, Schützenberger 63]

$$L(G) = h(D_n \cap R)$$

Alternative CS-Theorem [Harrison 78]

$$L(G) = h(g^{-1}(D_2) \cap R)$$

[Weir 88]:

CS-Theorem for string languages generated by tree-adjoining grammars

[Yoshinaka, Kaji, Seki 10]:

CS-Theorem for multiple context-free languages

Theorem (CS-Theorem) [Chomsky, Schützenberger 63]

$$L(G) = h(D_n \cap R)$$

Alternative CS-Theorem [Harrison 78]

$$L(G) = h(g^{-1}(D_2) \cap R)$$

[Weir 88]:

CS-Theorem for string languages generated by tree-adjoining grammars

[Yoshinaka, Kaji, Seki 10]:

CS-Theorem for multiple context-free languages

[Fratani, Voundy 15]:

CS-Theorem for indexed languages

weighted language: $L\colon \Sigma^* \to K$ for some weight algebra K
(power series)

weighted language: $L\colon \Sigma^* \to K$ for some weight algebra K
(power series)

[Salomaa, Soittola 78]:

CS-Theorem for semiring weighted CF languages

weighted language: $L\colon \Sigma^* \to K$ for some weight algebra K
(power series)

[Salomaa, Soittola 78]:

CS-Theorem for semiring weighted CF languages

[Droste, Vogler 13]:

CS-Theorem for unital valuation monoid weighted CF languages

weighted language: $L\colon \Sigma^* \to K$ for some weight algebra K
(power series)

[Salomaa, Soittola 78]:

CS-Theorem for semiring weighted CF languages

[Droste, Vogler 13]:

CS-Theorem for unital valuation monoid weighted CF languages

[Denkinger 15]:

CS-Theorem for strong bimonoid weighted multiple CF languages

weighted language: $L \colon \Sigma^* \to K$ for some weight algebra K
(power series)

[Salomaa, Soittola 78]:

CS-Theorem for semiring weighted CF languages

[Droste, Vogler 13]:

CS-Theorem for unital valuation monoid weighted CF languages

[Denkinger 15]:

CS-Theorem for strong bimonoid weighted multiple CF languages

our work:

CS-Theorem for weighted iterated pushdown languages

weighted language: $L\colon \Sigma^* \to K$ for some weight algebra K
(power series)

[Salomaa, Soittola 78]:

CS-Theorem for semiring weighted CF languages

[Droste, Vogler 13]:

CS-Theorem for unital valuation monoid weighted CF languages

[Denkinger 15]:

CS-Theorem for strong bimonoid weighted multiple CF languages

our work:

CS-Theorem for weighted iterated pushdown languages

CS-Theorem for weighted automata with storage

Table of Contents

Unital Valuation Monoid

$(K, +, \mathrm{val}, 0, 1)$ [Droste, Vogler 13]

- $(K, +, 0)$ commutative monoid
- $\mathrm{val}\colon K^* \to K$
  - $\mathrm{val}(a) = a$
  - $\mathrm{val}(\dots 0 \dots) = 0$
  - $\mathrm{val}(\dots 1 \dots) = \mathrm{val}(\dots\dots)$ ignore “1”s
  - $\mathrm{val}(\varepsilon) = 1$

Unital Valuation Monoid

$(K, +, \mathrm{val}, 0, 1)$ [Droste, Vogler 13]

- $(K, +, 0)$ commutative monoid
- $\mathrm{val}\colon K^* \to K$
  - $\mathrm{val}(a) = a$
  - $\mathrm{val}(\ldots 0 \ldots) = 0$
  - $\mathrm{val}(\ldots 1 \ldots) = \mathrm{val}(\ldots\ldots)$ ignore “1”s
  - $\mathrm{val}(\varepsilon) = 1$

Examples

1) average

Unital Valuation Monoid

$(K, +, \mathrm{val}, 0, 1)$ [Droste, Vogler 13]

- $(K, +, 0)$ commutative monoid
- $\mathrm{val}\colon K^* \to K$
  - $\mathrm{val}(a) = a$
  - $\mathrm{val}(\ldots 0 \ldots) = 0$
  - $\mathrm{val}(\ldots 1 \ldots) = \mathrm{val}(\ldots\ldots)$ ignore "1"s
  - $\mathrm{val}(\varepsilon) = 1$

Examples

1) average

2) strong bimonoid $(K, +, \cdot, 0, 1)$
   - $(K, +, 0)$ commutative monoid
   - $(K, \cdot, 1)$ monoid
   - 0 absorbing for $\cdot$

   e.g., semirings, bounded lattices

Storage [Scott 67]

