Lecture 8: Rule extraction and training

Andreas Maletti

Statistical Machine Translation

Stuttgart — January 13, 2012

Lecture 8

Last time

- Basic setup
- Tree automata

This time

- Synchronous grammars (tree transducers)
- Rule extraction
- Weight training (EM)

Contents

Weight structure

Definition

Commutative semiring $(C, +, \cdot, 0, 1)$ if

- $(C, +, 0)$ and $(C, \cdot, 1)$ commutative monoids
- $\cdot$ distributes over finite (incl. empty) sums

Example

- BOOLEAN semiring $(\{0, 1\}, \max, \min, 0, 1)$
- Semiring $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$ of probabilities
- Tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$
- Any field, ring, etc.

Most of the lecture: BOOLEAN semiring (rule extraction) or probabilities

Weight structure

Definition

Commutative semiring $(C, +, \cdot, 0, 1)$ if

- $(C, +, 0)$ and $(C, \cdot, 1)$ commutative monoids
- $\cdot$ distributes over finite (incl. empty) sums

Example

- BOOLEAN semiring $(\{0, 1\}, \max, \min, 0, 1)$
- Semiring $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$ of probabilities
- Tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$
- Any field, ring, etc.

Most of the lecture: BOOLEAN semiring (rule extraction) or probabilities

Weight structure

Definition

Commutative semiring $(C, +, \cdot, 0, 1)$ if

- $(C, +, 0)$ and $(C, \cdot, 1)$ commutative monoids
- $\cdot$ distributes over finite (incl. empty) sums

Example

- BOOLEAN semiring $(\{0, 1\}, \max, \min, 0, 1)$
- Semiring $(\mathbb{R}_{\geq 0}, +, \cdot, 0, 1)$ of probabilities
- Tropical semiring $(\mathbb{N} \cup \{\infty\}, \min, +, \infty, 0)$
- Any field, ring, etc.

Most of the lecture: BOOLEAN semiring (rule extraction) or probabilities

Computing in semirings

Distributivity

$$a \cdot \left(\sum_{i \in I} a_i \right) = \sum_{i \in I} (a \cdot a_i)$$

Absorption

$0 \cdot a = 0$

Questions

- $1 + a = 1$?
- $a + (\prod_{i \in I} a_i) = \prod_{i \in I} (a + a_i)$?
- $a \cdot b = a \cdot c$ with $a \neq 0$ yields $b = c$?

Computing in semirings

Distributivity

$$a \cdot \left(\sum_{i \in I} a_i\right) = \sum_{i \in I} (a \cdot a_i)$$

Absorption

$0 \cdot a = 0$

Questions

- $1 + a = 1$?
- $a + \left(\prod_{i \in I} a_i\right) = \prod_{i \in I} (a + a_i)$?
- $a \cdot b = a \cdot c$ with $a \neq 0$ yields $b = c$?

Computing in semirings

Distributivity

$$a \cdot \left(\sum_{i \in I} a_i\right) = \sum_{i \in I} (a \cdot a_i)$$

Absorption

$0 \cdot a = 0$

Questions

- $1 + a = 1$? NO!
- $a + \left(\prod_{i \in I} a_i\right) = \prod_{i \in I} (a + a_i)$?
- $a \cdot b = a \cdot c$ with $a \neq 0$ yields $b = c$?

Computing in semirings

Distributivity

$$a \cdot \left(\sum_{i \in I} a_i \right) = \sum_{i \in I} (a \cdot a_i)$$

Absorption

$0 \cdot a = 0$

Questions

- $1 + a = 1$? NO!
- $a + \left(\prod_{i \in I} a_i \right) = \prod_{i \in I} (a + a_i)$? NO!
- $a \cdot b = a \cdot c$ with $a \neq 0$ yields $b = c$?

Computing in semirings

Distributivity

$$a \cdot \left(\sum_{i \in I} a_i \right) = \sum_{i \in I} (a \cdot a_i)$$

Absorption

$0 \cdot a = 0$

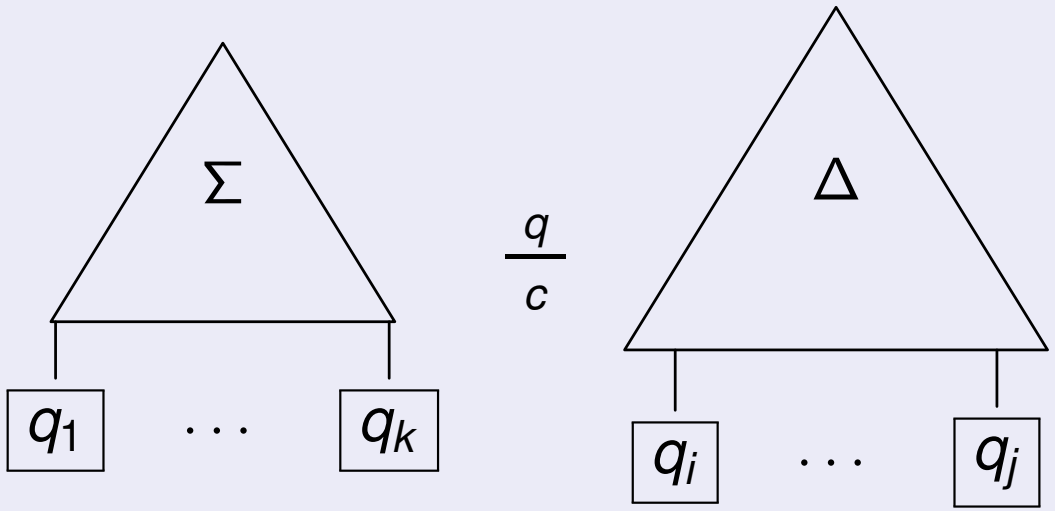
Questions

- $1 + a = 1$? NO!
- $a + \left(\prod_{i \in I} a_i \right) = \prod_{i \in I} (a + a_i)$? NO!
- $a \cdot b = a \cdot c$ with $a \neq 0$ yields $b = c$? NO!

Syntax

Definition (Arnold, Dauchet 1976, Graehl, Knight 2004)

Extended top-down tree transducer (XTOP) $M = (Q, \Sigma, \Delta, I, R)$ with finitely many rules

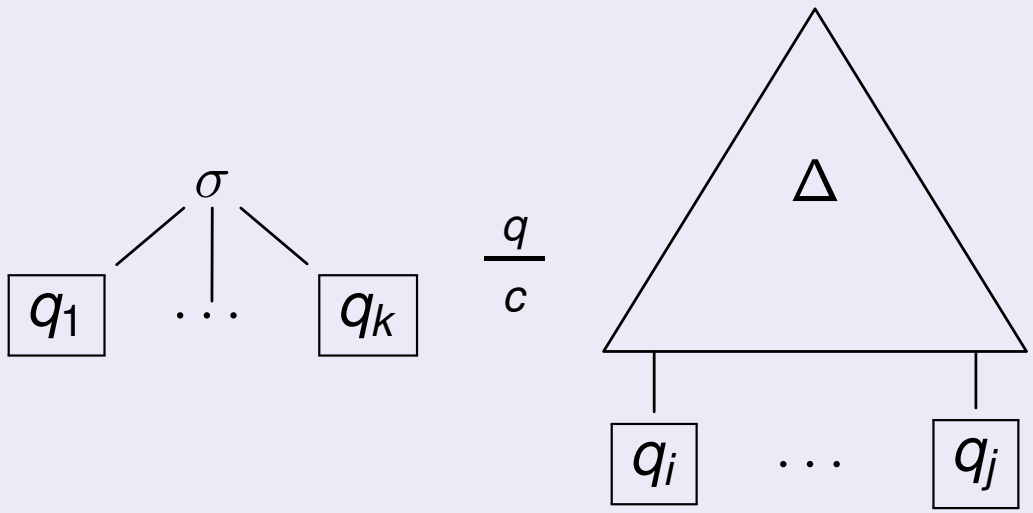


- $q, q_1, \ldots, q_k \in Q$ are states
- $c \in C$ (weight)

Syntax (cont'd)

Definition (Rounds 1970, Thatcher 1970)

Top-down tree transducer (TOP) if all rules



Remarks

- TOP well-studied device
- XTOP much less so, but gained a lot of traction

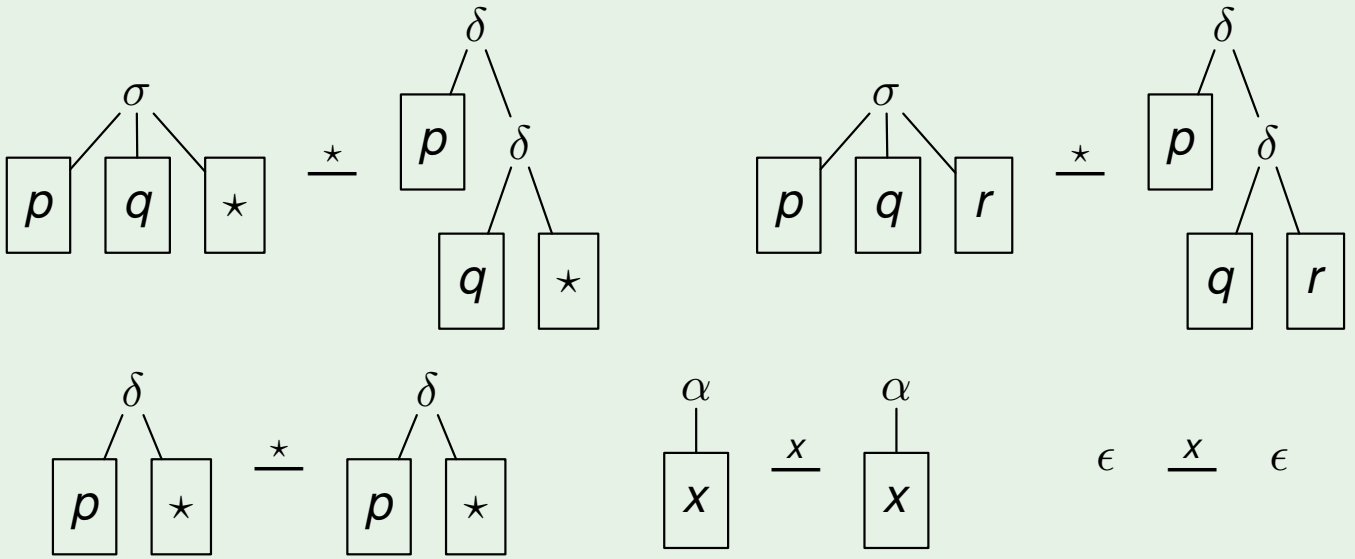
Extended Top-down Tree Transducer

Definition

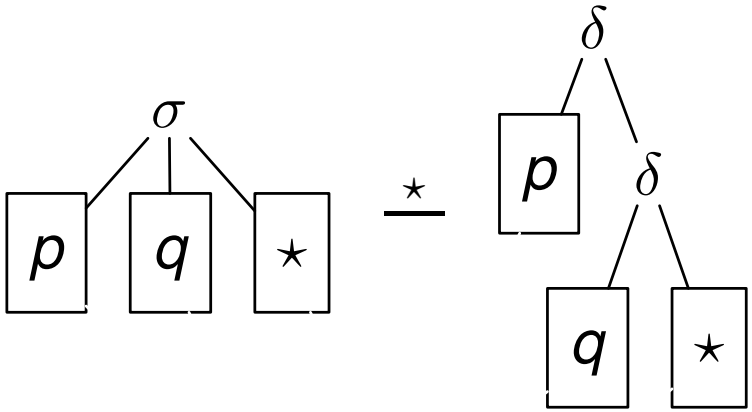
Each rule now has an input and an output side, which are both full trees (not just a single symbol followed by states)

Example

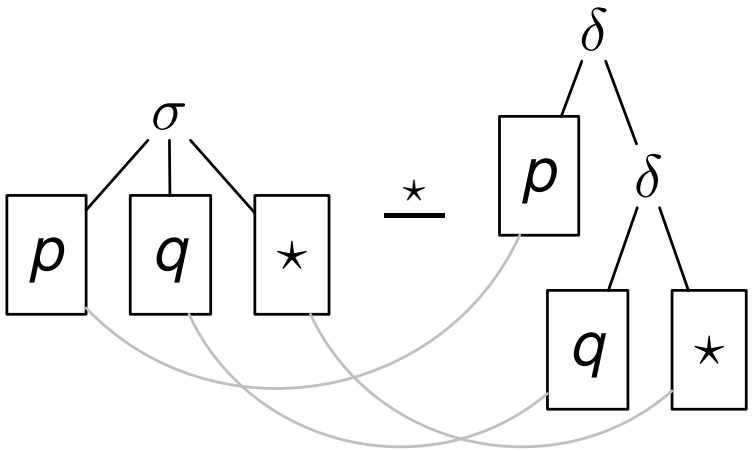
We ignore weights for the moment.



