Statistical Machine Translation of Natural Languages

Heiko Vogler
Technische Universität Dresden

Theorietag 2012
Prag, October 3, 2012

SMT = statistical machine translation

Weighted Tree Automata and Weighted Tree Transducers

can help in

Statistical Machine Translation of Natural Languages

Heiko Vogler
Technische Universität Dresden

Theorietag 2012
Prag, October 3, 2012

SMT = statistical machine translation
wta = weighted tree automata
wtt = weighted tree transducers

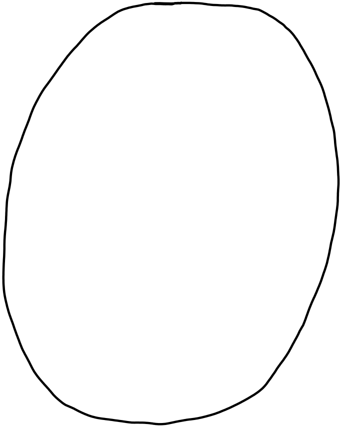
theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt
- input/output product
- Bar-Hillel, Shamir, Perles
  for wta and wsa
- characterization of wta
  (Büchi, Kleene, ...)
- variation of weight structure
  (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt
- input/output product
- Bar-Hillel, Shamir, Perles for wta and wsa
- characterization of wta (Büchi, Kleene, ...)
- variation of weight structure (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

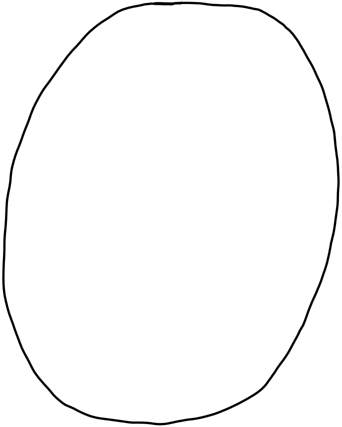
SMT



theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt
- input/output product
- Bar-Hillel, Shamir, Perles
  for wta and wsa
- characterization of wta
  (Büchi, Kleene, ...)
- variation of weight structure
  (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

SMT



[Kevin Knight et al. 03-...]

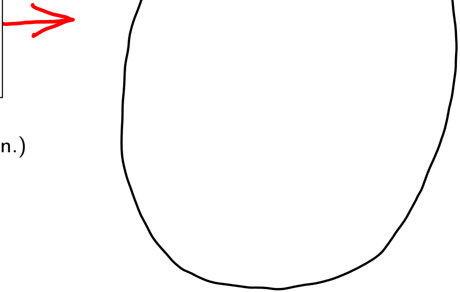
theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt

- input/output product
- Bar-Hillel, Shamir, Perles
  for wta and wsa

- variation of weight structure
  (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

SMT



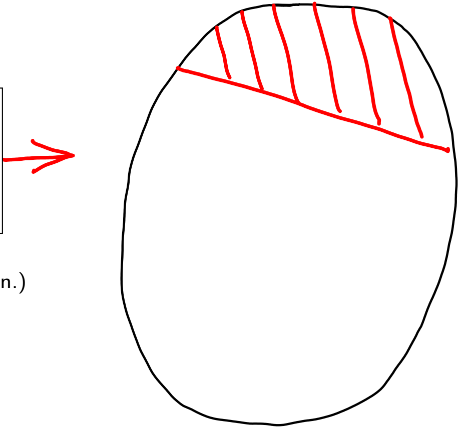
theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt

- input/output product
- Bar-Hillel, Shamir, Perles for wta and wsa

- variation of weight structure (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

SMT



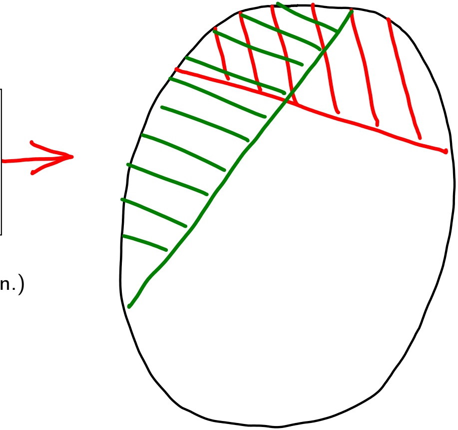
theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt

> - input/output product
> - Bar-Hillel, Shamir, Perles for wta and wsa

- variation of weight structure (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

SMT



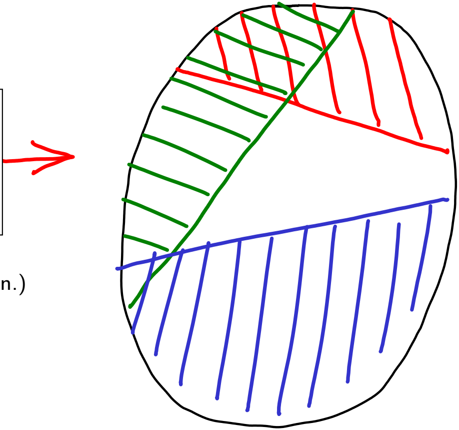
theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt

- input/output product
- Bar-Hillel, Shamir, Perles for wta and wsa

- variation of weight structure (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

SMT



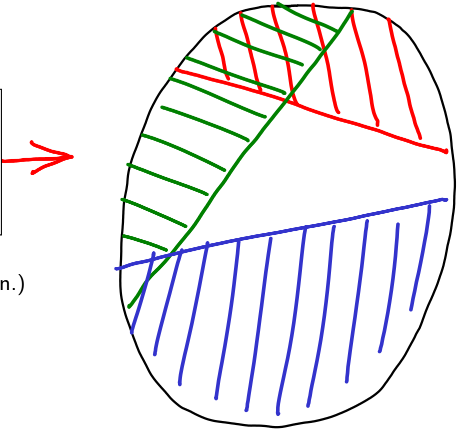
theory of wta and wtt

- determinization of wta
- minimization of wta
- (de-)composition of wtt

- input/output product
- Bar-Hillel, Shamir, Perles for wta and wsa

- variation of weight structure (semirings, fields, valuation mon.)
- ... [Fülöp, V. 09]

SMT



no survey on SMT!

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt
- Using output product to improve modelling

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt
- Using output product to improve modelling
- Using Bar-Hillel, Shamir, Perles and input product
  to improve decoding

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt
- Using output product to improve modelling
- Using Bar-Hillel, Shamir, Perles and input product
  to improve decoding

- Software system VANDA
  (M. Büchse, T. Dietze, J. Osterholzer)

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt
- Using output product to improve modelling
- Using Bar-Hillel, Shamir, Perles and input product
  to improve decoding

- Software system VANDA
  (M. Büchse, T. Dietze, J. Osterholzer)

given:

- source language SL
- target language TL

find:

$$\text{translation} \quad h : \mathrm{SL} \to \mathrm{TL}$$

e.g.

SL = English $\quad$ s = I saw the man with the telescope

TL = German $\quad$ h(s) = Ich sah den Mann durch das Tel.

given:

- source language SL
- target language TL

find:

machine translation $h : \mathrm{SL} \rightarrow \mathrm{TL}$

e.g.

SL = English s = I saw the man with the telescope
TL = German h(s) = Ich sah den Mann durch das Tel.

given:

- source language SL
- target language TL

find:

machine translation $h : \mathrm{SL} \to \mathrm{TL}$

e.g.