Storage

[Scott 67]

storage type $S = (C, P, F, C_0)$

- C set (configurations)
- P set of functions $p \colon C \to \{\text{true}, \text{false}\}$ (predicates)
- F set of partial functions $f \colon C \to C$ (instructions)
- C_0 subset of C (initial configurations)

Examples $S = (C, P, F, C_0)$

- TRIV $= (\{c\}, \{p_{\mathrm{true}}\}, \{f_{\mathrm{id}}\}, \{c\})$

Examples $S = (C, P, F, C_0)$

- TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

$$p_{\text{true}}(c) = \text{true}$$

$$f_{\text{id}}(c) = c$$

Examples $S = (C, P, F, C_0)$

- TRIV = $(\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\mathrm{P}(S) = (C', P', F', C_0')$ by

Examples $S = (C, P, F, C_0)$

- TRIV = $(\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\mathrm{P}(S) = (C', P', F', C_0')$ by

  - $C' = (\Gamma \times C)^+$, $C_0' = (\Gamma \times C_0)$ (no empty pushdown)

Examples $S = (C, P, F, C_0)$

- TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$
- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\quad \mathrm{P}(S) = (C', P', F', C_0')$ by

  - $C' = (\Gamma \times C)^+, \;\; C_0' = (\Gamma \times C_0)$ (no empty pushdown)
  - $P' = \{\text{ bottom }\} \cup \{\, (\text{top} = \gamma) \mid \gamma \in \Gamma\} \cup \{\, \text{test}(p) \mid p \in P\}$

Examples $\quad S = (C, P, F, C_0)$

- TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\quad \mathrm{P}(S) = (C', P', F', C'_0)$ $\quad$ by

  - $C' = (\Gamma \times C)^+, \;\; C'_0 = (\Gamma \times C_0)$ (no empty pushdown)
  - $P' = \{\, \text{bottom} \,\} \cup \{\, (\text{top} = \gamma) \;\mid \gamma \in \Gamma\} \cup \{\, \text{test}(p) \;\mid p \in P\}$

$$\text{bottom}\left(\boxed{\delta}\boxed{c} \right) = \text{true}$$

Examples $\quad S = (C, P, F, C_0)$

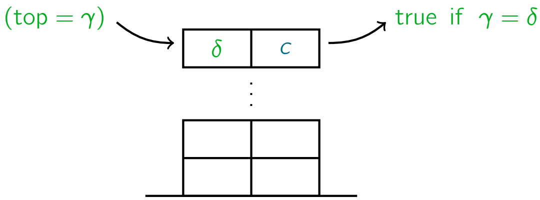
- TRIV = $(\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\quad \mathrm{P}(S) = (C', P', F', C_0')$ $\quad$ by

  - $C' = (\Gamma \times C)^+, \quad C_0' = (\Gamma \times C_0)$ (no empty pushdown)

  - $P' = \{ \text{ bottom } \} \cup \{ (\text{top} = \gamma) \mid \gamma \in \Gamma \} \cup \{ \text{ test}(p) \mid p \in P \}$

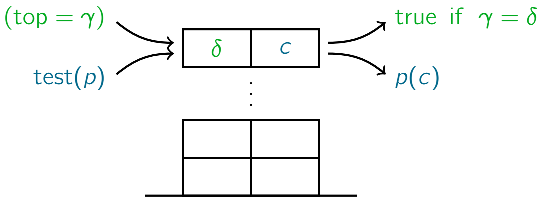


Examples $S = (C, P, F, C_0)$

- TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$
- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\mathrm{P}(S) = (C', P', F', C'_0)$ by

  - $C' = (\Gamma \times C)^+$, $C'_0 = (\Gamma \times C_0)$ (no empty pushdown)
  - $P' = \{\text{ bottom }\} \cup \{ (\text{top} = \gamma) \mid \gamma \in \Gamma\} \cup \{ \text{test}(p) \mid p \in P\}$



Examples $S = (C, P, F, C_0)$

- TRIV = $(\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\mathrm{P}(S) = (C', P', F', C'_0)$ by

  - $C' = (\Gamma \times C)^+$, $C'_0 = (\Gamma \times C_0)$ (no empty pushdown)

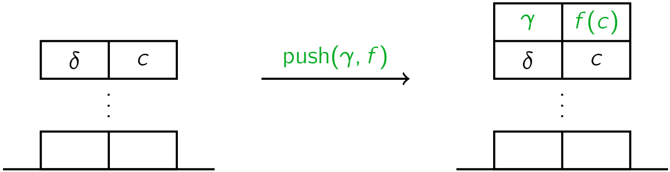
  - $P' = \{\ \text{bottom}\ \} \cup \{\ (\text{top} = \gamma)\ \mid \gamma \in \Gamma\} \cup \{\ \text{test}(p)\ \mid p \in P\}$