Link Structure

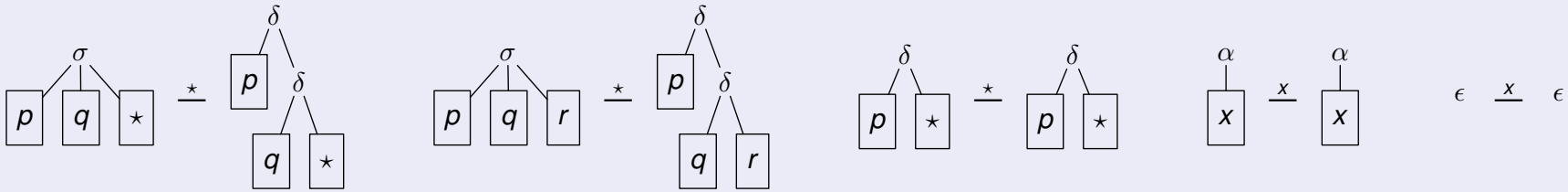


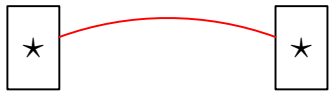
Link Structure



Derivation

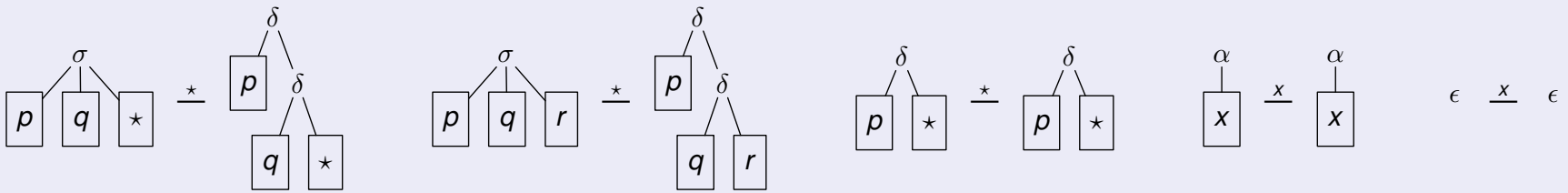
Rules

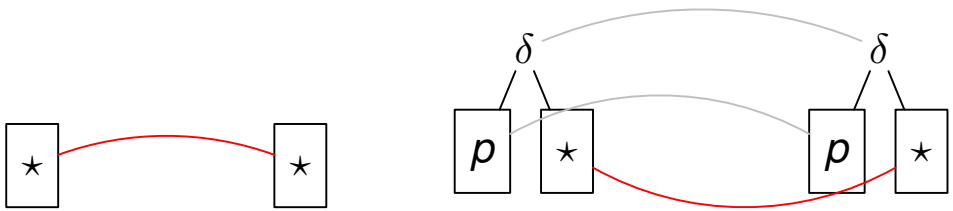




Derivation

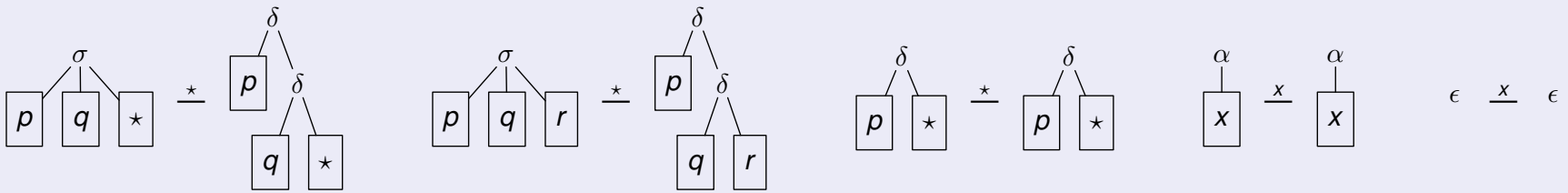
Rules

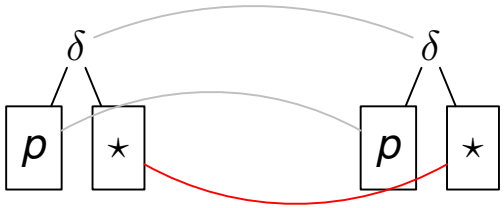


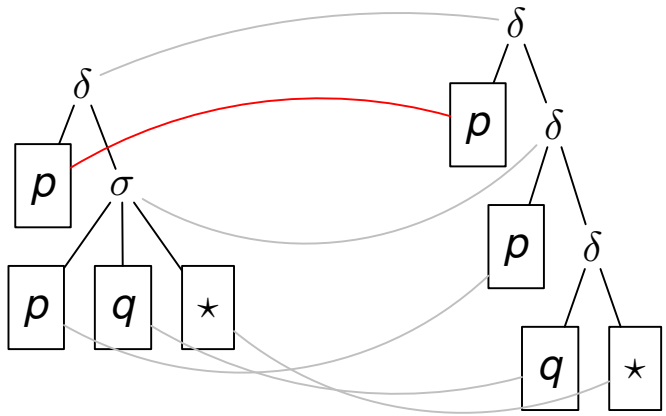


Derivation

Rules

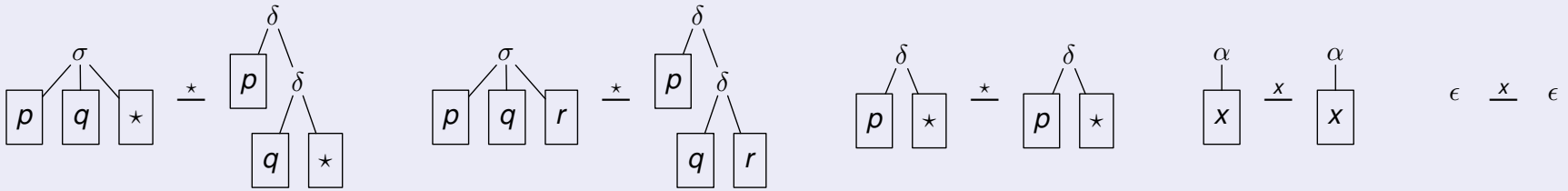


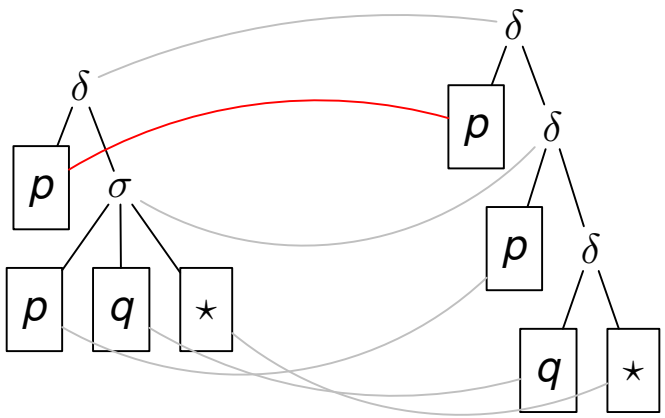


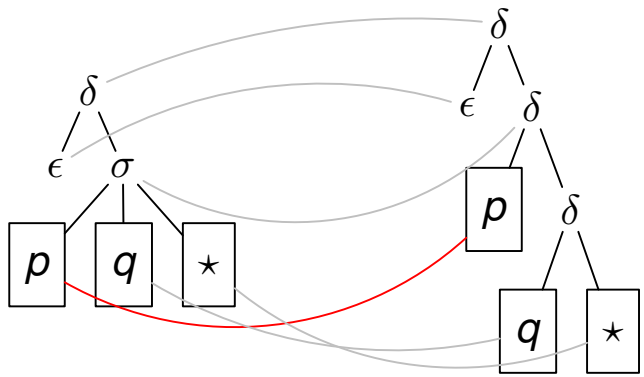


Derivation

Rules

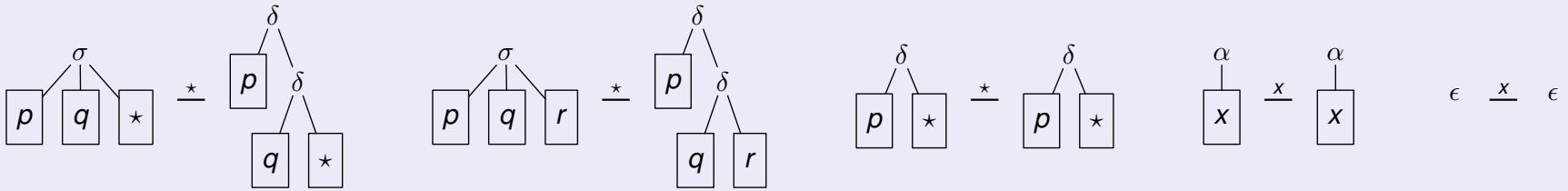


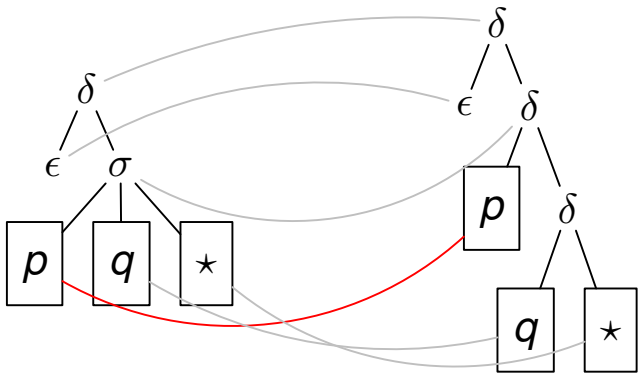


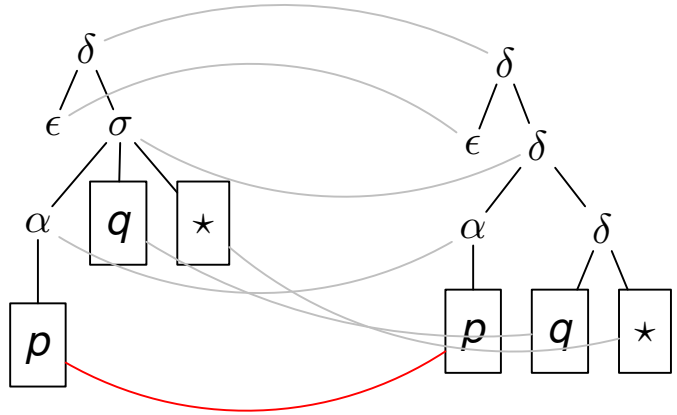


Derivation

Rules

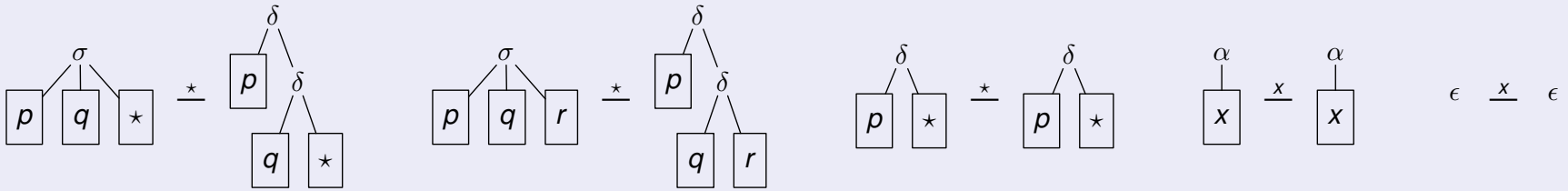


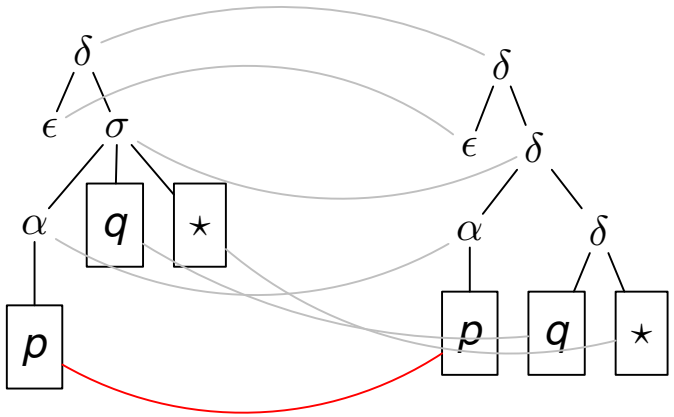


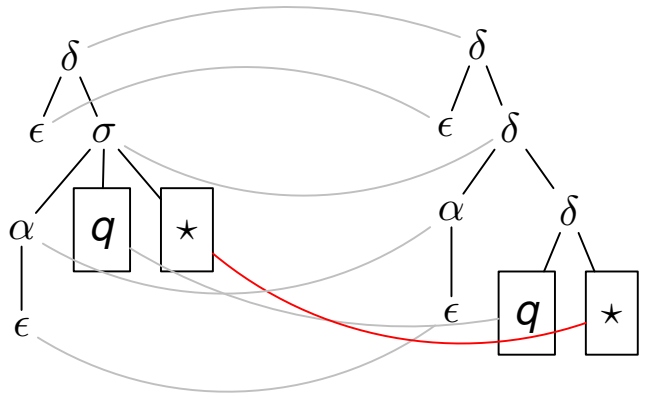


Derivation

Rules

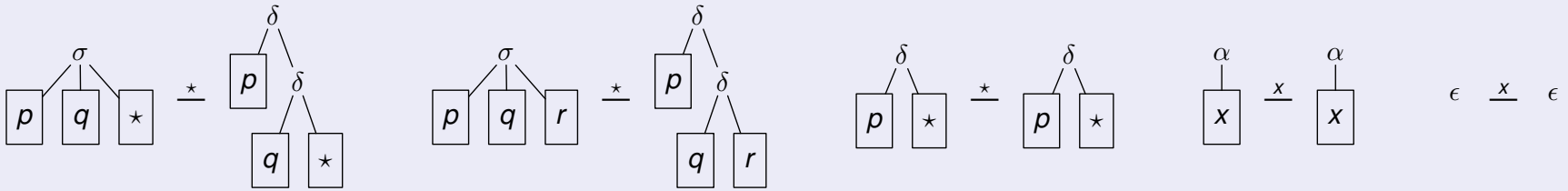


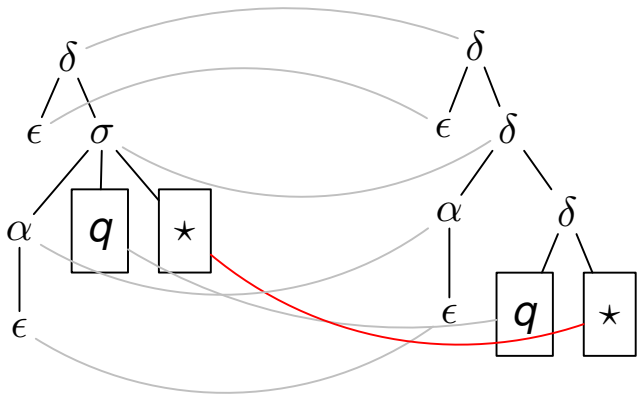


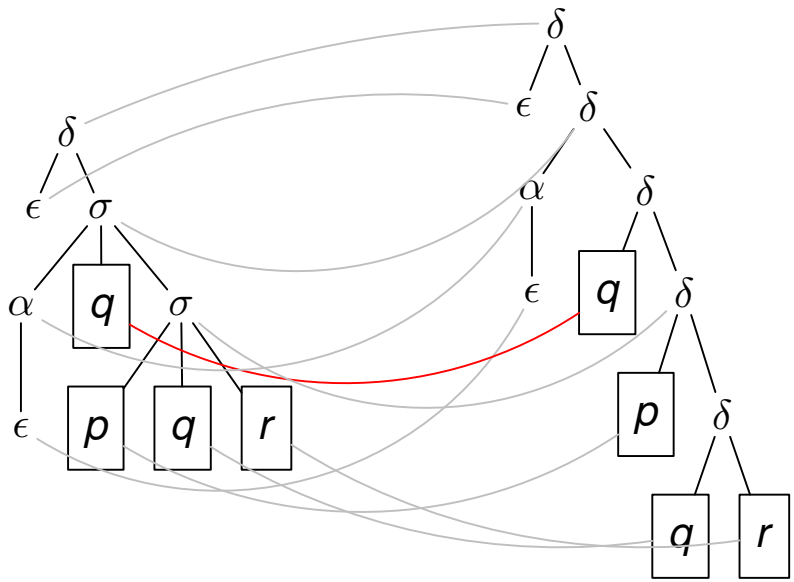


Derivation

Rules