SL = English s = I saw the man with the telescope
TL = German h(s) = Ich sah den Mann durch das Tel.

machine translation $\rightsquigarrow$ statistical machine translation

[Lopez 08]: *"SMT treats the translation of natural languages as a machine learning problem."*

- assumptions $\longrightarrow$ modelling $\longrightarrow$ $\mathcal{H}$ hypothesis space

assumptions: mental work, experience, no data
hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \to \mathrm{TL}\}$

- assumptions $\longrightarrow$ [modelling] $\longrightarrow$ $\mathcal{H}$ hypothesis space

  assumptions: mental work, experience, no data
  hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \rightarrow \mathrm{TL}\}$

- $\mathcal{H}$ and training data $\longrightarrow$ [training] $\longrightarrow$ $\hat{h} \in \mathcal{H}$

  [Lopez 08]: *"By examining many samples of human-produced translations, SMT algorithms automatically learn how to translate."*

- assumptions $\longrightarrow$ [modelling] $\longrightarrow$ $\mathcal{H}$ hypothesis space

  assumptions: mental work, experience, no data
  hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \to \mathrm{TL}\}$

- $\mathcal{H}$ and training data $\longrightarrow$ [training] $\longrightarrow$ $\hat{h} \in \mathcal{H}$

  $\hat{h} = \mathrm{argmin}_{h \in \mathcal{H}} L(h, d)$, loss function $L : \mathcal{H} \times \mathcal{D} \to \mathbb{R}$

- assumptions $\longrightarrow$ modelling $\longrightarrow$ $\mathcal{H}$ hypothesis space

  assumptions: mental work, experience, no data
  hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \rightarrow \mathrm{TL}\}$

- $\mathcal{H}$ and training data $\longrightarrow$ training $\longrightarrow$ $\hat{h} \in \mathcal{H}$

  $\hat{h} = \mathrm{argmin}_{h \in \mathcal{H}} L(h, d)$, loss function $L : \mathcal{H} \times \mathcal{D} \rightarrow \mathbb{R}$

  given: $h \in \mathcal{H}$, $d = \{(s_1, t_1), (s_2, t_2), \ldots, (s_n, t_n)\} \in \mathcal{D}$

- assumptions $\longrightarrow$ [modelling] $\longrightarrow$ $\mathcal{H}$ hypothesis space

assumptions: mental work, experience, no data
hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \to \mathrm{TL}\}$

- $\mathcal{H}$ and training data $\longrightarrow$ [training] $\longrightarrow$ $\hat{h} \in \mathcal{H}$

$\hat{h} = \mathrm{argmin}_{h \in \mathcal{H}} L(h, d)$, loss function $L : \mathcal{H} \times \mathcal{D} \to \mathbb{R}$

given: $h \in \mathcal{H}$, $d = \{(s_1, t_1), (s_2, t_2), \ldots, (s_n, t_n)\} \in \mathcal{D}$
evaluate: $L(h, d) = \varphi\big((h(s_1), t_1), (h(s_2), t_2), \ldots, (h(s_n), t_n)\big)$

- assumptions $\longrightarrow$ [modelling] $\longrightarrow$ $\mathcal{H}$ hypothesis space

  assumptions: mental work, experience, no data
  hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \to \mathrm{TL}\}$

- $\mathcal{H}$ and training data $\longrightarrow$ [training] $\longrightarrow$ $\hat{h} \in \mathcal{H}$

  $\hat{h} = \mathrm{argmin}_{h \in \mathcal{H}} L(h, d)$, loss function $L : \mathcal{H} \times \mathcal{D} \to \mathbb{R}$

  given: $h \in \mathcal{H}$, $d = \{(s_1, t_1), (s_2, t_2), \ldots, (s_n, t_n)\} \in \mathcal{D}$

  evaluate: $L(h, d) = \varphi\big((h(s_1), t_1), (h(s_2), t_2), \ldots, (h(s_n), t_n)\big)$

  goal/hope: $\hat{h}(s)$ is also "good" for $s \notin \{s_1, \ldots, s_n\}$

- assumptions $\longrightarrow$ $\boxed{\text{modelling}}$ $\longrightarrow$ $\mathcal{H}$ hypothesis space

  assumptions: mental work, experience, no data
  hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \to \mathrm{TL}\}$

- $\mathcal{H}$ and training data $\longrightarrow$ $\boxed{\text{training}}$ $\longrightarrow$ $\hat{h} \in \mathcal{H}$

  $\hat{h} = \mathrm{argmin}_{h \in \mathcal{H}} L(h, d)$, loss function $L : \mathcal{H} \times \mathcal{D} \to \mathbb{R}$

- assumptions $\longrightarrow$ [modelling] $\longrightarrow$ $\mathcal{H}$ hypothesis space

  assumptions: mental work, experience, no data
  hypothesis space: $\mathcal{H} \subseteq \{h \mid h : \mathrm{SL} \rightarrow \mathrm{TL}\}$

- $\mathcal{H}$ and training data $\longrightarrow$ [training] $\longrightarrow$ $\hat{h} \in \mathcal{H}$

  $\hat{h} = \mathrm{argmin}_{h \in \mathcal{H}} L(h, d)$, loss function $L : \mathcal{H} \times \mathcal{D} \rightarrow \mathbb{R}$

- $\hat{h}$ and test data $\longrightarrow$ [evaluation] $\longrightarrow$ score

  BLEU (bilingual evaluation understanding), WER (word error rate), TER (translation error rate)

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt
- Using output product to improve modelling
- Using Bar-Hillel, Shamir, Perles to improve decoding

- Software system VANDA
  (M. Büchse, T. Dietze, J. Osterholzer)

first assumption:

SL and TL are generated by probabilistic context-free (cf) grammars

first assumption:

SL and TL are generated by probabilistic context-free (cf) grammars

S	→	*NP VP*	#	1.0
NP	→	*NP PP*	#	0.5
NP	→	I	#	0.3
NP	→	the man	#	0.2
PP	→	with the tel.	#	1.0
VP	→	saw *NP PP*	#	0.4
VP	→	saw *NP*	#	0.6

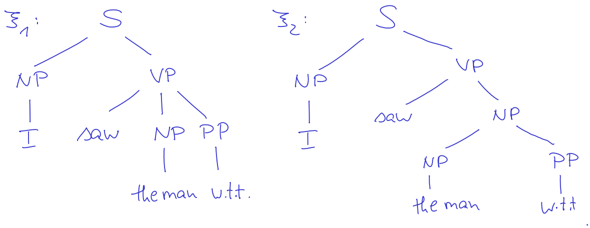
NP : noun phrase, VP: verb phrase, PP : prepositional phrase

first assumption:

SL and TL are generated by
probabilistic context-free (cf) grammars

S	→	*NP VP*	#	1.0					
NP	→	*NP PP*	#	0.5	*VP*	→	saw *NP PP*	#	0.4
NP	→	I	#	0.3	*VP*	→	saw *NP*	#	0.6
NP	→	the man	#	0.2					
PP	→	with the tel.	#	1.0					

derivation trees:



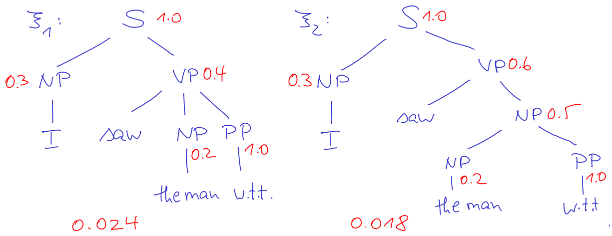
$\text{yield}(\xi_1) = \text{yield}(\xi_2) = \text{I saw the man w.t.t.}$

first assumption:

SL and TL are generated by
probabilistic context-free (cf) grammars

S	$\rightarrow$	$NP\ VP$	#	1.0					
NP	$\rightarrow$	$NP\ PP$	#	0.5	VP	$\rightarrow$	saw $NP\ PP$	#	0.4
NP	$\rightarrow$	I	#	0.3	VP	$\rightarrow$	saw NP	#	0.6
NP	$\rightarrow$	the man	#	0.2					
PP	$\rightarrow$	with the tel.	#	1.0					

derivation trees:



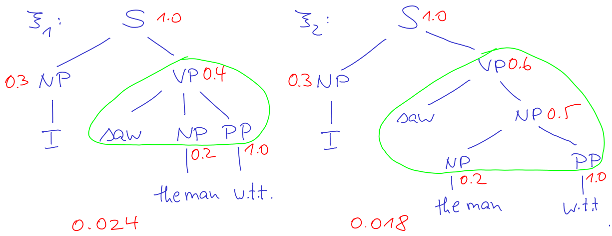
$\text{yield}(\xi_1) = \text{yield}(\xi_2) = \text{I saw the man w.t.t.}$

first assumption:

SL and TL are generated by probabilistic context-free (cf) grammars

S	→	*NP VP*	#	1.0					
NP	→	*NP PP*	#	0.5	*VP*	→	saw *NP PP*	#	0.4
NP	→	I	#	0.3	*VP*	→	saw *NP*	#	0.6
NP	→	the man	#	0.2					
PP	→	with the tel.	#	1.0					

derivation trees:



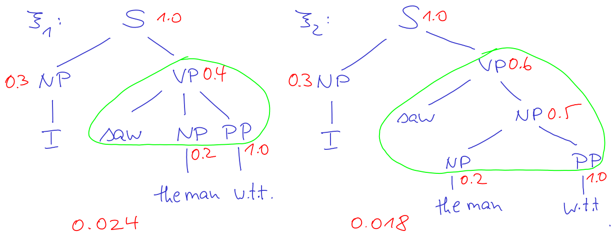
$\mathrm{yield}(\xi_1) = \mathrm{yield}(\xi_2) = \text{I saw the man w.t.t.}$

first assumption:

SL and TL are generated by
probabilistic context-free (cf) grammars

S	→	*NP VP*	#	1.0					
NP	→	*NP PP*	#	0.5	*VP*	→	saw *NP PP*	#	0.4
NP	→	I	#	0.3	*VP*	→	saw *NP*	#	0.6
NP	→	the man	#	0.2					
PP	→	with the tel.	#	1.0					

derivation trees:



prob. cf grammar for English `gram3.txt`

(Stanford-parser `http://nlp.stanford.edu:8080/parser/index.jsp`)

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.
- Let L be a recognizable tree language.
  There is a cf grammar $\mathcal{G}$ and a relabeling f s.t. $L = f(D_{\mathcal{G}})$.

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.
- Let L be a recognizable tree language.
  There is a cf grammar $\mathcal{G}$ and a relabeling f s.t. $L = f(D_{\mathcal{G}})$.

consider $L = \{S(S(a), S(b))\}$

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.
- Let L be a recognizable tree language.
  There is a cf grammar $\mathcal{G}$ and a relabeling f s.t. $L = f(D_{\mathcal{G}})$.

consider $L = \{S(S(a), S(b))\}$

- L is a recognizable tree language

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.
- Let L be a recognizable tree language.
  There is a cf grammar $\mathcal{G}$ and a relabeling f s.t. $L = f(D_{\mathcal{G}})$.

consider $L = \{S(S(a), S(b))\}$

- L is a recognizable tree language
- there is no cf grammar $\mathcal{G}$ s.t. $D_{\mathcal{G}} = L$

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.
- Let L be a recognizable tree language.
  There is a cf grammar $\mathcal{G}$ and a relabeling f s.t. $L = f(D_{\mathcal{G}})$.

consider $L = \{S(S(a), S(b))\}$ $S \to SS$ rule in $\mathcal{G}$

- L is a recognizable tree language
- there is no cf grammar $\mathcal{G}$ s.t. $D_{\mathcal{G}} = L$

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.
- Let L be a recognizable tree language.
  There is a cf grammar $\mathcal{G}$ and a relabeling f s.t. $L = f(D_{\mathcal{G}})$.

consider $L = \{S(S(a), S(b))\}$ $\qquad$ $S \to SS$ rule in $\mathcal{G}$

- L is a recognizable tree language
- there is no cf grammar $\mathcal{G}$ s.t. $D_{\mathcal{G}} = L$
- cf grammar $\mathcal{G}'$:

$$\begin{array}{lcl} S & \to & S_a\, S_b \\ S_a & \to & a \\ S_b & \to & b \end{array} \qquad f(S_a) = f(S_b) = S$$

Theorem [Thatcher 67]

- Let $\mathcal{G}$ be a cf grammar. $D_{\mathcal{G}}$ is a recognizable tree language.
- Let L be a recognizable tree language.
  There is a cf grammar $\mathcal{G}$ and a relabeling f s.t. $L = f(D_{\mathcal{G}})$.

consider $L = \{S(S(a), S(b))\}$ $\quad$ $S \to SS$ rule in $\mathcal{G}$

- L is a recognizable tree language
- there is no cf grammar $\mathcal{G}$ s.t. $D_{\mathcal{G}} = L$
- cf grammar $\mathcal{G}'$:

$$\begin{aligned} S &\to S_a\, S_b \qquad\qquad f(S_a) = f(S_b) = S \\ S_a &\to a \\ S_b &\to b \end{aligned}$$

recognizable tree languages are closed under relabelings.

first assumption:

SL and TL are generated by
probabilistic context-free (cf) grammars

refined first assumption:

SL and TL are the yields of
weighted recognizable tree languages

first assumption:

SL and TL are generated by
probabilistic context-free (cf) grammars

refined first assumption:

SL and TL are the yields of
weighted recognizable tree languages

weighted tree language: $L : T_\Sigma \to \mathbb{R}$

first assumption:

SL and TL are generated by
probabilistic context-free (cf) grammars

refined first assumption:

SL and TL are the yields of
weighted recognizable tree languages

weighted tree language: $L : T_{\Sigma} \to \mathbb{R}$

L is recognizable:

if there is a wta $\mathcal{A}$
which "recognizes" (computes) L

weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

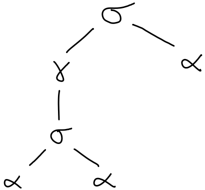
- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\qquad \delta_\sigma : Q^k \times Q \to \mathbb{R}$

  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\qquad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

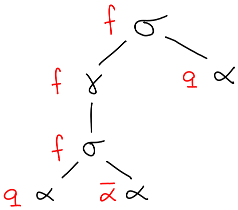
run on $\xi \in T_\Sigma$: $\quad r : \mathrm{pos}(\xi) \to Q$ $\qquad$ set of runs on ξ: $\quad R_{\mathcal{A}}(\xi)$



weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\quad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $\quad r : \mathrm{pos}(\xi) \to Q$ $\qquad$ set of runs on ξ: $\quad R_{\mathcal{A}}(\xi)$



weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

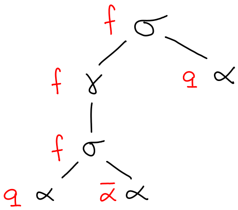
- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\quad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $r : \mathrm{pos}(\xi) \to Q$ set of runs on ξ: $R_\mathcal{A}(\xi)$

weight of r: $\mathrm{wt}(r) = \prod_{w \in \mathrm{pos}(\xi)} \delta_\sigma(r(w1) \cdots r(wk), r(w))$