  - $F' = \{\ \text{push}(\gamma, f)\ \mid \gamma \in \Gamma, f \in F\} \cup \{\ \text{stay}(\gamma)\ \mid \gamma \in \Gamma\} \cup \{\ \text{pop}\ \}$

Examples $S = (C, P, F, C_0)$

- TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$
- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

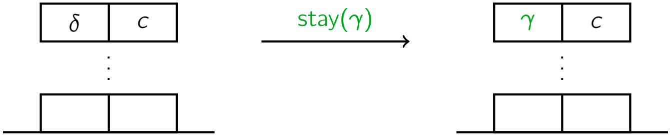
  define pushdown of S $\quad \mathrm{P}(S) = (C', P', F', C'_0)$ by
  - $C' = (\Gamma \times C)^+, \quad C'_0 = (\Gamma \times C_0)$ (no empty pushdown)
  - $P' = \{\text{ bottom }\} \cup \{\ (\text{top} = \gamma)\ \mid \gamma \in \Gamma\} \cup \{\text{ test}(p)\ \mid p \in P\}$
  - $F' = \{\ \text{push}(\gamma, f)\ \mid \gamma \in \Gamma, f \in F\} \cup \{\text{ stay}(\gamma)\ \mid \gamma \in \Gamma\} \cup \{\text{ pop }\}$



Examples $S = (C, P, F, C_0)$

- $\text{TRIV} = (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$
- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\quad \mathrm{P}(S) = (C', P', F', C_0')$ by
  - $C' = (\Gamma \times C)^+, \quad C_0' = (\Gamma \times C_0)$ (no empty pushdown)
  - $P' = \{\text{ bottom }\} \cup \{ \ (\text{top} = \gamma) \ \mid \gamma \in \Gamma\} \cup \{ \text{ test}(p) \ \mid p \in P\}$
  - $F' = \{ \text{ push}(\gamma, f) \ \mid \gamma \in \Gamma, f \in F\} \cup \{ \text{stay}(\gamma) \mid \gamma \in \Gamma\} \cup \{ \text{ pop } \}$



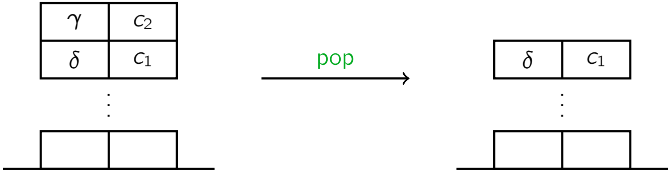
Examples $S = (C, P, F, C_0)$

- TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

- let $S = (C, P, F, C_0)$ storage type [Engelfriet 86/14]

  define pushdown of S $\quad \mathrm{P}(S) = (C', P', F', C_0')$ $\quad$ by

  - $C' = (\Gamma \times C)^+, \quad C_0' = (\Gamma \times C_0)$ (no empty pushdown)
  - $P' = \{\ \text{bottom}\ \} \cup \{\ (\text{top} = \gamma) \mid \gamma \in \Gamma\} \cup \{\ \text{test}(p) \mid p \in P\}$
  - $F' = \{\ \text{push}(\gamma, f) \mid \gamma \in \Gamma, f \in F\} \cup \{\ \text{stay}(\gamma) \mid \gamma \in \Gamma\} \cup \{\text{pop}\}$



Examples $S = (C, P, F, C_0)$

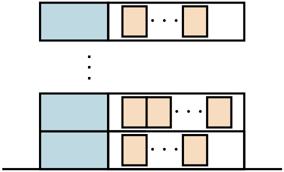
- TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$
- $\mathrm{P}(S) = (C', P', F', I', E')$ for given S [Engelfriet 86/14]

Examples $S = (C, P, F, C_0)$

- TRIV = $(\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$
- $\mathrm{P}(S) = (C', P', F', I', E')$ for given S [Engelfriet 86/14]
- iterated pushdown storage [Engelfriet 86/14; Engelfriet, Vogler 86]
  - $\mathrm{P}^0 = \mathrm{TRIV}$
  - $\mathrm{P}^{n+1} = \mathrm{P}(\mathrm{P}^n)$

Examples $S = (C, P, F, C_0)$

- TRIV = $(\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$
- $\mathrm{P}(S) = (C', P', F', I', E')$ for given S [Engelfriet 86/14]
- iterated pushdown storage [Engelfriet 86/14; Engelfriet, Vogler 86]
  - $\mathrm{P}^0 = \mathrm{TRIV}$
  - $\mathrm{P}^{n+1} = \mathrm{P}(\mathrm{P}^n)$



Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ $\quad (S, \Sigma, K)$-automaton

Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

- Q finite set (states), $Q_f \subseteq Q$ (final states), $q_0 \in Q$ (initial state)

Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

- Q finite set (states), $Q_f \subseteq Q$ (final states), $q_0 \in Q$ (initial state)
- $c_0 \in C_0$

Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

- Q finite set (states), $Q_f \subseteq Q$ (final states), $q_0 \in Q$ (initial state)
- $c_0 \in C_0$
- T finite set (transitions)

$$\tau = (q_1, x, p, q_2, f)$$

$q_1, q_2 \in Q, \quad x \in \Sigma \cup \{\varepsilon\}, \quad p \in P, \quad f \in F$

Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

- Q finite set (states), $Q_f \subseteq Q$ (final states), $q_0 \in Q$ (initial state)
- $c_0 \in C_0$
- T finite set (transitions)

$$\tau = (q_1, x, p, q_2, f)$$

$q_1, q_2 \in Q, \quad x \in \Sigma \cup \{\varepsilon\}, \quad p \in P, \quad f \in F$

- $\mathrm{wt}\colon T \to K$ (weight assignment)

Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

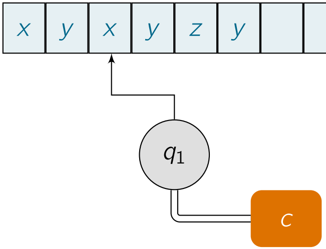
$$\tau = (q_1, x, p, q_2, f)$$

Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

$$\tau = (q_1, x, p, q_2, f)$$

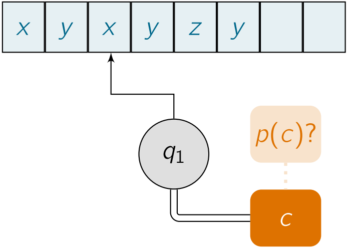


Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

$$\tau = (q_1, x, p, q_2, f)$$

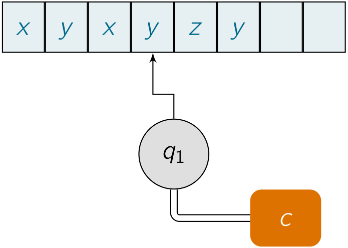


Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ $\quad (S, \Sigma, K)$-automaton

$$\tau = (q_1, x, p, q_2, f)$$

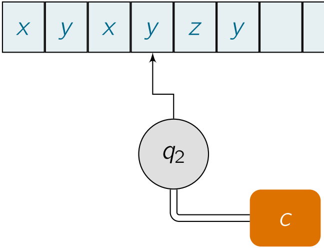


Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

$$\tau = (q_1, x, p, q_2, f)$$

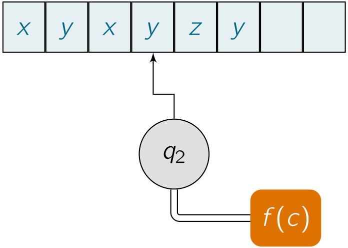


Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ $\quad (S, \Sigma, K)$-automaton

$$\tau = (q_1, x, p, q_2, f)$$

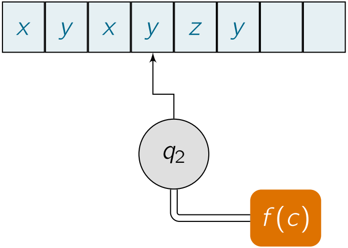


Weighted Automata with Storage

$S = (C, P, F, C_0)$ storage type
Σ alphabet
K unital valuation monoid

$A = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ (S, Σ, K)-automaton

$$\tau = (q_1, x, p, q_2, f)$$



computation step with τ

$$(q_1, xw, c) \vdash^{\tau} (q_2, w, f(c))$$

if $p(c) = \mathrm{true}$
and $f(c)$ defined

Weighted Automata with Storage

$w = x_1 \dots x_n \in \Sigma^*$
$\theta = \tau_1 \dots \tau_n, \quad \tau_i \in T$
$q_f \in Q_f$

Weighted Automata with Storage

$w = x_1 \ldots x_n \in \Sigma^*$
$\theta = \tau_1 \ldots \tau_n, \quad \tau_i \in T$
$q_f \in Q_f$

computation:

$$(q_0, x_1 x_2 \ldots x_n, c_0) \vdash^{\tau_1} (q_1, x_2 \ldots x_n, c_1) \vdash^{\tau_2} \ldots \vdash^{\tau_n} (q_f, \varepsilon, c_n)$$