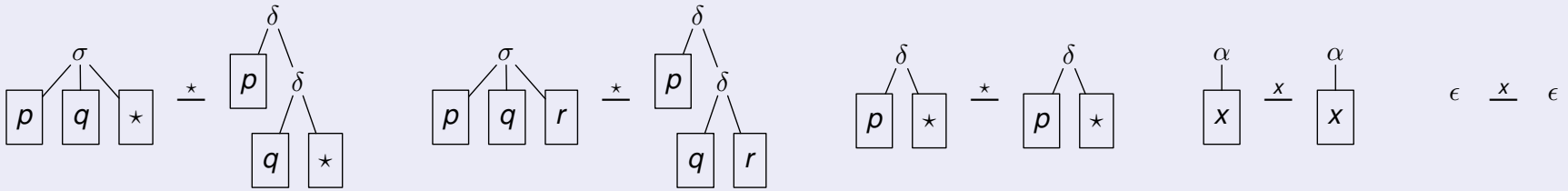


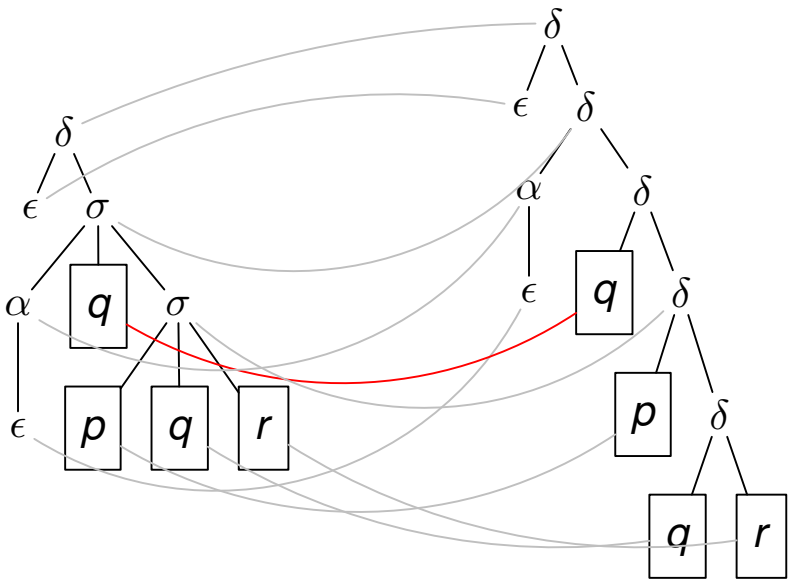


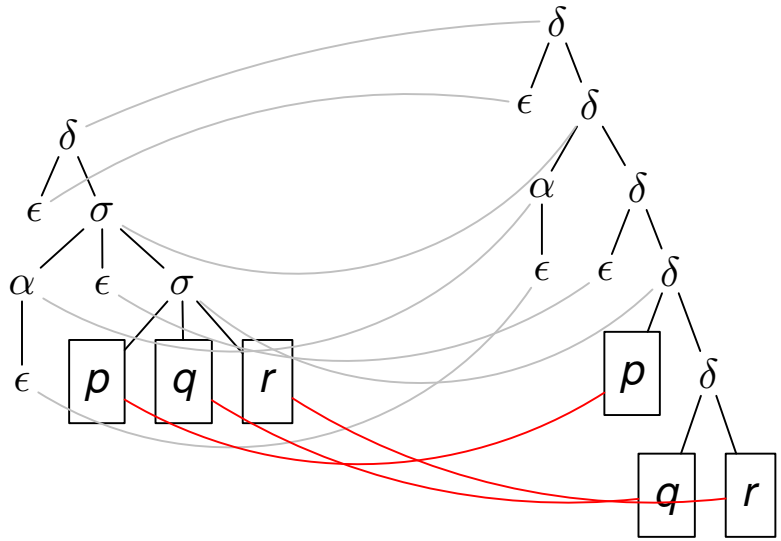


Derivation

Rules

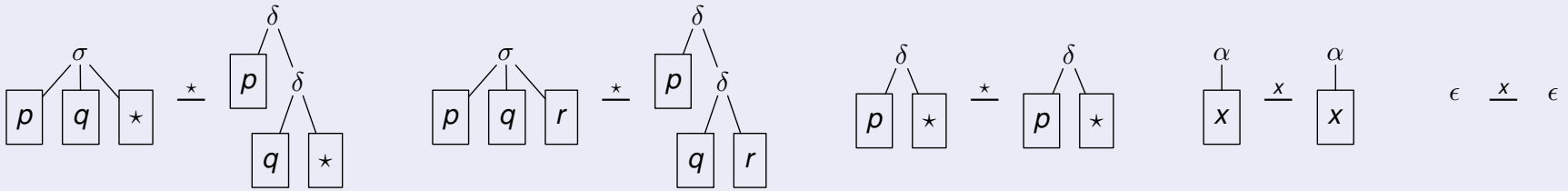


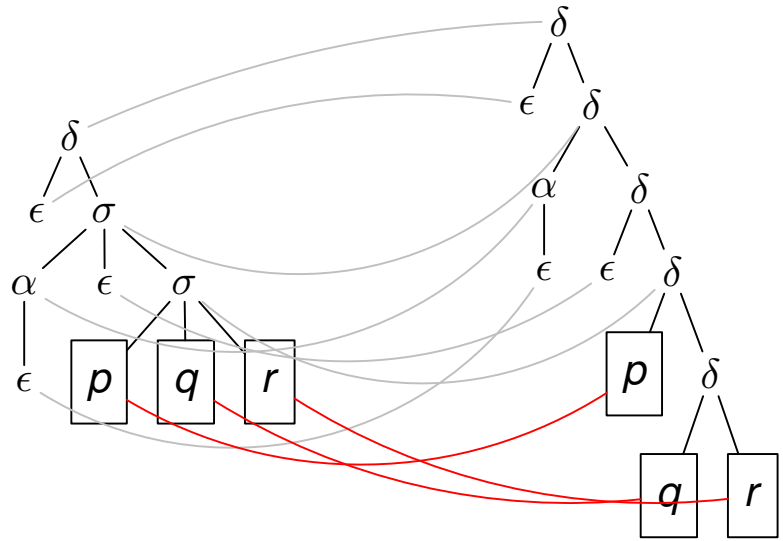


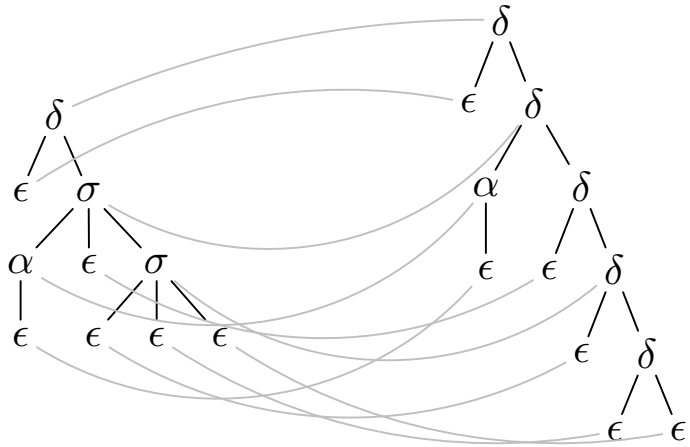


Derivation

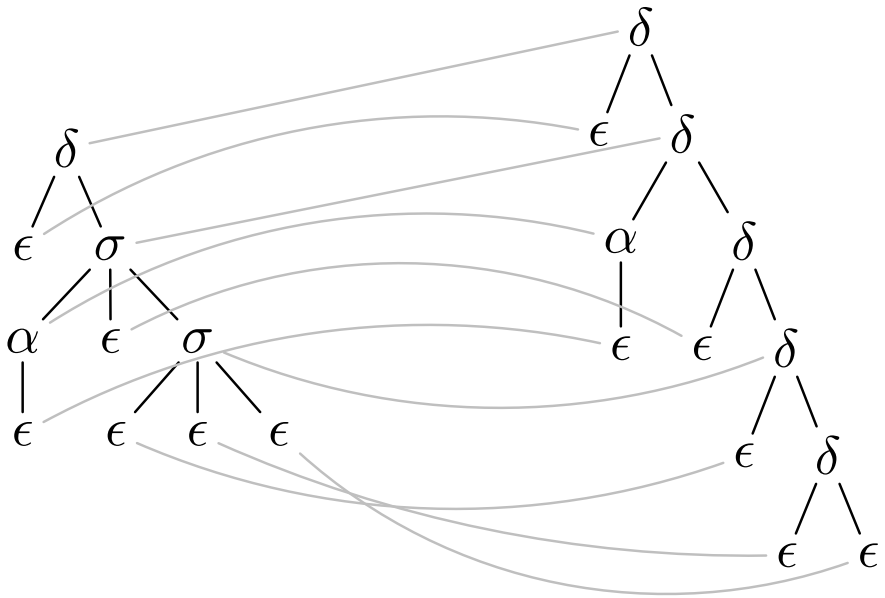
Rules



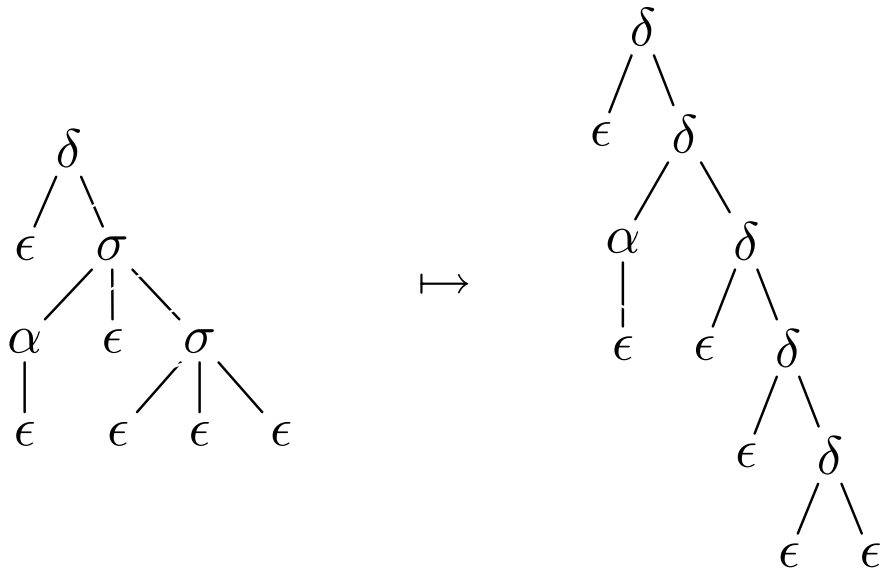




Semantics of XTOP



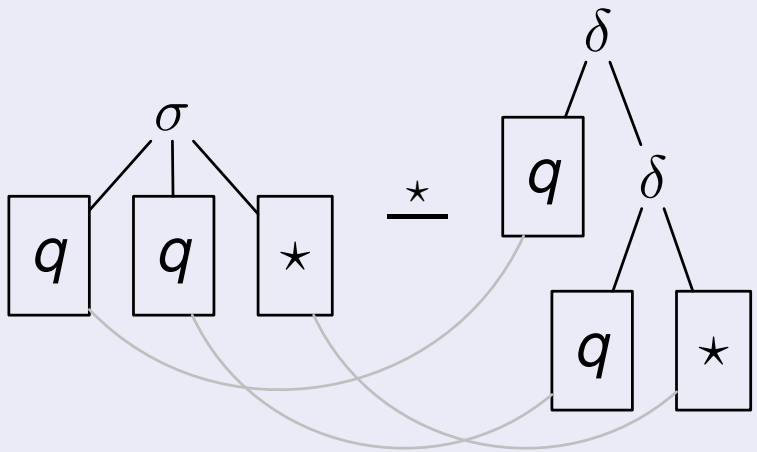
Semantics of XTOP



Comprehension

Question

Suppose we have the following (more general) link structure:

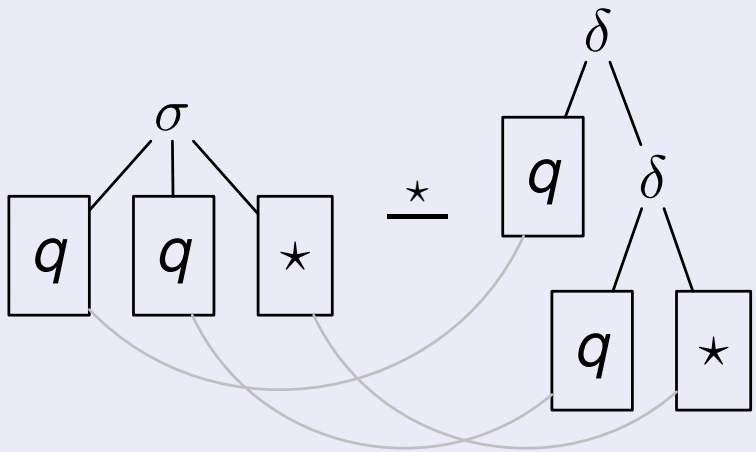


Can we implement it with an XTOP?

Comprehension

Question

Suppose we have the following (more general) link structure:



Can we implement it with an XTOP?

Answer

Yes. Copy all rules for q with a new head state p.
Replace one q consistently by p.

Contents

Rule extraction

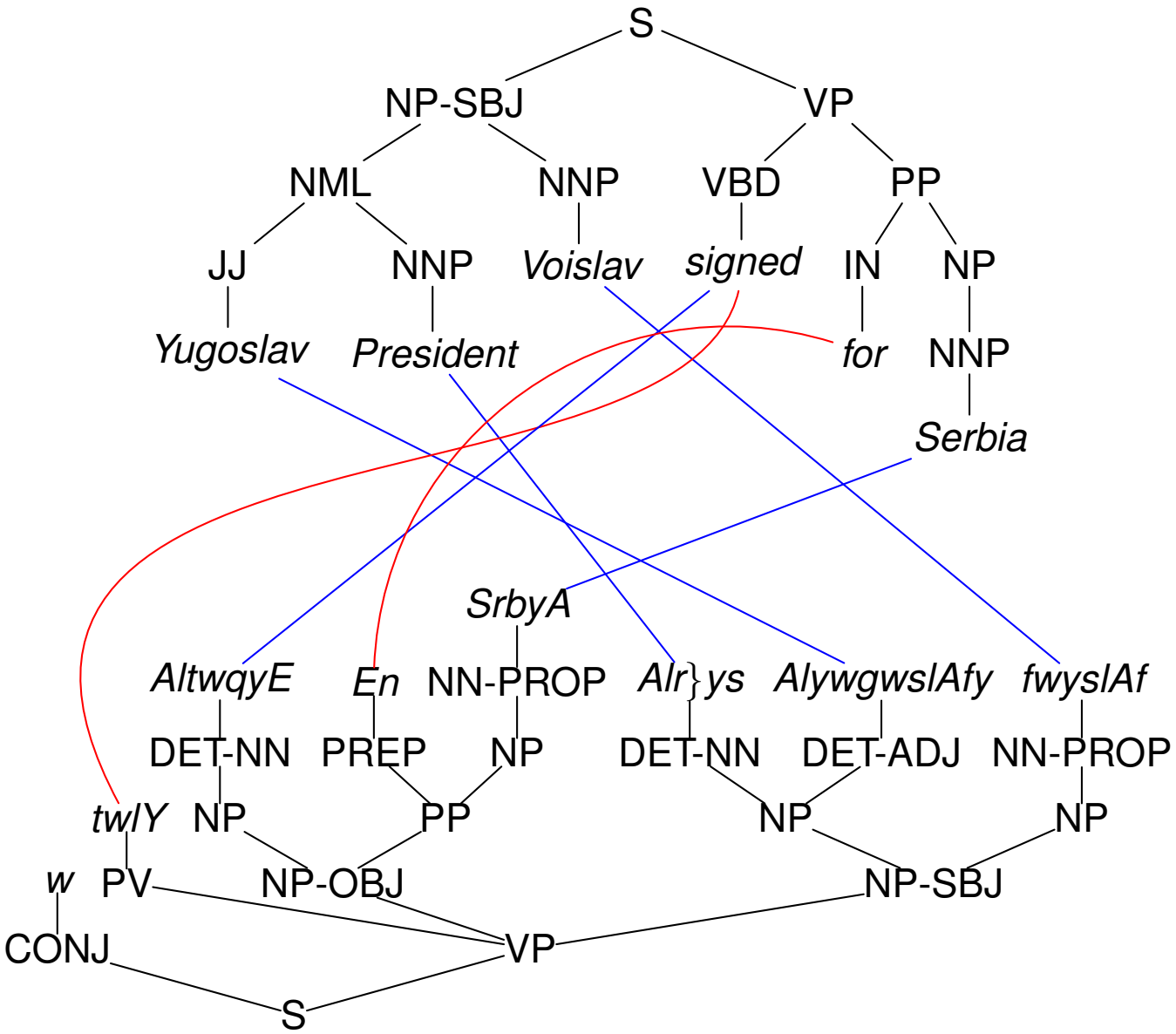
Goal

- automatic rule generation for XTOP
- only “useful” rules