σ: label of ξ at w

k: rank of σ

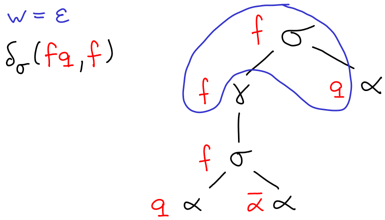


weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\quad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $\quad r : \mathrm{pos}(\xi) \to Q$ $\qquad$ set of runs on ξ: $\quad R_{\mathcal{A}}(\xi)$

weight of r: $\quad \mathrm{wt}(r) = \prod_{w \in \mathrm{pos}(\xi)} \delta_\sigma(r(w1) \cdots r(wk), r(w))$



σ: label of ξ at w

k: rank of σ

weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

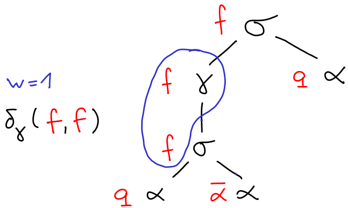
- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\quad \delta_\sigma : Q^k \times Q \to \mathbb{R}$, $\quad \delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $\quad r : \mathrm{pos}(\xi) \to Q$ $\qquad$ set of runs on ξ: $\quad R_{\mathcal{A}}(\xi)$

weight of r: $\quad \mathrm{wt}(r) = \prod_{w \in \mathrm{pos}(\xi)} \delta_\sigma(r(w1) \cdots r(wk), r(w))$

σ: label of ξ at w

k: rank of σ



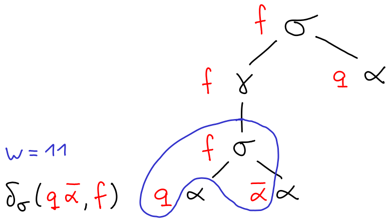
weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\quad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $r : \mathrm{pos}(\xi) \to Q$ set of runs on ξ: $R_{\mathcal{A}}(\xi)$

weight of r: $\mathrm{wt}(r) = \prod_{w \in \mathrm{pos}(\xi)} \delta_\sigma(r(w1) \cdots r(wk), r(w))$

σ: label of ξ at w
k: rank of σ



weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

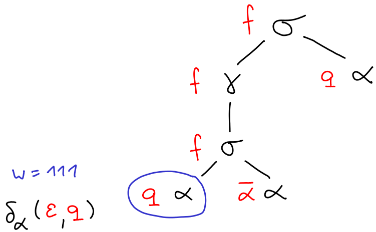
- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\qquad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $\quad r : \mathrm{pos}(\xi) \to Q$ $\qquad$ set of runs on ξ: $\quad R_\mathcal{A}(\xi)$

weight of r: $\quad \mathrm{wt}(r) = \prod_{w \in \mathrm{pos}(\xi)} \delta_\sigma(r(w1) \cdots r(wk), r(w))$

σ: label of ξ at w

k: rank of σ



weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\quad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $\quad r : \mathrm{pos}(\xi) \to Q$ $\qquad$ set of runs on ξ: $\quad R_\mathcal{A}(\xi)$

weight of r: $\quad \mathrm{wt}(r) = \prod_{w \in \mathrm{pos}(\xi)} \delta_\sigma(r(w1) \cdots r(wk), r(w))$

σ: label of ξ at w

k: rank of σ

weighted tree automaton (wta) $\mathcal{A} = (Q, \Sigma, \delta, F)$

- Q finite set (states)
- Σ ranked alphabet (input symbols)
- $\delta = (\delta_\sigma \mid \sigma \in \Sigma)$ $\quad \delta_\sigma : Q^k \times Q \to \mathbb{R}$
  $\delta_\sigma(q_1 \cdots q_k, q) \in \mathbb{R}$
- $F \subseteq Q$ (final states)

run on $\xi \in T_\Sigma$: $\quad r : \mathrm{pos}(\xi) \to Q$ $\qquad$ set of runs on ξ: $\quad R_\mathcal{A}(\xi)$

weight of r: $\quad \mathrm{wt}(r) = \prod_{w \in \mathrm{pos}(\xi)} \delta_\sigma(r(w1) \cdots r(wk), r(w))$

σ: label of ξ at w

k: rank of σ

weighted tree language recognized by $\mathcal{A}$:

$$L_\mathcal{A} : T_\Sigma \to \mathbb{R}, \quad L_\mathcal{A}(\xi) = \max_{r \in R_\mathcal{A}(\xi)} \mathrm{wt}(r)$$

second assumption:

translation from SL and TL is specified by a weighted tree transducer

[Yamada, Knight 01] translation from English to Japanese

weighted tree transducer (wtt) $\mathcal{M} = (Q, \Sigma, q_0, R)$

- Q, Σ as for wta

Σ: input and output symbols

- $q_0 \in Q$ (initial state)

weighted tree transducer (wtt) $\mathcal{M} = (Q, \Sigma, q_0, R)$

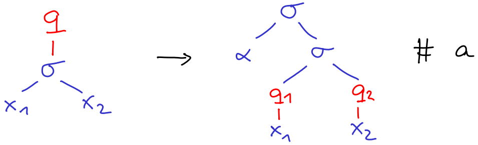
- Q, Σ as for wta

Σ: input and output symbols

- $q_0 \in Q$ (initial state)
- R finite set of particular term rewrite rules with weights

$$\rho : \; q(\sigma(x_1, \ldots, x_k)) \to \xi[q_1(x_1), \ldots, q_k(x_k)] \; \# \; a$$

$q, q_1, \ldots, q_k \in Q$, $\sigma \in \Sigma$, $\xi \in T_\Sigma(X_k)$
ξ linear, nondeleting in $x_1, \ldots, x_k$
$a \in \mathbb{R}$



weighted tree transducer (wtt) $\mathcal{M} = (Q, \Sigma, q_0, R)$

- Q, Σ as for wta

Σ: input <u>and</u> output symbols

- $q_0 \in Q$ (initial state)
- R finite set of particular term rewrite rules with weights

$$\rho : \; q(\sigma(x_1, \ldots, x_k)) \to \xi[q_1(x_1), \ldots, q_k(x_k)] \mathbin{\#} a$$

weighted tree transducer (wtt) $\mathcal{M} = (Q, \Sigma, q_0, R)$

- Q, Σ as for wta Σ: input <u>and</u> output symbols
- $q_0 \in Q$ (initial state)
- R finite set of particular term rewrite rules with weights

$$\rho: \; q(\sigma(x_1, \ldots, x_k)) \to \xi[q_1(x_1), \ldots, q_k(x_k)] \# a$$

(leftmost) derivation: $d = \rho_1 \cdots \rho_n$

weighted tree transducer (wtt) $\mathcal{M} = (Q, \Sigma, q_0, R)$

- Q, Σ as for wta — Σ: input <u>and</u> output symbols
- $q_0 \in Q$ (initial state)
- R finite set of particular term rewrite rules with weights

$$\rho: \; q(\sigma(x_1, \ldots, x_k)) \to \xi[q_1(x_1), \ldots, q_k(x_k)] \# a$$

(leftmost) derivation: $d = \rho_1 \cdots \rho_n$

weight of a derivation d: $\mathrm{wt}(d) = \prod_{i=1}^{n} \mathrm{wt}(\rho_i)$

weighted tree transducer (wtt) $\mathcal{M} = (Q, \Sigma, q_0, R)$

- Q, Σ as for wta

Σ: input <u>and</u> output symbols

- $q_0 \in Q$ (initial state)
- R finite set of particular term rewrite rules with weights

$$\rho : \; q(\sigma(x_1, \ldots, x_k)) \rightarrow \xi[q_1(x_1), \ldots, q_k(x_k)] \; \# \; a$$