Weighted Automata with Storage

$w = x_1 \dots x_n \in \Sigma^*$
$\theta = \tau_1 \dots \tau_n, \quad \tau_i \in T$
$q_f \in Q_f$

computation:

$$(q_0, x_1 x_2 \dots x_n, c_0) \vdash^{\tau_1} (q_1, x_2 \dots x_n, c_1) \vdash^{\tau_2} \dots \vdash^{\tau_n} (q_f, \varepsilon, c_n)$$

$\boxed{\Theta(w)}$

Weighted Automata with Storage

$w = x_1 \dots x_n \in \Sigma^*$
$\theta = \tau_1 \dots \tau_n, \quad \tau_i \in T$
$q_f \in Q_f$

computation:

$$(q_0, x_1 x_2 \dots x_n, c_0) \vdash^{\tau_1} (q_1, x_2 \dots x_n, c_1) \vdash^{\tau_2} \dots \vdash^{\tau_n} (q_f, \varepsilon, c_n)$$

$\boxed{\Theta(w)}$

weighted language recognized by A: $\|A\|\colon \Sigma^* \to K$

$$\|A\|(w) = \sum_{\theta \in \Theta(w)} \mathrm{wt}(\theta)$$

appropriate finiteness condition!

$$\mathrm{wt}(\tau_1 \dots \tau_n) = \mathrm{val}(\mathrm{wt}(\tau_1) \dots \mathrm{wt}(\tau_n))$$

Weighted Automata with Storage

$w = x_1 \dots x_n \in \Sigma^*$
$\theta = \tau_1 \dots \tau_n, \quad \tau_i \in T$
$q_f \in Q_f$

computation:

$$(q_0, x_1 x_2 \dots x_n, c_0) \vdash^{\tau_1} (q_1, x_2 \dots x_n, c_1) \vdash^{\tau_2} \dots \vdash^{\tau_n} (q_f, \varepsilon, c_n)$$

$\boxed{\Theta(w)}$

weighted language recognized by A: $\|A\| \colon \Sigma^* \to K$

$$\|A\|(w) = \sum_{\theta \in \Theta(w)} \mathrm{wt}(\theta)$$

appropriate finiteness condition!

$$\mathrm{wt}(\tau_1 \dots \tau_n) = \mathrm{val}(\mathrm{wt}(\tau_1) \dots \mathrm{wt}(\tau_n))$$

$\boxed{\mathrm{REC}(S, \Sigma, K)}$

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$

Examples for REC(S, Σ, K)

- REC(TRIV, Σ, $\mathbb{B}$) TRIV $= (\{c\}, \{p_{\text{true}}\}, \{f_{\text{id}}\}, \{c\})$

$$(q_0, x_1 x_2 \ldots x_n, c) \vdash^{\tau_1} (q_1, x_2 \ldots x_n, c) \vdash^{\tau_2} \ldots \vdash^{\tau_n} (q_n, \varepsilon, c)$$

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$ $\qquad$ $\mathrm{TRIV} = (\{c\}, \{p_{\mathrm{true}}\}, \{f_{\mathrm{id}}\}, \{c\})$

$$(q_0, x_1 x_2 \ldots x_n \quad) \vdash^{\tau_1} (q_1, x_2 \ldots x_n \quad) \vdash^{\tau_2} \ldots \vdash^{\tau_n} (q_n, \varepsilon \quad)$$

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$ recognizable languages

$$(q_0, x_1 x_2 \dots x_n \quad) \vdash^{\tau_1} (q_1, x_2 \dots x_n \quad) \vdash^{\tau_2} \dots \vdash^{\tau_n} (q_n, \varepsilon \quad)$$

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$ recognizable languages
- $\mathrm{REC}(\mathrm{P}^1, \Sigma, K)$

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$ recognizable languages
- $\mathrm{REC}(\mathrm{P}^1, \Sigma, K)$

$$(q_0, x_1 x_2 \ldots x_n, \square) \vdash^{\tau_1} (q_1, x_2 \ldots x_n, \begin{smallmatrix}\square\\ \square\end{smallmatrix}) \vdash^{\tau_2} \ldots \vdash^{\tau_n} (q_n, \varepsilon, \begin{smallmatrix}\square\\ \vdots\\ \square\end{smallmatrix})$$

Examples for REC(S, Σ, K)

- REC($\mathrm{TRIV}, \Sigma, \mathbb{B}$) recognizable languages
- REC(P^1, Σ, K) quantitative CF languages over Σ and K