Requirements

- bi-text
- parses for both sides
- string alignment (GIZA++)

Rule extraction



Rule extraction

Additional real requirements

- good alignments (use alignment correction / completion)
- side conditions (typical: POS never without lexical item)

Approach

- bottom-up procedure
- extend alignments to parse trees
- extract aligned subtrees

Definition

Two subtrees *t* and *u* are consistently aligned if all alignments from nodes of *t* align to nodes inside *u* and vice versa.

Rule extraction

Additional real requirements

- good alignments (use alignment correction / completion)
- side conditions (typical: POS never without lexical item)

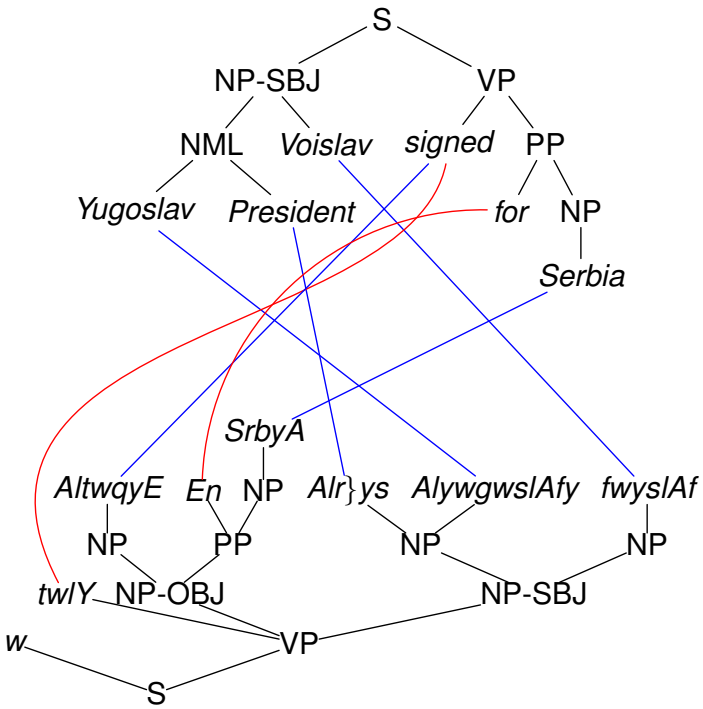
Approach

- bottom-up procedure
- extend alignments to parse trees
- extract aligned subtrees

Definition

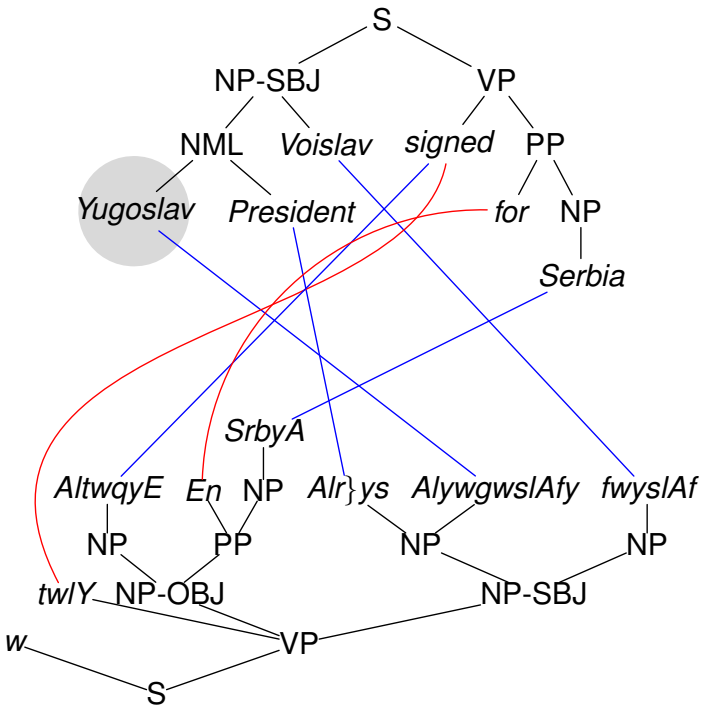
Two subtrees *t* and *u* are consistently aligned if all alignments from nodes of *t* align to nodes inside *u* and vice versa.