(leftmost) derivation: $d = \rho_1 \cdots \rho_n$

weight of a derivation d: $\mathrm{wt}(d) = \prod_{i=1}^{n} \mathrm{wt}(\rho_i)$

weighted tree transformation computed by $\mathcal{M}$:

$$\tau_{\mathcal{M}} : T_\Sigma \times T_\Sigma \rightarrow \mathbb{R}, \quad \tau_{\mathcal{M}}(\xi_1, \xi_2) = \max_{\substack{d \in D_{\mathcal{M}}: \\ \pi(d) = (\xi_1, \xi_2)}} \mathrm{wt}(d)$$

$D_{\mathcal{M}}$: set of all derivations

modelling with wtt and wta:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$

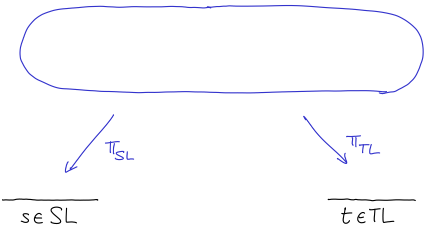
modelling with wtt and wta:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$

$s \in SL$ $t \in TL$

modelling with wtt and wta:

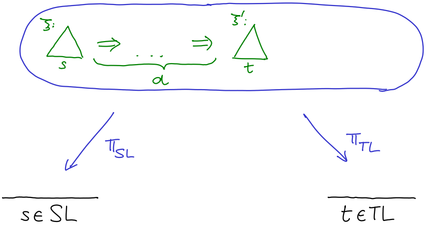
- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$



correspondence structure:
[Liang et al. 06]

modelling with wtt and wta:

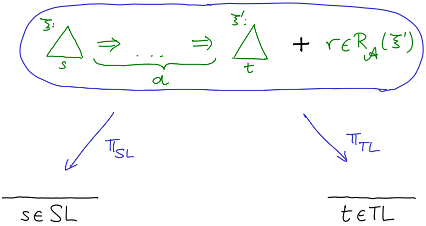
- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$



correspondence structure:
[Liang et al. 06]

modelling with wtt and wta:

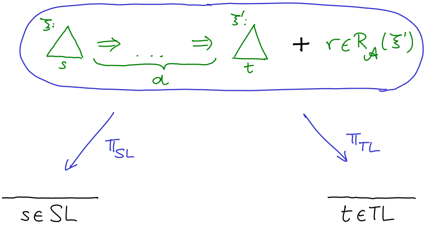
- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$



correspondence structure:
[Liang et al. 06]

modelling with wtt and wta:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$



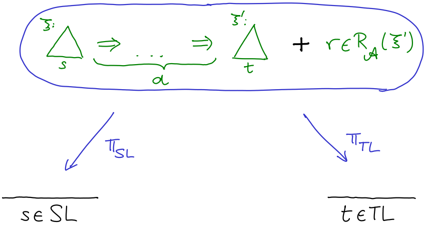
correspondence structure:
[Liang et al. 06]

$$Y = \{(d, r) \in D_{\mathcal{M}} \times R_{\mathcal{A}} \mid r \in R_{\mathcal{A}}(\mathrm{last}(d))\}$$

$$\pi_{SL}(d, r) = \mathrm{yield}(\mathrm{first}(d))$$
$$\pi_{TL}(d, r) = \mathrm{yield}(\mathrm{last}(d))$$

modelling with wtt and wta:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$



correspondence structure:
[Liang et al. 06]

$$Y = \{(d, r) \in D_{\mathcal{M}} \times R_{\mathcal{A}} \mid r \in R_{\mathcal{A}}(\mathrm{last}(d))\}$$

$$\pi_{SL}(d, r) = \mathrm{yield}(\mathrm{first}(d))$$
$$\pi_{TL}(d, r) = \mathrm{yield}(\mathrm{last}(d))$$

hypothesis space: $\mathcal{H} = \{h_{\mathcal{M},\mathcal{A}} \mid \mathrm{wtt}\ \mathcal{M}, \mathrm{wta}\ \mathcal{A}\}$

$$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$$
$$s \mapsto \pi_{\mathrm{TL}}\left(\operatorname*{argmax}_{\substack{(d,r)\in Y: \\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d) \cdot \mathrm{wt}(r)\right)$$

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt
- Using output product to improve modelling
- Using Bar-Hillel, Shamir, Perles and input product
  to improve decoding

- Software system VANDA
  (M. Büchse, T. Dietze, J. Osterholzer)

Let $\tau : T_\Sigma \times T_\Sigma \to \mathbb{R}$ and $L : T_\Sigma \to \mathbb{R}$

Let $\tau : T_\Sigma \times T_\Sigma \to \mathbb{R}$ and $L : T_\Sigma \to \mathbb{R}$

output product of τ and L:

$$\tau \triangleright L : T_\Sigma \times T_\Sigma \to \mathbb{R}$$
$$(\tau \triangleright L)(\xi, \zeta) \mapsto \tau(\xi, \zeta) \cdot L(\zeta)$$

Let $\tau : T_\Sigma \times T_\Sigma \to \mathbb{R}$ and $L : T_\Sigma \to \mathbb{R}$

output product of τ and L:

$$\tau \triangleright L : T_\Sigma \times T_\Sigma \to \mathbb{R}$$

$$(\tau \triangleright L)(\xi, \zeta) \mapsto \tau(\xi, \zeta) \cdot L(\zeta)$$

input product of L and τ:

$$L \triangleleft \tau : T_\Sigma \times T_\Sigma \to \mathbb{R}$$

$$(L \triangleleft \tau)(\xi, \zeta) \mapsto L(\xi) \cdot \tau(\xi, \zeta)$$

Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.

Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

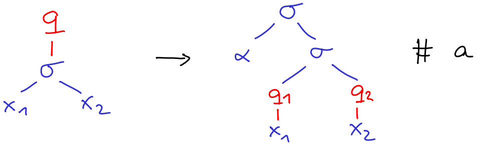
Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

Proof: [Baker 79, Engelfriet, Fülöp,V. 02]

Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M}\triangleright\mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

Proof: [Baker 79, Engelfriet, Fülöp,V. 02]

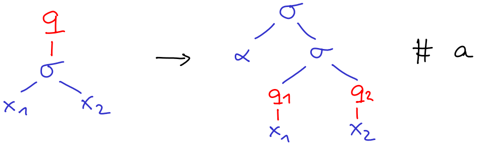
rule of $\mathcal{M}$:



Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M}\triangleright\mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

Proof: [Baker 79, Engelfriet, Fülöp,V. 02]

rule of $\mathcal{M}$:

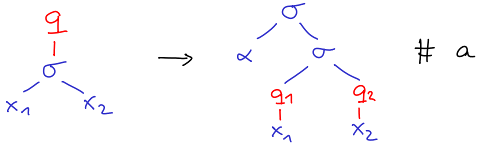


states of $\mathcal{A}$: p, p_1, p_2

Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

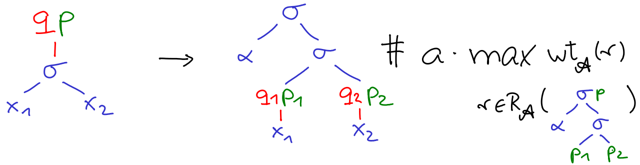
Proof: [Baker 79, Engelfriet, Fülöp,V. 02]

rule of $\mathcal{M}$:

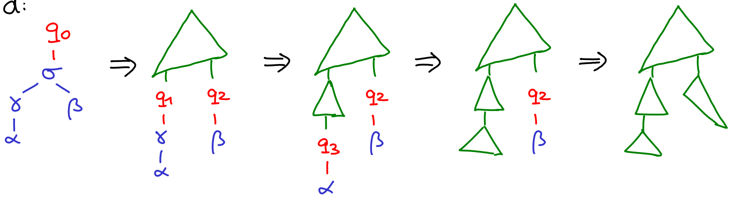


states of $\mathcal{A}$: p, p_1, p_2

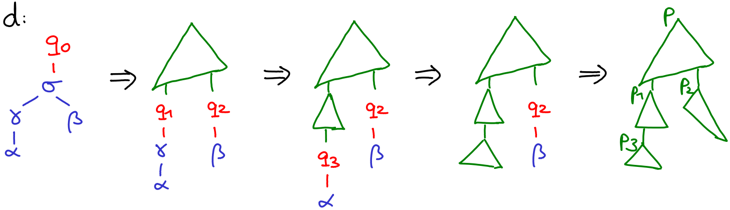
rule of $\mathcal{M} \triangleright \mathcal{A}$:



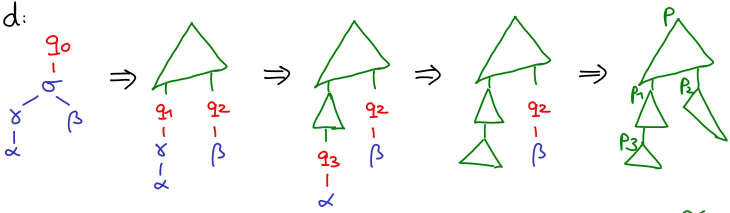
d:



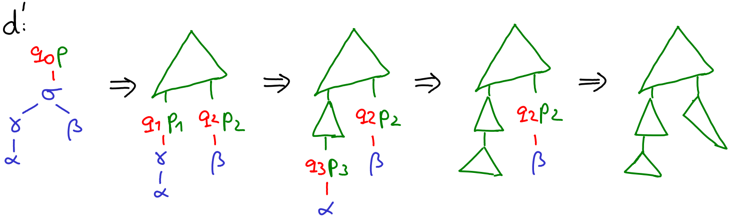
q_0
σ
γ
β
α
$\Rightarrow$
q_1
q_2
γ
β
α
$\Rightarrow$
q_2
q_3
β
α
$\Rightarrow$
q_2
β
$\Rightarrow$

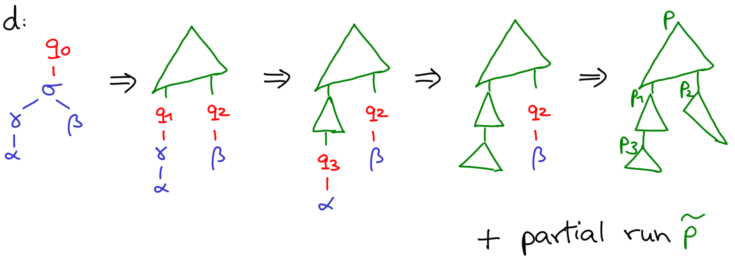


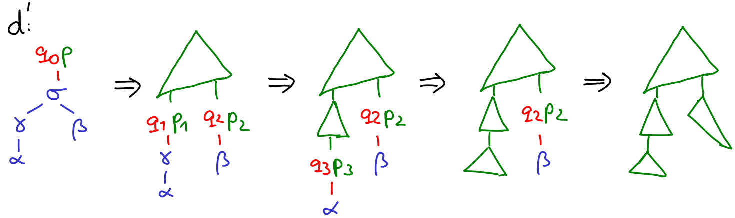
+ partial run $\tilde{p}$



+ partial run $\tilde{p}$







$\varphi : D_{\mathcal{M}'} \to \{(d, \tilde{p}) \mid d \in D_{\mathcal{M}}, \tilde{p} \in R_{\mathcal{A}}^{\text{partial}}(d)\}$ bijection

$\mathrm{wt}(d') = \mathrm{wt}(d) \cdot \max_{r \in \mathrm{completion}(\tilde{p})} \mathrm{wt}(r)$

recall:

$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$

$$s \quad \mapsto \quad \pi_{\mathrm{TL}} \left(\operatorname{argmax}_{\substack{(d,r) \in Y: \\ \pi_{\mathrm{SL}}(d,r) = s}} \mathrm{wt}(d) \cdot \mathrm{wt}(r) \right)$$

$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$

$$s \mapsto \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d)\cdot\mathrm{wt}(r)\right)$$

$$= \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\cdot\max_{r\in R_{\mathcal{A}}(\mathrm{last}(d))}\mathrm{wt}(r)\right)$$

$$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$$

$$\begin{aligned}
s \quad &\mapsto \quad \pi_{\mathrm{TL}} \left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d) \cdot \mathrm{wt}(r) \right) \\
&= \quad \pi_{\mathrm{TL}} \left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d) \cdot \max_{r\in R_{\mathcal{A}}(\mathrm{last}(d))} \mathrm{wt}(r) \right) \\
&= \quad \pi_{\mathrm{TL}} \left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d) \cdot \max_{\tilde{p}\in R^{\mathrm{p}}_{\mathcal{A}}(d)} \max_{r\in \mathrm{compl.}(\tilde{p})} \mathrm{wt}(r) \right)
\end{aligned}$$

$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$

$$
\begin{aligned}
s \quad \mapsto \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d)\cdot\mathrm{wt}(r)\right)\\
= \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\cdot\max_{r\in R_{\mathcal{A}}(\mathrm{last}(d))}\mathrm{wt}(r)\right)\\
= \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\cdot\max_{\tilde{p}\in R^{\mathrm{p}}_{\mathcal{A}}(d)}\max_{r\in\mathrm{compl.}(\tilde{p})}\mathrm{wt}(r)\right)\\
= \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s\\ \tilde{p}\in R^{\mathrm{p}}_{\mathcal{A}}(d)}} \mathrm{wt}(d)\cdot\max_{r\in\mathrm{compl.}(\tilde{p})}\mathrm{wt}(r)\right)
\end{aligned}
$$

$$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$$

$$
\begin{aligned}
s \quad \mapsto \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d)\cdot \mathrm{wt}(r)\right) \\
= \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\cdot \max_{r\in R_{\mathcal{A}}(\mathrm{last}(d))} \mathrm{wt}(r)\right) \\
= \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\cdot \max_{\tilde{p}\in R^{\mathrm{p}}_{\mathcal{A}}(d)} \max_{r\in \mathrm{compl.}(\tilde{p})} \mathrm{wt}(r)\right) \\
= \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}}\\ \pi_{\mathrm{SL}}(d)=s\\ \tilde{p}\in R^{\mathrm{p}}_{\mathcal{A}}(d)}} \mathrm{wt}(d)\cdot \max_{r\in \mathrm{compl.}(\tilde{p})} \mathrm{wt}(r)\right) \\
= \quad & \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}\triangleright\mathcal{A}}\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\right)
\end{aligned}
$$

recall:

Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

recall:

Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

generalization to mildly context-sensitive languages

Theorem [Büchse, Nederhof, V. 11]:
Let $\mathcal{M}$ synchronized tree-adjoining grammar (STAG)
and $\mathcal{A}$ wta.
There is an STAG $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

recall:

Theorem [Maletti 06]: Let $\mathcal{M}$ wtt and $\mathcal{A}$ wta.
There is a wtt $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

generalization to mildly context-sensitive languages

Theorem [Büchse, Nederhof, V. 11]:
Let $\mathcal{M}$ synchronized tree-adjoining grammar (STAG)
and $\mathcal{A}$ wta.
There is an STAG $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

Theorem [Nederhof, V. 12]:
Let $\mathcal{M}$ synchronized context-free tree grammar (SCFTG)
and $\mathcal{A}$ wta.
There is an SCFTG $\mathcal{M} \triangleright \mathcal{A}$ such that: $\tau_{\mathcal{M} \triangleright \mathcal{A}} = \tau_{\mathcal{M}} \triangleright L_{\mathcal{A}}$

outline of the talk:

- Statistical machine translation
  (modelling, training, evaluation)
- Modelling with wta and wtt
- Using output product to improve modelling
- Using Bar-Hillel, Shamir, Perles and input product
  to improve decoding

- Software system VANDA
  (M. Büchse, T. Dietze, J. Osterholzer)

decoding:

given: $h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$ and $s \in \mathrm{SL}$

compute: $h_{\mathcal{M},\mathcal{A}}(s) = \pi_{\mathrm{TL}}\left(\mathrm{argmax}_{\substack{d \in D_{\mathcal{M} \triangleright \mathcal{A}}: \\ \pi_{\mathrm{SL}}(d) = s}} \ \mathrm{wt}(d)\right)$

decoding:

given: $h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$ and $s \in \mathrm{SL}$

compute: $h_{\mathcal{M},\mathcal{A}}(s) = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d \in D_{\mathcal{M} \triangleright \mathcal{A}}: \\ \pi_{\mathrm{SL}}(d) = s}} \mathrm{wt}(d)\right)$

Algorithm:

- apply input product to wta $\mathcal{A}_s$ and wtt $\mathcal{M} \triangleright \mathcal{A}$ resulting in wtt $\mathcal{A}_s \triangleleft (\mathcal{M} \triangleright \mathcal{A})$
- apply Knuth's algorithm to $\mathcal{A}_s \triangleleft (\mathcal{M} \triangleright \mathcal{A})$ resulting in the derivation with maximal weight.

decoding:

given: $h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$ and $s \in \mathrm{SL}$

compute: $$h_{\mathcal{M},\mathcal{A}}(s) = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d \in D_{\mathcal{M} \triangleright \mathcal{A}}:\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\right)$$

Algorithm:

- apply input product to wta $\mathcal{A}_s$ and wtt $\mathcal{M} \triangleright \mathcal{A}$ resulting in wtt $\mathcal{A}_s \triangleleft (\mathcal{M} \triangleright \mathcal{A})$
- apply Knuth's algorithm to $\mathcal{A}_s \triangleleft (\mathcal{M} \triangleright \mathcal{A})$ resulting in the derivation with maximal weight.

given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

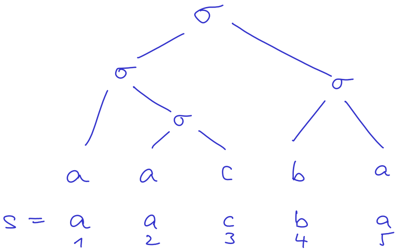
idea:

given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

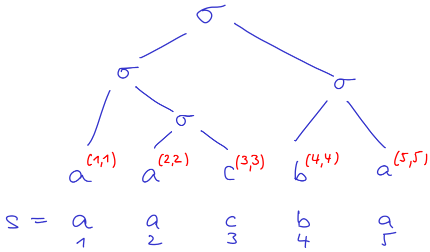


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

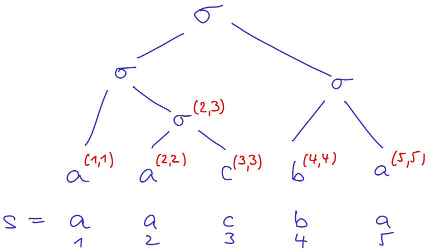


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

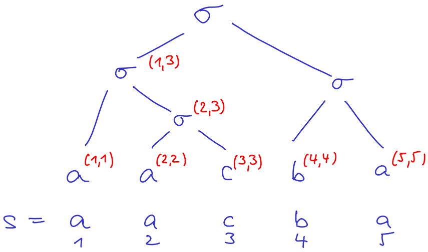


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

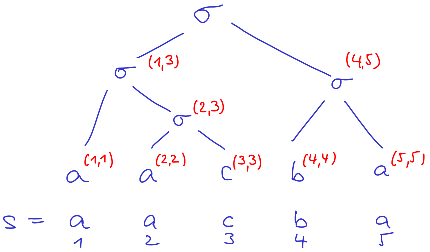


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

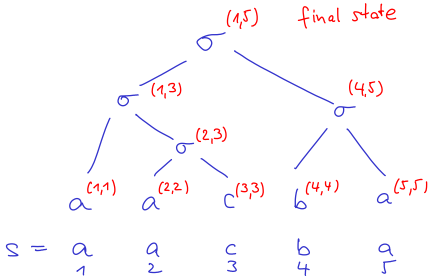


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

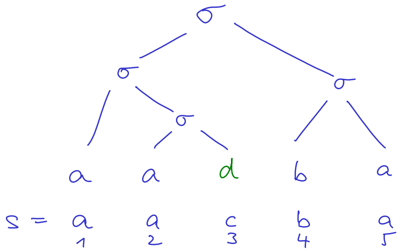


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

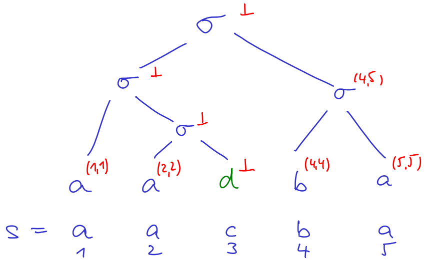


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:

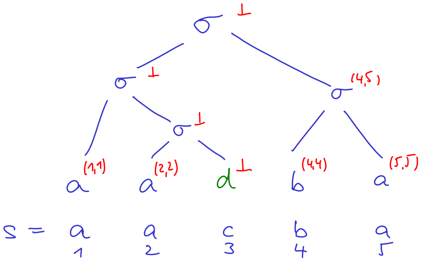


given: ranked alphabet Σ and $s = a_1 \cdots a_n$

construct: wta $\mathcal{A}_s$ such that for every $\xi \in T_\Sigma$

$$L_{\mathcal{A}_s}(\xi) = \begin{cases} 1 & \text{if } \mathrm{yield}(\xi) = s \\ 0 & \text{otherwise} \end{cases}$$

idea:



> Theorem [Bar-Hillel, Shamir, Perles 61]
> The class of cf languages is closed under intersection with regular languages.

recall:

$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$

$$s \quad \mapsto \quad \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d \in D_{\mathcal{M} \triangleright \mathcal{A}}: \\ \pi_{\mathrm{SL}}(d) = s}} \mathrm{wt}(d)\right)$$

$$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$$

$$s \mapsto \pi_{\mathrm{TL}} \left(\mathrm{argmax}_{\substack{d \in D_{\mathcal{M} \triangleright \mathcal{A}}: \\ \pi_{\mathrm{SL}}(d) = s}} \mathrm{wt}(d) \right)$$

$$= \pi_{\mathrm{TL}} \left(\mathrm{argmax}_{\substack{(d,r): \\ d \in D_{\mathcal{M} \triangleright \mathcal{A}} \\ r \in R_{\mathcal{A}_s}(\mathrm{first}(d))}} \mathrm{wt}(r) \cdot \mathrm{wt}(d) \right)$$

$h_{\mathcal{M},\mathcal{A}} : \mathrm{SL} \to \mathrm{TL}$

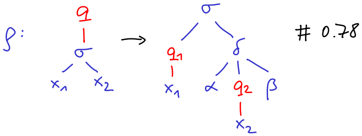
$$
\begin{aligned}
s &\mapsto \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d \in D_{\mathcal{M} \triangleright \mathcal{A}}:\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\right) \\
&= \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r):\\ d \in D_{\mathcal{M} \triangleright \mathcal{A}}\\ r \in R_{\mathcal{A}_s}(\mathrm{first}(d))}} \mathrm{wt}(r) \cdot \mathrm{wt}(d)\right) \\
&= \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{d \in D_{\mathcal{A}_s \triangleleft (\mathcal{M} \triangleright \mathcal{A})}} \mathrm{wt}(d)\right)
\end{aligned}
$$

recall:

Algorithm:

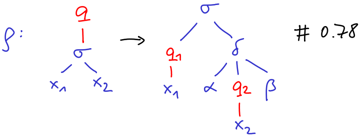
- apply input product to wta $\mathcal{A}_s$ and wtt $\mathcal{M} \triangleright \mathcal{A}$ resulting in wtt $\mathcal{A}_s \triangleleft (\mathcal{M} \triangleright \mathcal{A})$
- apply Knuth's algorithm to $\mathcal{A}_s \triangleleft (\mathcal{M} \triangleright \mathcal{A})$ resulting in the derivation with maximal weight.