[Droste, Vogler 13]

$$(q_0, x_1 x_2 \ldots x_n, \square) \vdash^{\tau_1} (q_1, x_2 \ldots x_n, \boxminus) \vdash^{\tau_2} \ldots \vdash^{\tau_n} (q_n, \varepsilon, \begin{smallmatrix}\square\\ \vdots \\ \square\end{smallmatrix})$$

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$ recognizable languages
- $\mathrm{REC}(\mathrm{P}^1, \Sigma, K)$ quantitative CF languages over Σ and K
  [Droste, Vogler 13]
- $\mathrm{REC}(\mathrm{P}^n, \Sigma, \mathbb{B})$

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$ recognizable languages
- $\mathrm{REC}(\mathrm{P}^1, \Sigma, K)$ quantitative CF languages over Σ and K
  [Droste, Vogler 13]
- $\mathrm{REC}(\mathrm{P}^n, \Sigma, \mathbb{B})$ iterated pushdown languages
  [Maslov 76; Engelfriet 83; Damm, Goerdt 86]

Examples for $\mathrm{REC}(S, \Sigma, K)$

- $\mathrm{REC}(\mathrm{TRIV}, \Sigma, \mathbb{B})$ — recognizable languages
- $\mathrm{REC}(\mathrm{P}^1, \Sigma, K)$ — quantitative CF languages over Σ and K
  [Droste, Vogler 13]
- $\mathrm{REC}(\mathrm{P}^n, \Sigma, \mathbb{B})$ — iterated pushdown languages
  [Maslov 76; Engelfriet 83; Damm, Goerdt 86]

Let $L\colon \Sigma^* \to K$.

L is a weighted iterated pushdown language if

$$L \in \bigcup_n \mathrm{REC}(\mathrm{P}^n, \Sigma, K)$$

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid.

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

$$r\colon \Sigma^* \to K$$

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

- separating weights

$$h : \Delta \to K[\Sigma \cup \{\varepsilon\}]$$
$$h(\delta) = a.y \qquad y \in \Sigma \cup \{\varepsilon\}$$

ε-free $(S, \Delta, \mathbb{B})$-automaton $\mathcal{A}$

$$\Delta^* \supseteq L(\mathcal{A})$$
$$\downarrow h$$
$$r\colon \Sigma^* \to K$$

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

$$
\begin{array}{ccc}
\mathrm{B}(\Omega, c) \subseteq \Omega^* & \xrightarrow{\mathcal{M}} & \Delta^* \supseteq L(\mathcal{A}) \\
 & & \big\downarrow h \\
 & & r\colon \Sigma^* \to K
\end{array}
$$

- separating weights
- separating storage

$\mathcal{M}$ generalized sequential machine (GSM)

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

$$
\begin{tikzcd}
 & R \subseteq \Phi^* \arrow[dl, "g_1"'] \arrow[dr, "g_2"] & \\
\mathrm{B}(\Omega, c) \subseteq \Omega^* \arrow[rr, "\mathcal{M}"] & & \Delta^* \supseteq L(\mathcal{A}) \arrow[d, "h"] \\
 & & r\colon \Sigma^* \to K
\end{tikzcd}
$$

- separating weights
- separating storage
- decomposition of GSM

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r \colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

$$\begin{tikzcd}
& R \subseteq \Phi^* \arrow[dl, "g_1"'] \arrow[dr, "g_2"] \arrow[ddr, bend left, "h'"] & \\
\mathrm{B}(\Omega, c) \subseteq \Omega^* \arrow[rr, "\mathcal{M}"] & & \Delta^* \supseteq L(\mathcal{A}) \arrow[d, "h"] \\
& & r\colon \Sigma^* \to K
\end{tikzcd}$$

- separating weights
- separating storage
- decomposition of GSM
- composition of g_2 and h

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r \colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

$$\begin{tikzcd}
 & R \subseteq \Phi^* \arrow[dl, "g_1"'] \arrow[dr, "g_2"] \arrow[ddr, bend left, "h'"] & \\
\mathrm{B}(\Omega, c) \subseteq \Omega^* \arrow[rr, "\mathcal{M}"] & & \Delta^* \supseteq L(\mathcal{A}) \arrow[d, "h"] \\
 & & r \colon \Sigma^* \to K
\end{tikzcd}$$

- separating weights
- separating storage
- decomposition of GSM
- composition of g_2 and h

Separating Storage

$$(q_0, ab, c_0) \; \underset{p_1(c_0) = \text{true}}{\overset{(q_0, a, p_1, q_1, f_1)}{\longmapsto}} \; (q_1, b, f_1(c_0)) \; \underset{p_2(f_1(c_0)) = \text{true}}{\overset{(q_1, b, p_2, q_2, f_2)}{\longmapsto}} \; (q_2, \varepsilon, f_2(f_1(c_0)))$$