Rule extraction



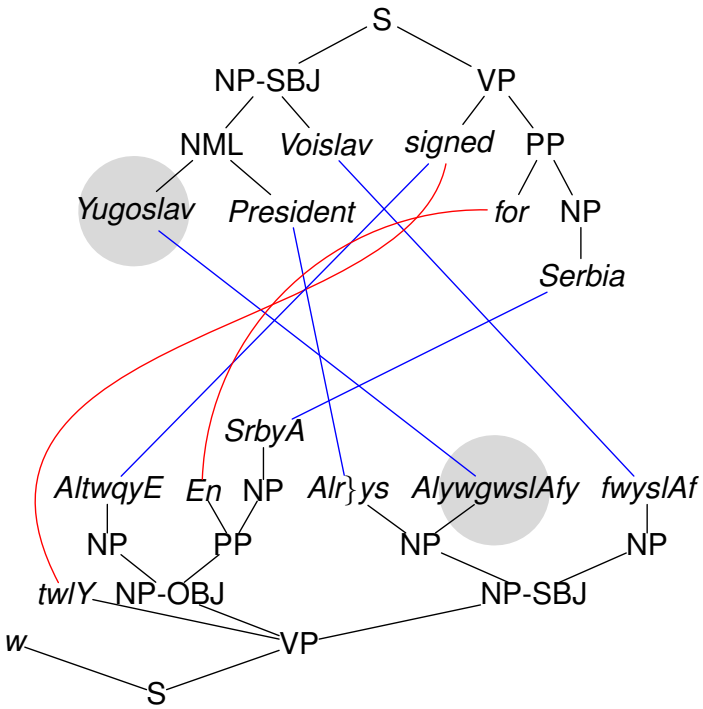
- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

Rule extraction



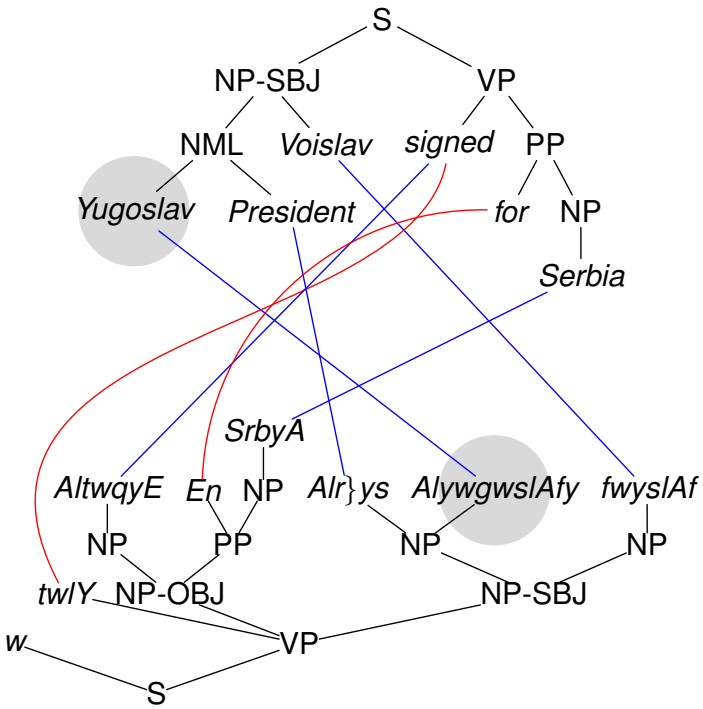
- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

Rule extraction



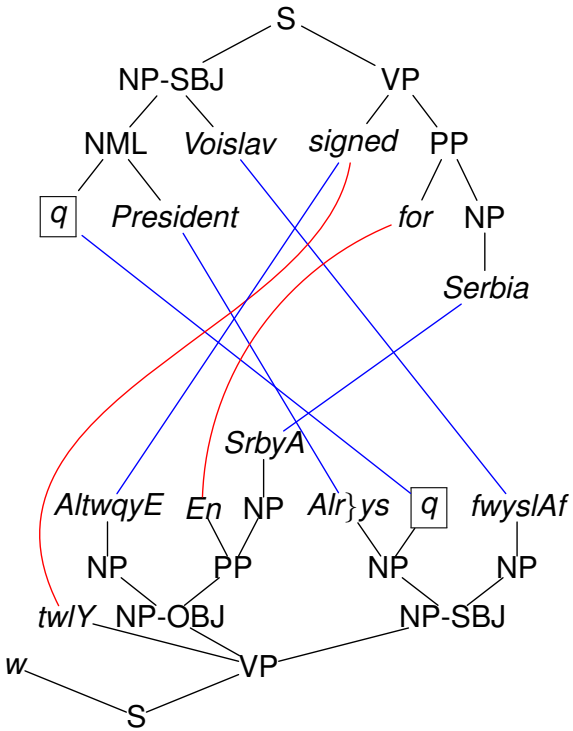
- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

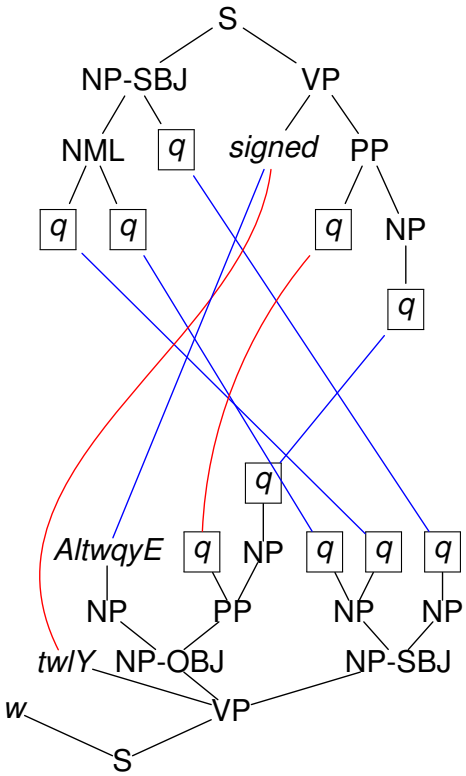
Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

$$\textit{Yugoslav} \underline{\quad}^{q} \textit{AlywgwslAfy}$$

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

$$\textit{Yugoslav} \xrightarrow{q} \textit{AlywgwslAfy}$$

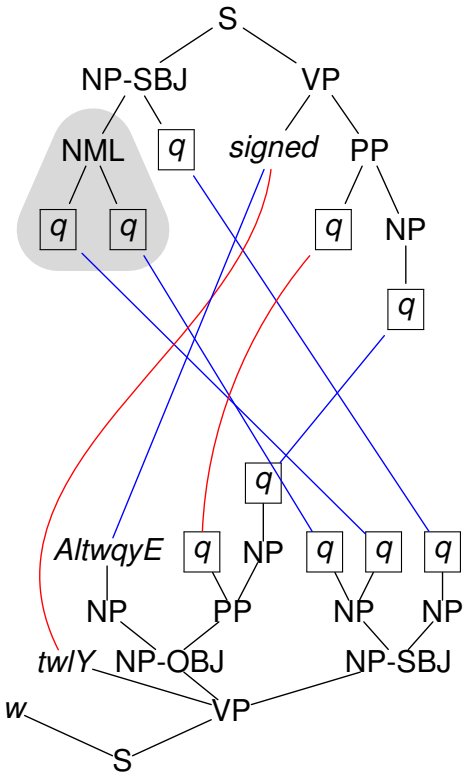
$$\textit{President} \xrightarrow{q} \textit{Alr}\}\textit{ys}$$

$$\textit{Voislav} \xrightarrow{q} \textit{fwyslAf}$$

$$\textit{for} \xrightarrow{q} \textit{En}$$

$$\textit{Serbia} \xrightarrow{q} \textit{SrbyA}$$

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

$$\textit{Yugoslav} \xrightarrow{q} \textit{AlywgwslAfy}$$

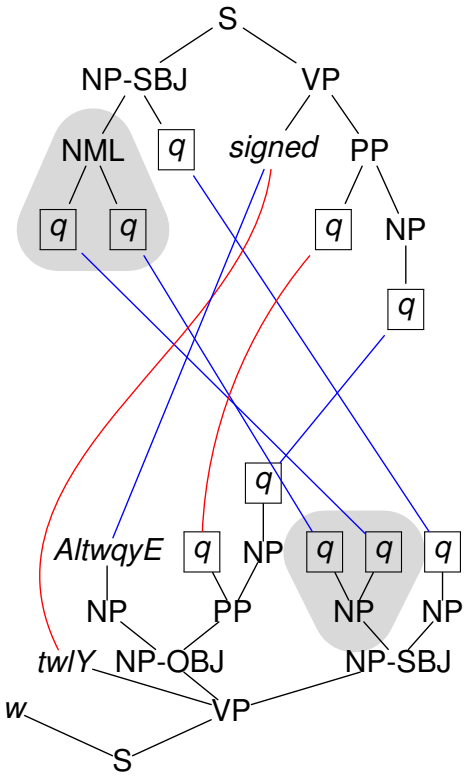
$$\textit{President} \xrightarrow{q} \textit{Alr}\}\textit{ys}$$

$$\textit{Voislav} \xrightarrow{q} \textit{fwyslAf}$$

$$\textit{for} \xrightarrow{q} \textit{En}$$

$$\textit{Serbia} \xrightarrow{q} \textit{SrbyA}$$

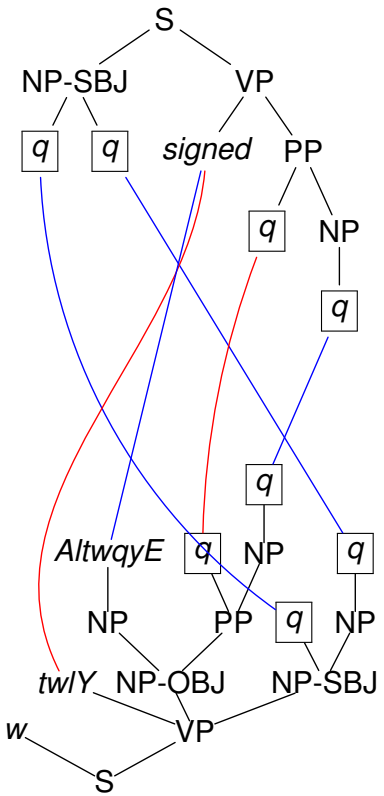
Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

$$\mathrm{NML}(q_1, q_2) \xrightarrow{q} \mathrm{NP}(q_2, q_1)$$

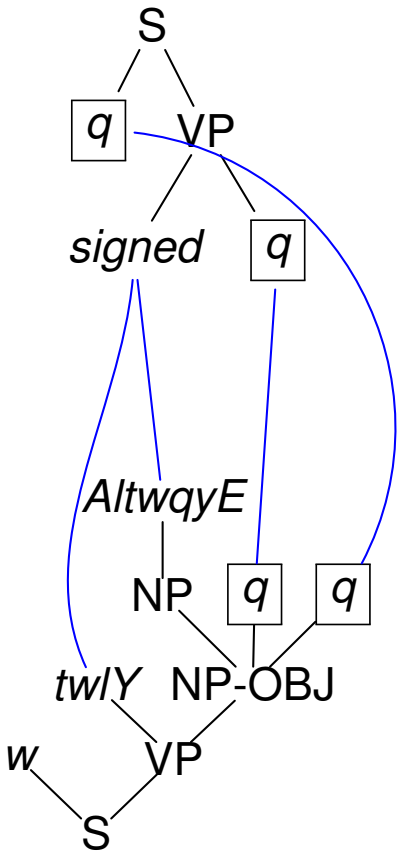
Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

$$\mathrm{NML}(q_1, q_2) \xrightarrow{q} \mathrm{NP}(q_2, q_1)$$

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

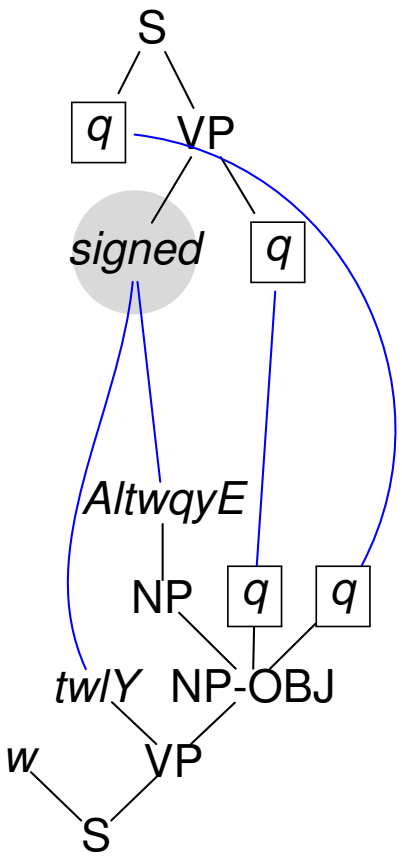
$$\mathrm{NML}(q_1, q_2) \xrightarrow{q} \mathrm{NP}(q_2, q_1)$$

$$\mathrm{NP}(q) \xrightarrow{q} \mathrm{NP}(q)$$

$$\mathrm{PP}(q_1, q_2) \xrightarrow{q} \mathrm{PP}(q_1, q_2)$$

$$\mathrm{NP\text{-}SBJ}(q_1, q_2) \xrightarrow{q} \mathrm{NP\text{-}SBJ}(q_1, \mathrm{NP}(q_2))$$