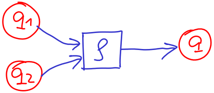
wtt $\mathcal{M}' := \mathcal{A}_s \lhd (\mathcal{M} \rhd \mathcal{A})$



wtt $\mathcal{M}' := \mathcal{A}_s \lhd (\mathcal{M} \rhd \mathcal{A})$

hypergraph $G(\mathcal{M}')$



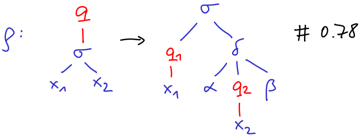


$g_\rho : [0,1] \times [0,1] \to [0,1]$

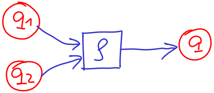
$(n_1, n_2) \mapsto n_1 \cdot n_2 \cdot 0.78$

- $g_\rho(n_1, n_2) \leq \min(n_1, n_2)$
- g_ρ monotone

wtt $\mathcal{M}' := \mathcal{A}_s \lhd (\mathcal{M} \rhd \mathcal{A})$



hypergraph $G(\mathcal{M}')$



$g_\rho : [0,1] \times [0,1] \to [0,1]$

$(n_1, n_2) \mapsto n_1 \cdot n_2 \cdot 0.78$

- $g_\rho(n_1, n_2) \leq \min(n_1, n_2)$
- g_ρ monotone

hypergraph $G(\mathcal{M}')$ $\longrightarrow$ **Knuth's algorithm** $\longrightarrow$ $\operatorname{argmax}_{d \in D_{\mathcal{M}'}} \mathrm{wt}_{\mathcal{M}'}(d)$

[Knuth 77] A generalization of Dijkstra's shortest path algorithm

$\mathcal{O}(|E| \cdot \log |V|)$

summary:

summary:

- model for translation from SL to TL: wtt $\mathcal{M}$

summary:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$

summary:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$
- sentence s of SL

summary:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$
- sentence s of SL

noisy channel:

$$h_{\mathcal{M},\mathcal{A}}(s) \quad = \quad \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d)\cdot\mathrm{wt}(r)\right)$$

summary:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$
- sentence s of SL

noisy channel:

$$h_{\mathcal{M},\mathcal{A}}(s) = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d)\cdot\mathrm{wt}(r)\right)$$

$$\text{(output product)} = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}\triangleright\mathcal{A}}:\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\right)$$

summary:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$
- sentence s of SL

noisy channel:

$$h_{\mathcal{M},\mathcal{A}}(s) = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d)\cdot\mathrm{wt}(r)\right)$$

$$\text{(output product)} \quad = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}\triangleright\mathcal{A}}:\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\right)$$

$$\text{(B-H,S,P; input product)} \quad = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{d\in D_{\mathcal{A}_s\triangleleft(\mathcal{M}\triangleright\mathcal{A})}} \mathrm{wt}(d)\right)$$

summary:

- model for translation from SL to TL: wtt $\mathcal{M}$
- model for TL: wta $\mathcal{A}$
- sentence s of SL

noisy channel:

$$h_{\mathcal{M},\mathcal{A}}(s) = \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{(d,r)\in Y:\\ \pi_{\mathrm{SL}}(d,r)=s}} \mathrm{wt}(d)\cdot\mathrm{wt}(r)\right)$$

(output product) $$= \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{\substack{d\in D_{\mathcal{M}\triangleright\mathcal{A}}:\\ \pi_{\mathrm{SL}}(d)=s}} \mathrm{wt}(d)\right)$$

(B-H,S,P; input product) $$= \pi_{\mathrm{TL}}\left(\operatorname{argmax}_{d\in D_{\mathcal{A}_s\triangleleft(\mathcal{M}\triangleright\mathcal{A})}} \mathrm{wt}(d)\right)$$

$$= \pi_{\mathrm{TL}}\big(\,\mathrm{Knuth}(\mathcal{A}_s\triangleleft(\mathcal{M}\triangleright\mathcal{A}))\,\big)$$

weighted tree automata and weighted tree transducers

can help in modelling and decoding of

statistical machine translation of natural languages

weighted tree automata and weighted tree transducers

can help in modelling and decoding of

statistical machine translation of natural languages

but: SMT is an engineering task!

weighted tree automata and weighted tree transducers

can help in modelling and decoding of

statistical machine translation of natural languages

conceptually

but: SMT is an engineering task!

References:

- [Baker 79] Composition of top-down and bottom-up tree transductions
- [Bar-Hillel, Shamir, Perles 61] On formal properties of simple phrase structure grammars
- [Büchse, Nederhof, V. 11] Tree Parsing with Synchronous Tree-Adjoining Grammars
- [Engelfriet, Fülöp, V. 02] Bottom-up and Top-down Tree Series Transformations
- [Fülöp, V. 09] Weighted tree automata and tree transducers
- [Knight et al. 03-...] ...
- [Knuth 77] A generalization of Dijkstra's algorithm
- [Lopez 08] Statistical Machine Translation
- [Liang, Bouchard-Côté, Klein, Taskar 06] An End-to-End Discriminative Approach to Machine Translation
- [Maletti 06] Compositions of Tree Series Transformations
- [Maletti, Satta 09] Parsing Algorithms based on Tree Automata
- [Nederhof, V. 12] Synchronous Context-Free Tree Grammars
- [Thatcher 67] Characterizing derivation trees of context-free grammars through a generalization of finite automata theory
- [Yamada, Knight 01] A syntax-based statistical translation model

[Knight et al. 03-...]

- [Charniak, Knight, Yamada 03] Syntax-based language models for statistical machine translation
- [Galley, Hopkins, Knight, Marcu 04] What's in a translation rule?
- [Graehl, Knight 04] Training tree transducers
- [Graehl, Knight 05] An overview of probabilistic tree transducers for natural language processing

[Knight et al. 03-...]

- [Charniak, Knight, Yamada 03] Syntax-based language models for statistical machine translation
- [Galley, Hopkins, Knight, Marcu 04] What's in a translation rule?
- [Graehl, Knight 04] Training tree transducers
- [Graehl, Knight 05] An overview of probabilistic tree transducers for natural language processing

[Maletti 11]
Survey: Weighted Extended Top-down Tree Transducers –
Part III: Applications in Machine Translation

[Knight et al. 03-...]

- [Charniak, Knight, Yamada 03] Syntax-based language models for statistical machine translation
- [Galley, Hopkins, Knight, Marcu 04] What's in a translation rule?
- [Graehl, Knight 04] Training tree transducers
- [Graehl, Knight 05] An overview of probabilistic tree transducers for natural language processing

[Maletti 11]
Survey: Weighted Extended Top-down Tree Transducers –
Part III: Applications in Machine Translation

Thank you!