Separating Storage

of $(S, \Delta, \mathbb{B})$-automaton $\mathcal{A}$

$$(q_0, ab, c_0) \xrightarrow[p_1(c_0) = \text{true}]{(q_0, a, p_1, q_1, f_1)} (q_1, b, f_1(c_0)) \xrightarrow[p_2(f_1(c_0)) = \text{true}]{(q_1, b, p_2, q_2, f_2)} (q_2, \varepsilon, f_2(f_1(c_0)))$$

$P_f \times F_f$

$(p_1, f_1)(p_2, f_2) \in (P_f \times F_f)^*$

Separating Storage

of $(S, \Delta, \mathbb{B})$-automaton $\mathcal{A}$

$$(q_0, ab, c_0) \;\underset{p_1(c_0) = \text{true}}{\overset{(q_0, a, p_1, q_1, f_1)}{\vdash}}\; (q_1, b, f_1(c_0)) \;\underset{p_2(f_1(c_0)) = \text{true}}{\overset{(q_1, b, p_2, q_2, f_2)}{\vdash}}\; (q_2, \varepsilon, f_2(f_1(c_0)))$$

$P_f \times F_f$

$(p_1, f_1)(p_2, f_2) \in (P_f \times F_f)^*$

$\boxed{\mathrm{B}(P_f \times F_f, c_0)}$

Separating Storage

of $(S, \Delta, \mathbb{B})$-automaton $\mathcal{A}$

$$(q_0, ab, c_0) \; \overset{(q_0,\, a,\, p_1,\, q_1,\, f_1)}{\underset{p_1(c_0) = \text{true}}{\longmapsto}} \; (q_1, b, f_1(c_0)) \; \overset{(q_1,\, b,\, p_2,\, q_2,\, f_2)}{\underset{p_2(f_1(c_0)) = \text{true}}{\longmapsto}} \; (q_2, \varepsilon, f_2(f_1(c_0)))$$

$P_f \times F_f$

$(p_1, f_1)(p_2, f_2) \in (P_f \times F_f)^*$

$$\boxed{\mathrm{B}(P_f \times F_f, c_0)}$$

$$\begin{tikzcd} q_0 \arrow[r, bend left, "(p_1, f_1) \,/\, a"] & q_1 \arrow[r, bend left, "(p_2, f_2) \,/\, b"] & q_2 \end{tikzcd}$$

$$\boxed{\text{GSM } \mathcal{M}}$$

Theorem (CS-Theorem for weighted automata with storage)

Let $S = (C, P, F, C_0)$ be a storage type, Σ an alphabet, and K a unital valuation monoid. If $r\colon \Sigma^* \to K$ is (S, Σ, K)-recognizable, then there are

- a regular language $R \subseteq \Phi^*$,
- a finite set $\Omega \subseteq P \times F$ and a configuration $c \in C$,
- a letter-to-letter morphism $g_1 : \Phi \to \Omega$, and
- an alphabetic morphism $h' : \Phi \to K[\Sigma \cup \{\varepsilon\}]$

such that $r = h'(g_1^{-1}(\mathrm{B}(\Omega, c)) \cap R)$.

$$
\begin{tikzcd}
& R \subseteq \Phi^* \arrow[dl, "g_1"'] \arrow[dr, "g_2"] \arrow[ddr, bend left, "h'"] & \\
\mathrm{B}(\Omega, c) \subseteq \Omega^* \arrow[rr, "\mathcal{M}"] & & \Delta^* \supseteq L(\mathcal{A}) \arrow[d, "h"] \\
& & r\colon \Sigma^* \to K
\end{tikzcd}
$$

- separating weights
- separating storage
- decomposition of GSM
- composition of g_2 and h

Instantiated CS-Theorem for (S, Σ, K)-recognizable languages

Instantiated CS-Theorem for (S, Σ, K)-recognizable languages

- $S = \mathrm{P}^n$: K-weighted n-iterated pushdown languages

Instantiated CS-Theorem for (S, Σ, K)-recognizable languages

- $S = \mathrm{P}^n$: K-weighted n-iterated pushdown languages
- $S = \mathrm{P}^1$: quantitative CF languages

  [Droste, Vogler 13]

Instantiated CS-Theorem for (S, Σ, K)-recognizable languages

- $S = \mathrm{P}^n$: K-weighted n-iterated pushdown languages
- $S = \mathrm{P}^1$: quantitative CF languages
  [Droste, Vogler 13]
- $S = \mathrm{SC}(\mathrm{TRIV})$: K-weighted stack automata
  [Greibach 69]