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

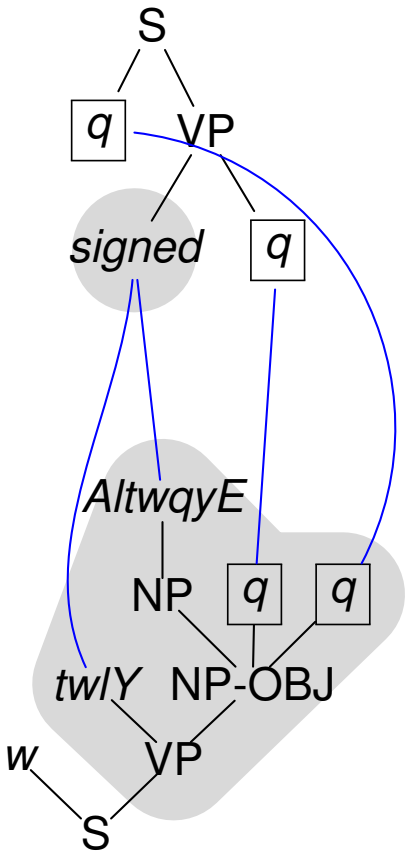
$$\mathrm{NML}(q_1, q_2) \xrightarrow{q} \mathrm{NP}(q_2, q_1)$$

$$\mathrm{NP}(q) \xrightarrow{q} \mathrm{NP}(q)$$

$$\mathrm{PP}(q_1, q_2) \xrightarrow{q} \mathrm{PP}(q_1, q_2)$$

$$\mathrm{NP\text{-}SBJ}(q_1, q_2) \xrightarrow{q} \mathrm{NP\text{-}SBJ}(q_1, \mathrm{NP}(q_2))$$

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

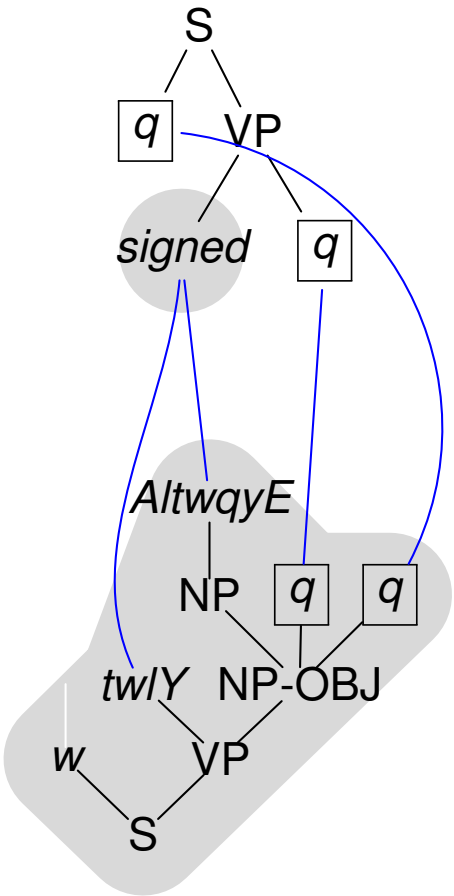
$$\mathrm{NML}(q_1, q_2) \xrightarrow{q} \mathrm{NP}(q_2, q_1)$$

$$\mathrm{NP}(q) \xrightarrow{q} \mathrm{NP}(q)$$

$$\mathrm{PP}(q_1, q_2) \xrightarrow{q} \mathrm{PP}(q_1, q_2)$$

$$\mathrm{NP\text{-}SBJ}(q_1, q_2) \xrightarrow{q} \mathrm{NP\text{-}SBJ}(q_1, \mathrm{NP}(q_2))$$

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

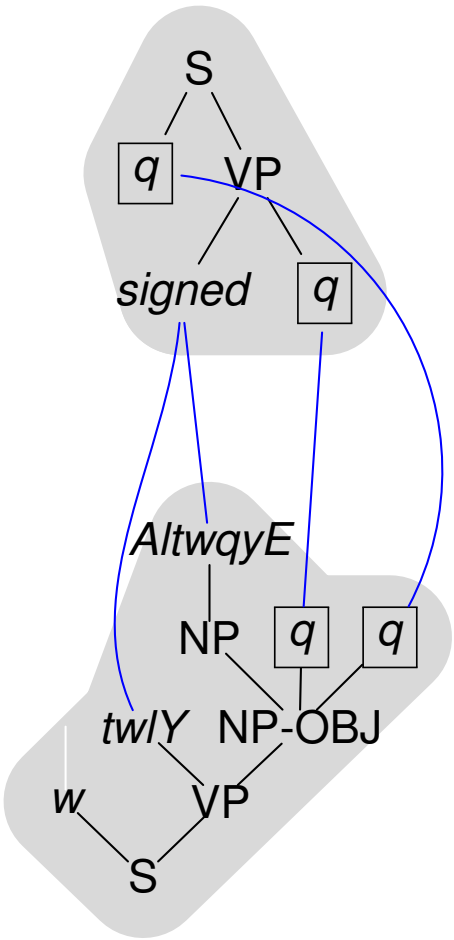
$$\mathrm{NML}(q_1, q_2) \xrightarrow{q} \mathrm{NP}(q_2, q_1)$$

$$\mathrm{NP}(q) \xrightarrow{q} \mathrm{NP}(q)$$

$$\mathrm{PP}(q_1, q_2) \xrightarrow{q} \mathrm{PP}(q_1, q_2)$$

$$\mathrm{NP\text{-}SBJ}(q_1, q_2) \xrightarrow{q} \mathrm{NP\text{-}SBJ}(q_1, \mathrm{NP}(q_2))$$

Rule extraction



- Select next node bottom-up
- Identify maximal subtree of aligned nodes
- Identify subtree of nodes aligned to aligned nodes, etc.
- Extract rule and leave state
- Repeat

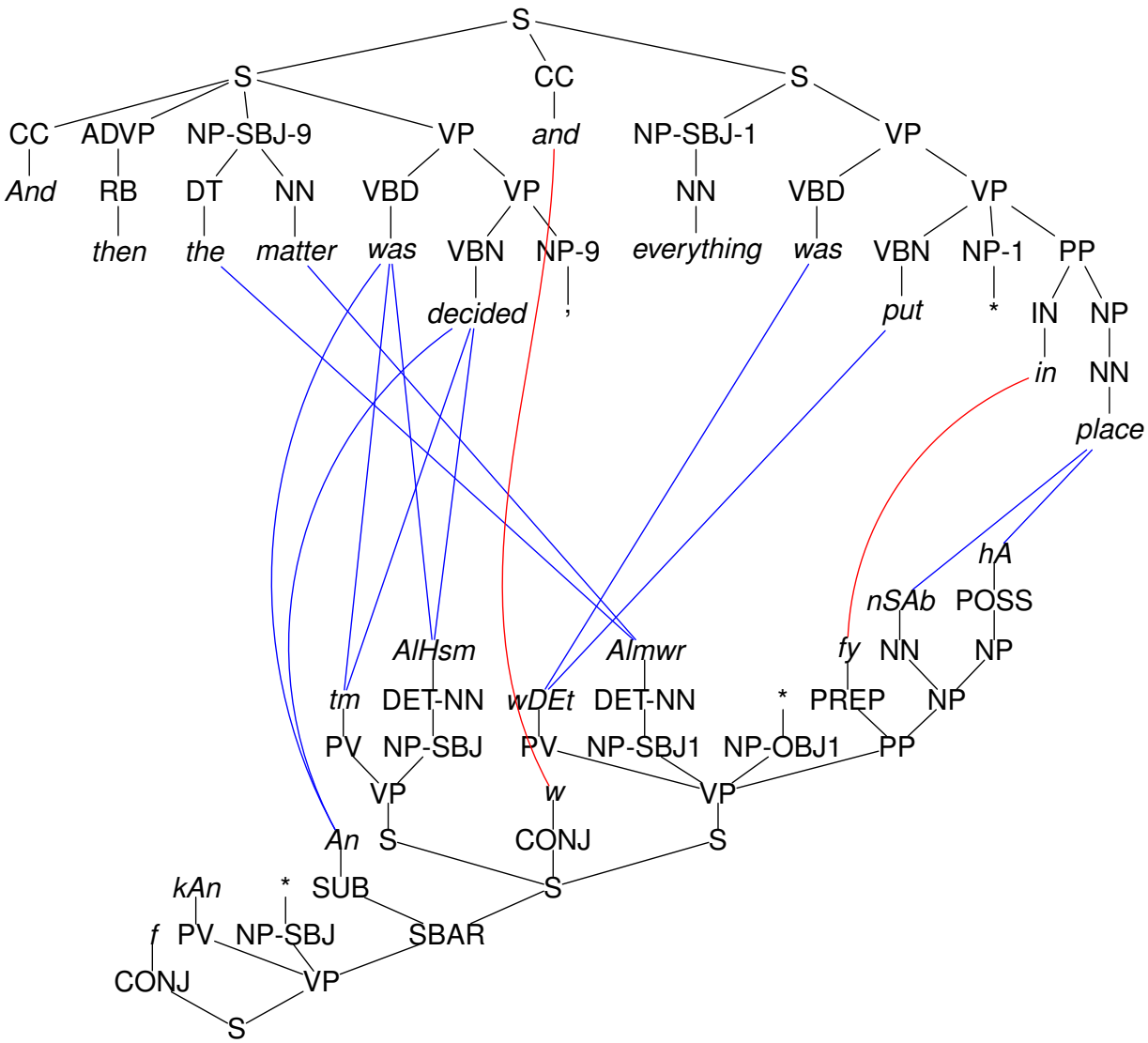
$$\mathrm{NML}(q_1, q_2) \xrightarrow{q} \mathrm{NP}(q_2, q_1)$$

$$\mathrm{NP}(q) \xrightarrow{q} \mathrm{NP}(q)$$

$$\mathrm{PP}(q_1, q_2) \xrightarrow{q} \mathrm{PP}(q_1, q_2)$$

$$\mathrm{NP\text{-}SBJ}(q_1, q_2) \xrightarrow{q} \mathrm{NP\text{-}SBJ}(q_1, \mathrm{NP}(q_2))$$

Rule extraction exercise



Contents

The problem

Input

- weighted bi-parsed bi-text T (our training corpus)
- unweighted XTOP M (the extracted XTOP)

Goal

- Obtain rule weight for each rule

The problem

Input

- weighted bi-parsed bi-text T (our training corpus)
- unweighted XTOP M (the extracted XTOP)

Goal

- Obtain rule weight for each rule

Weighted XTOP

How to handle weights?

- weights in a derivation are multiplied
- weights of derivations are summed

Derivation strategy

- given a partial derivation
- returns the next redex
- we need a deterministic one

Example

- Left-most derivation (deterministic)
- Right-most derivation (deterministic)
- Innermost derivation (non-deterministic)

Weighted XTOP

How to handle weights?

- weights in a derivation are multiplied
- weights of derivations are summed

Derivation strategy

- given a partial derivation
- returns the next redex
- we need a deterministic one

Example

- Left-most derivation (deterministic)
- Right-most derivation (deterministic)
- Innermost derivation (non-deterministic)

Weighted XTOP

How to handle weights?

- weights in a derivation are multiplied
- weights of derivations are summed

Derivation strategy

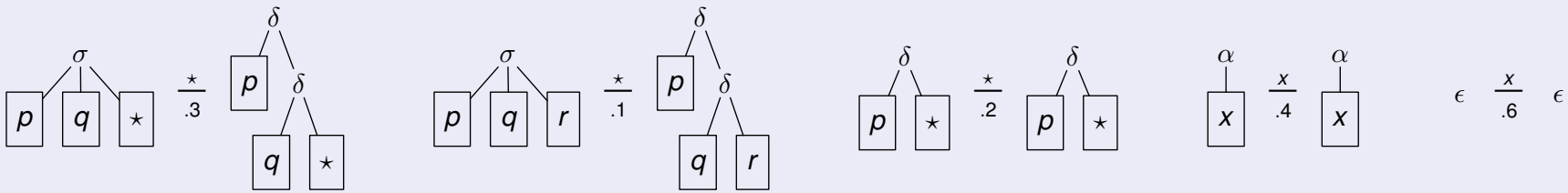
- given a partial derivation
- returns the next redex
- we need a deterministic one

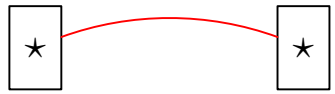
Example

- Left-most derivation (deterministic)
- Right-most derivation (deterministic)
- Innermost derivation (non-deterministic)

Derivation

Rules

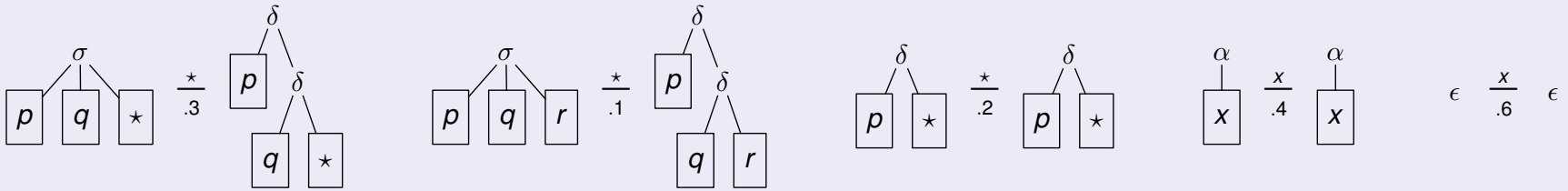


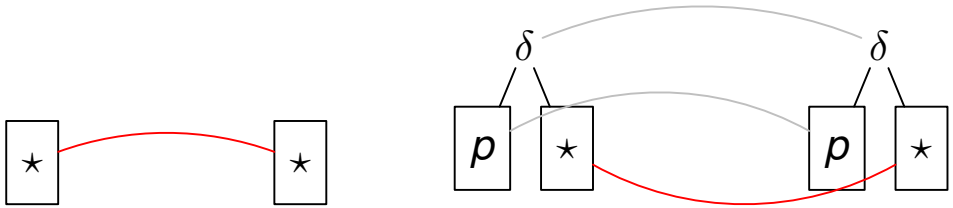


Derivation weight:

Derivation

Rules

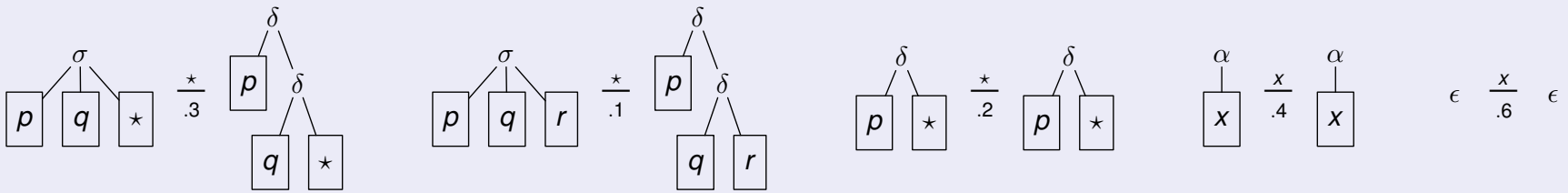


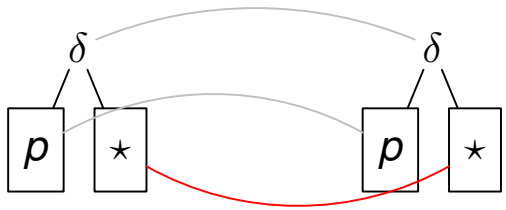


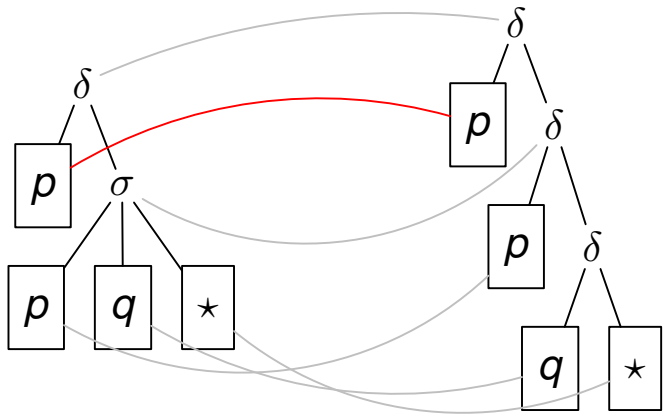
Derivation weight: 0.2

Derivation

Rules



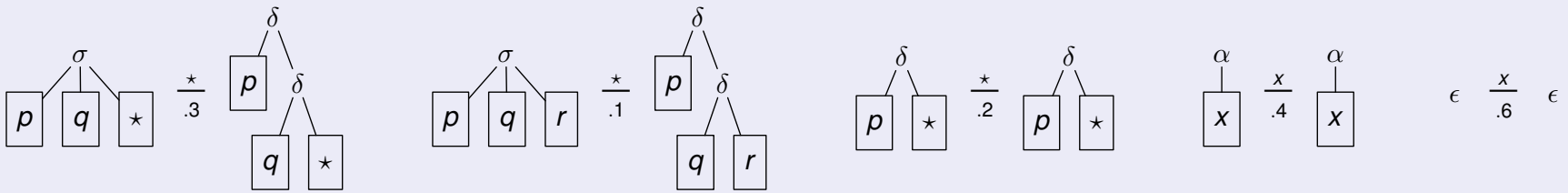


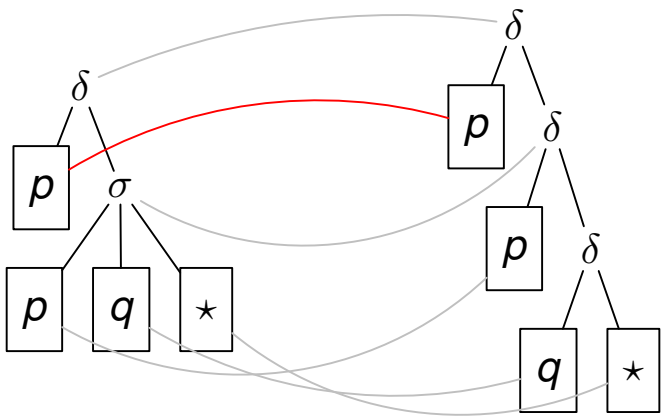


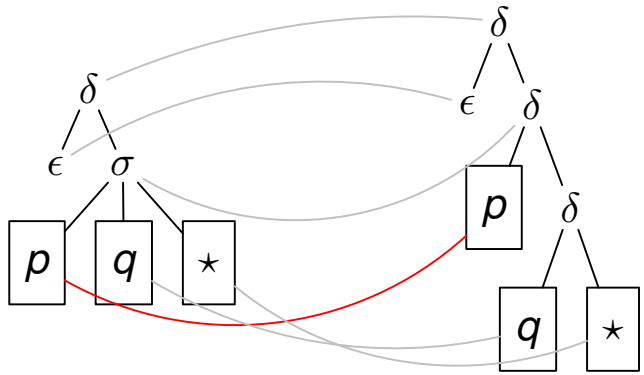
Derivation weight: $0.2 \cdot 0.3$

Derivation

Rules



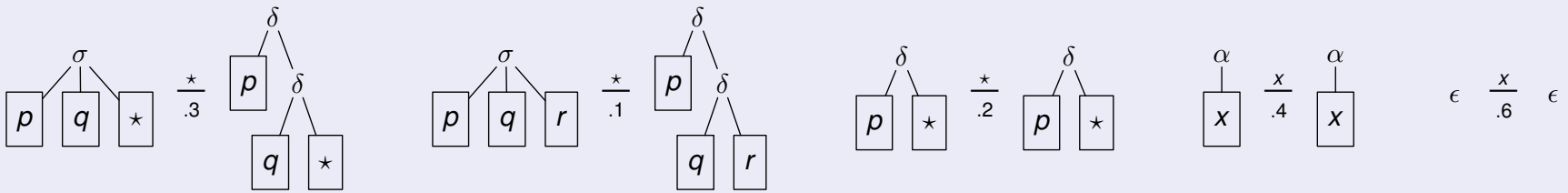


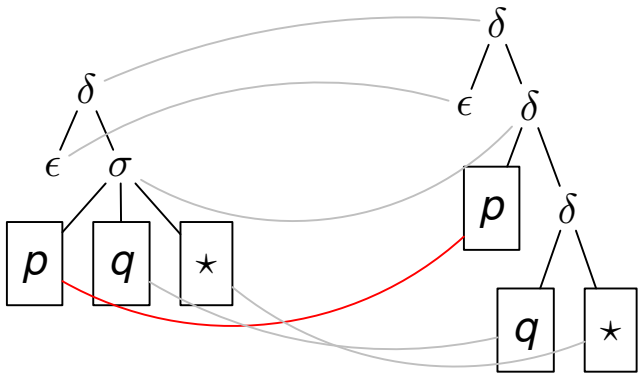


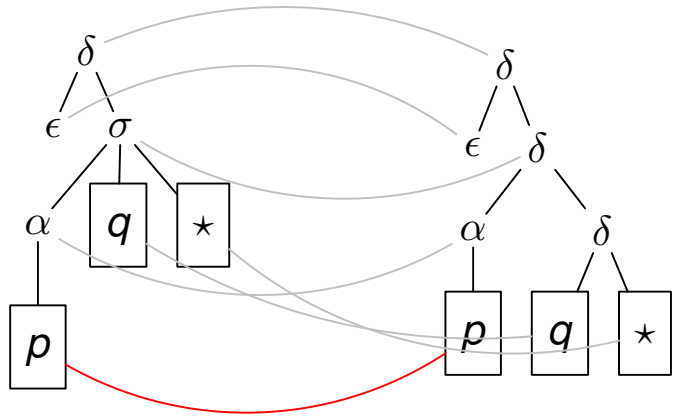
Derivation weight: $0.2 \cdot 0.3 \cdot 0.6$

Derivation

Rules



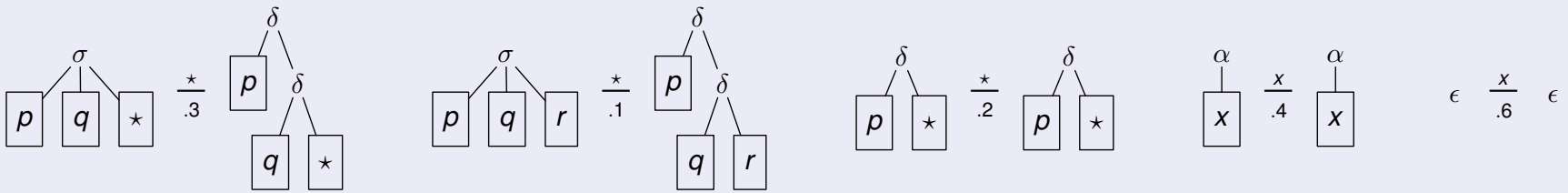


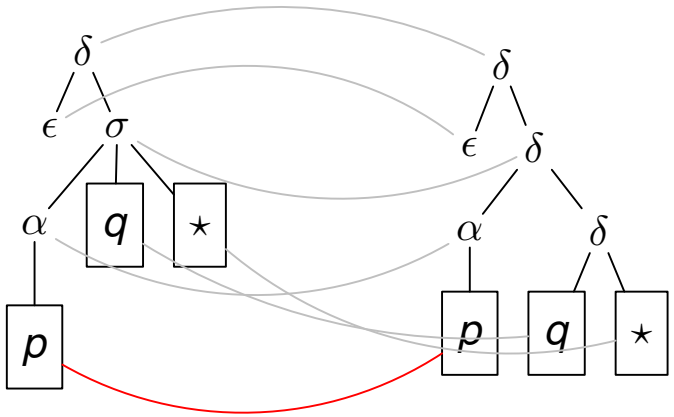


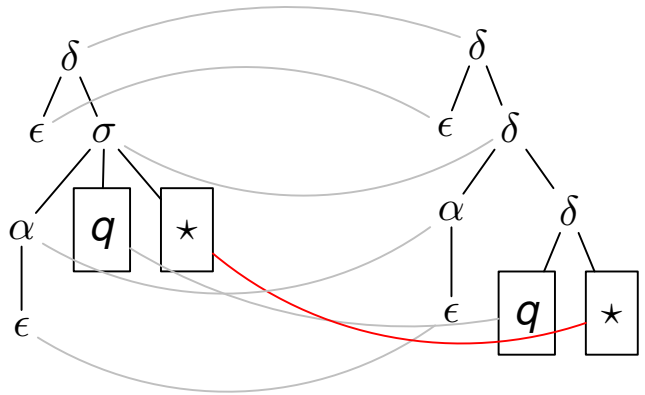
Derivation weight: $0.2 \cdot 0.3 \cdot 0.6 \cdot 0.4$

Derivation

Rules



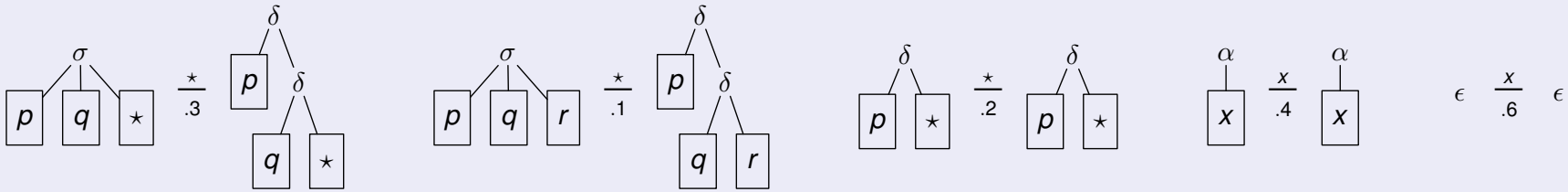


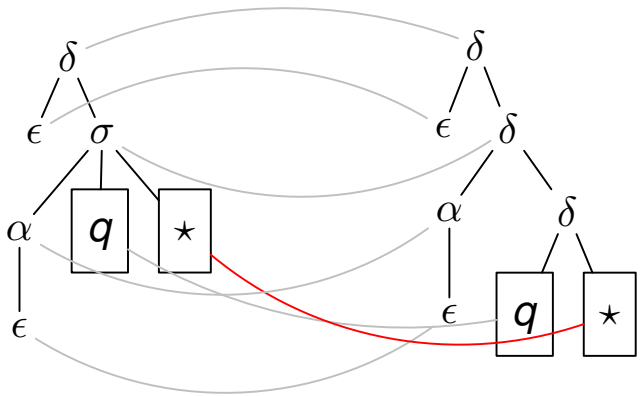


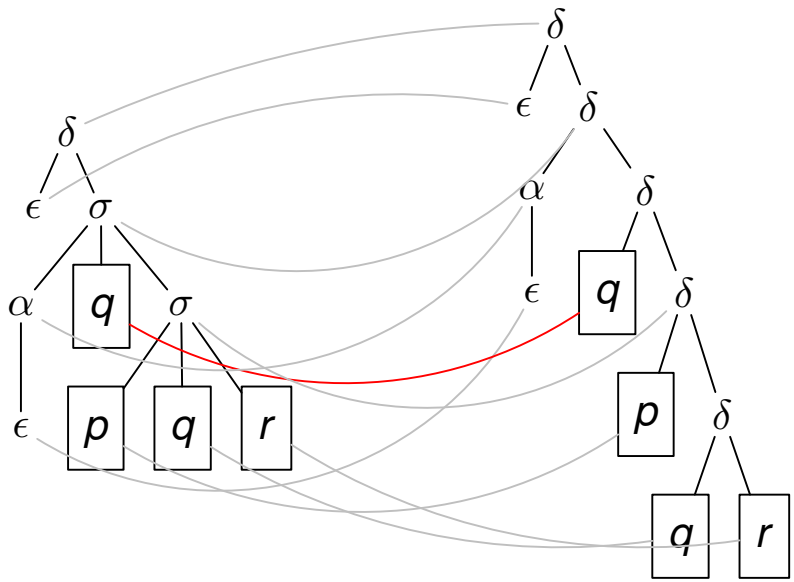
Derivation weight: $0.2 \cdot 0.3 \cdot 0.6 \cdot 0.4 \cdot 0.6$