Instantiated CS-Theorem for (S, Σ, K)-recognizable languages

- $S = \mathrm{P}^n$: K-weighted n-iterated pushdown languages
- $S = \mathrm{P}^1$: quantitative CF languages
  [Droste, Vogler 13]
- $S = \mathrm{SC}(\mathrm{TRIV})$: K-weighted stack automata
  [Greibach 69]
- $S = \mathrm{NS}(\mathrm{TRIV})$: K-weighted nested stack automata
  [Engelfriet, Vogler 86]

Instantiated CS-Theorem for (S, Σ, K)-recognizable languages

- $S = \mathrm{P}^n$: K-weighted n-iterated pushdown languages
- $S = \mathrm{P}^1$: quantitative CF languages [Droste, Vogler 13]
- $S = \mathrm{SC}(\mathrm{TRIV})$: K-weighted stack automata [Greibach 69]
- $S = \mathrm{NS}(\mathrm{TRIV})$: K-weighted nested stack automata [Engelfriet, Vogler 86]
- $S = \mathrm{MON}(M)$: K-weighted M-automata for some monoid M [Kambites 07]

References I

N. Chomsky and M.-P. Schützenberger (1963). The algebraic theory of context-free languages. *Computer Programming and Formal Systems,* pp. 118–161.

T. Denkinger (2015). A Chomsky-Schützenberger representation for weighted multiplecontext-free languages. *FSMNLP*, accepted for publication.

M. Droste and H. Vogler (2013). The Chomsky-Schützenberger Theorem for Quantitative Context-Free Languages. *DLT,* pp. 203–214.

J. Engelfriet (2014). Context-Free Grammars with Storage. *CoRR abs/1408.0683*

S. Fratani and E.M. Voundy (2015). Dyck-based characterizations of indexed languages. published on arXiv http://arxiv.org/abs/1409.6112, March 13, 2015.

S. Greibach (1969). Checking automata and one-way stack languages. *Journal of Computer and System Sciences*, 3(2), pp. 196–217.

L. Herrmann and H. Vogler (2015). A Chomsky-Schützenberger Theorem for Weighted Automata with Storage. *CAI*, accepted for publication.

M. Kambites (2009). Formal languages and groups as memory. *Communications in Algebra*, 37(1), pp. 193–208.

References II

A. N. Maslov (1976). Multilevel Stack Automata. *Probl. Peredachi Inf.*, pp. 55–62.

W. Damm and A. Goerdt (1986). An automata-theoretical characterization of the OI-hierarchy. *Inform. Control*, 71:1-32.

D. Scott (1967). Some definitional suggestions for automata theory. *Journal of Computer and System Sciences 1.2*, pp. 187–212.

A. Salomaa and M. Soittola (1978). Automata-theoretic aspects of formal power series. *Springer New York*.

D. J. Weir (1988). Characterizing Mildly Context-Sensitive Grammar Formalisms. PhD thesis, University of Pennsylvania.

R. Yoshinaka, Y. Kaji, and H. Seki (2010). Chomsky-Schützenberger-type characterization of multiple context-free languages. *LATA*, pp. 596-607.

Separating Weights

"$\Rightarrow$"

$\mathcal{B} = (Q, \Sigma, c_0, q_0, Q_f, T, \mathrm{wt})$ $\rightsquigarrow$ $\mathcal{A} = (Q, T, c_0, q_0, Q_f, T')$

$\tau = (q, x, p, q', f)$ in T $\qquad$ $\tau' = (q, \tau, p, q', f)$ in T'

$h(\tau') = \mathrm{wt}(\tau).x$

"$\Leftarrow$"

$\mathcal{A} = (Q, \Delta, c_0, q_0, Q_f, T)$ $\rightsquigarrow$ $\mathcal{B} = (Q', \Sigma, c_0, q_0', \Delta \times Q_f, T', \mathrm{wt})$

$h\colon \Delta \to K[\Sigma \cup \{\varepsilon\}]$ $\qquad$ $Q' = \{q_0'\} \cup \Delta \times Q$

(q, δ, p, q', f) in T $\qquad$ $\tau = ((\delta', q), y, p, (\delta, q'), f)$ in T'

$h(\delta) = a.y$ $\qquad$ $\mathrm{wt}(\tau) = a$

Decomposition of GSM

gsm $\mathcal{M}$	$\leadsto$	regular grammar R, letter-to-letter morphisms h_1, h_2
$\tau = (q, x, q', y)$		$q \to \tau q'$ rule in R
		$h_1(\tau) = x$
		$h_2(\tau) = y$