Derivation

Rules



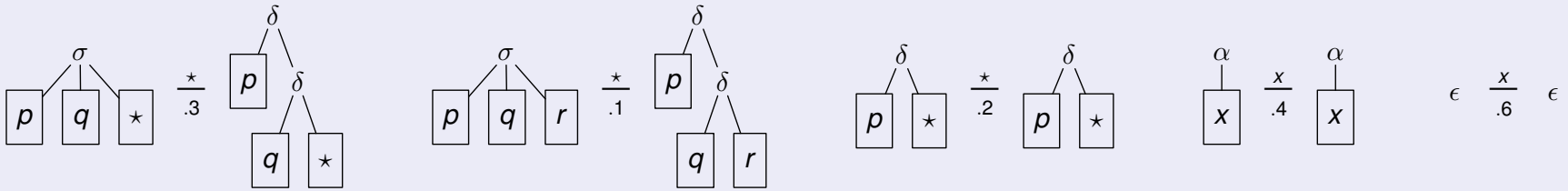


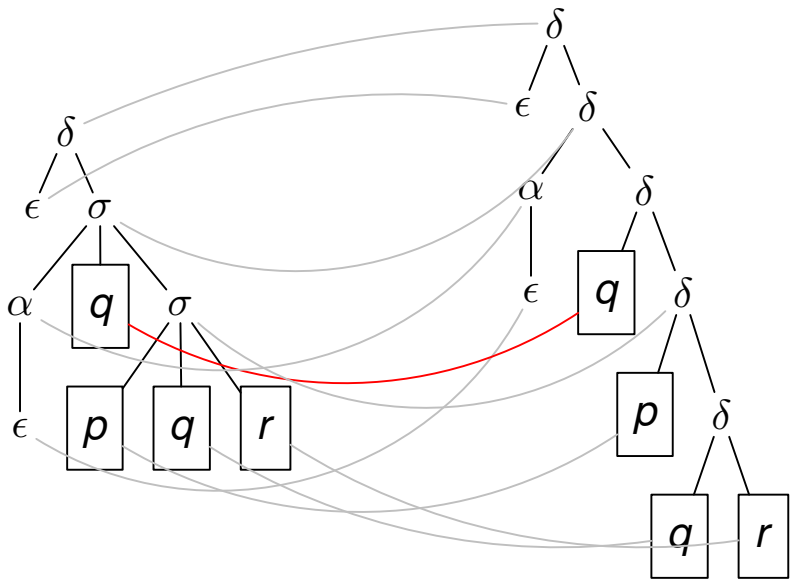


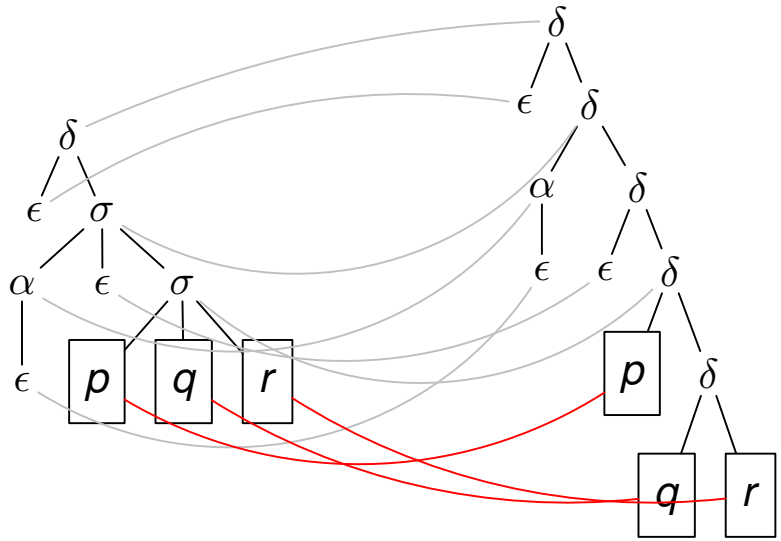
Derivation weight: $0.2 \cdot 0.3 \cdot 0.6 \cdot 0.4 \cdot 0.6 \cdot 0.1$

Derivation

Rules



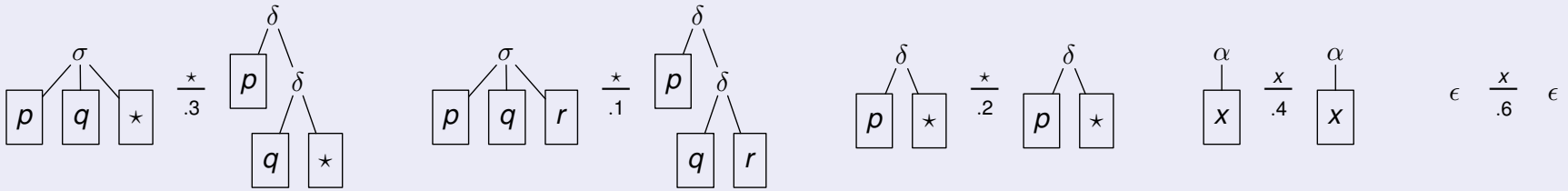


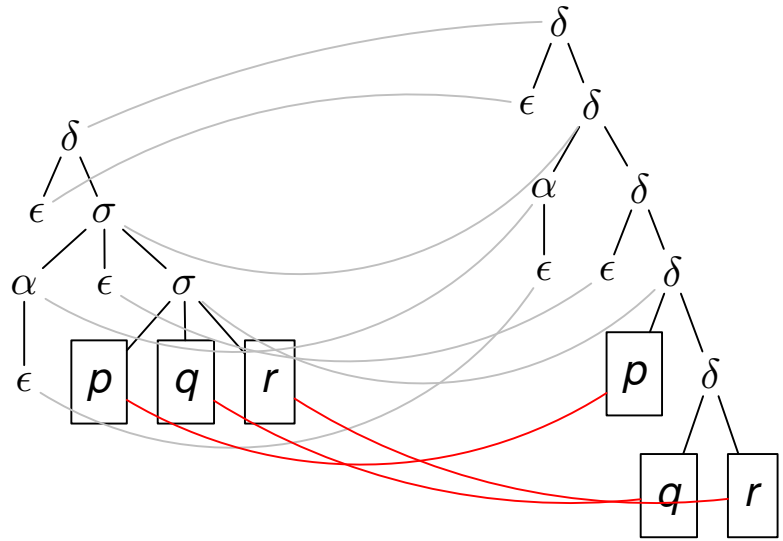


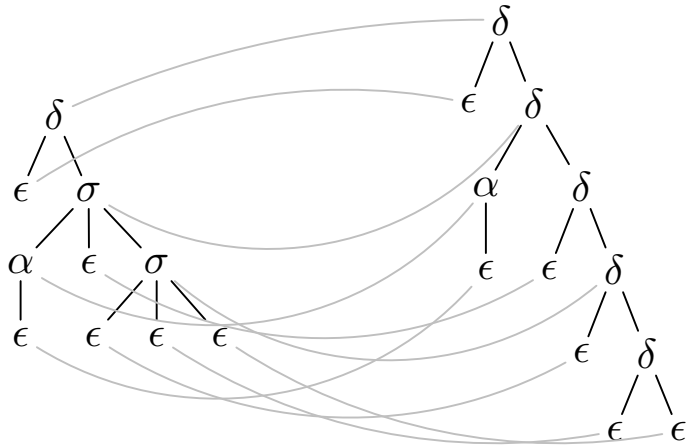
Derivation weight: $0.2 \cdot 0.3 \cdot 0.6 \cdot 0.4 \cdot 0.6 \cdot 0.1 \cdot 0.6$

Derivation

Rules







Derivation weight: $0.2 \cdot 0.3 \cdot 0.6 \cdot 0.4 \cdot 0.6 \cdot 0.1 \cdot 0.6 \cdot 0.6^3$

Weighted Semantics of XTOP

$D_M(t, u)$: set of all (left-most) derivations for the tree pair (t, u)

$\mathrm{wt}(d) = \prod_{i=1}^{n} c_i$ for derivation $d = (q, q) \overset{c_1}{\Rightarrow} \cdots \overset{c_n}{\Rightarrow} (t, u)$ in $D_M(t, u)$

Definition

Computed transformation ($t \in T_\Sigma$ and $u \in T_\Delta$):

$$M(t, u) = \sum_{d \in D_M(t,u)} \mathrm{wt}(d)$$

EM Training

Basics

- EM = expectation maximization [Dempster, Laird, Rubin 1977]
- parameter improvement technique

EM Training

Basics

- EM = expectation maximization [Dempster, Laird, Rubin 1977]
- parameter improvement technique

EM Training

Basics

- EM = expectation maximization [Dempster, Laird, Rubin 1977]
- parameter improvement technique

Expectation

$$c_r = E_{(t,u)\in T}\left[\frac{\sum_{d\in D_M(t,u)} \#_r(d)\cdot p(d)}{\sum_{d\in D_M(t,u)} p(d)}\right]$$

where $\#_r(d)$ is the number of occurrences of the rule r in d

EM Training

Basics

- EM = expectation maximization [Dempster, Laird, Rubin 1977]
- parameter improvement technique

Expectation

$$c_r = E_{(t,u)\in T}\left[\frac{\sum_{d\in D_M(t,u)} \#_r(d)\cdot p(d)}{\sum_{d\in D_M(t,u)} p(d)}\right]$$

$$= E_{(t,u)\in T}\left[\frac{\sum_{d\in D_M(t,u)} \#_r(d)\cdot \mathrm{wt}(d)}{M(t,u)}\right]$$

where $\#_r(d)$ is the number of occurrences of the rule r in d

EM Training

Basics

- EM = expectation maximization [Dempster, Laird, Rubin 1977]
- parameter improvement technique

Maximization

$$\tau = \frac{w_r}{\sum_{r' \in R(r)} w_{r'}}$$

where $R(r)$ contains all rules that are for the same state as r

Question

Which probability distribution is modelled here?

EM Training

Remaining issues

- How to obtain initial weights?
- How to compute $D_M(t, u)$? all derivations for t and u
- How to compute the expectation?

Initial Weights

Possibilities

- uniform distribution on all rules for the same state
- uniform distribution based on state and “parseable” input (difficult)
- use normalized extraction counts

Derivations

- see last lecture
- can all be represented as weighted tree grammar
- often you can just try to enumerate them
  (finite language ⟶ trivial weighted tree grammar)

Initial Weights

Possibilities

- uniform distribution on all rules for the same state
- uniform distribution based on state and “parseable” input (difficult)
- use normalized extraction counts

Derivations

- see last lecture
- can all be represented as weighted tree grammar
- often you can just try to enumerate them
  (finite language $\rightarrow$ trivial weighted tree grammar)

Initial Weights

Possibilities

- uniform distribution on all rules for the same state
- uniform distribution based on state and “parseable” input (difficult)
- use normalized extraction counts

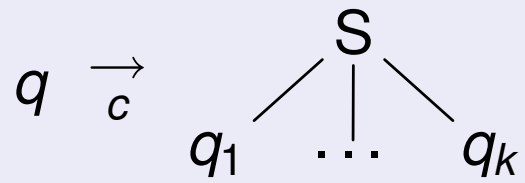
Derivations

- see last lecture
- can all be represented as weighted tree grammar
- often you can just try to enumerate them
  (finite language $\rightarrow$ trivial weighted tree grammar)

Recall: Weighted Tree Grammar

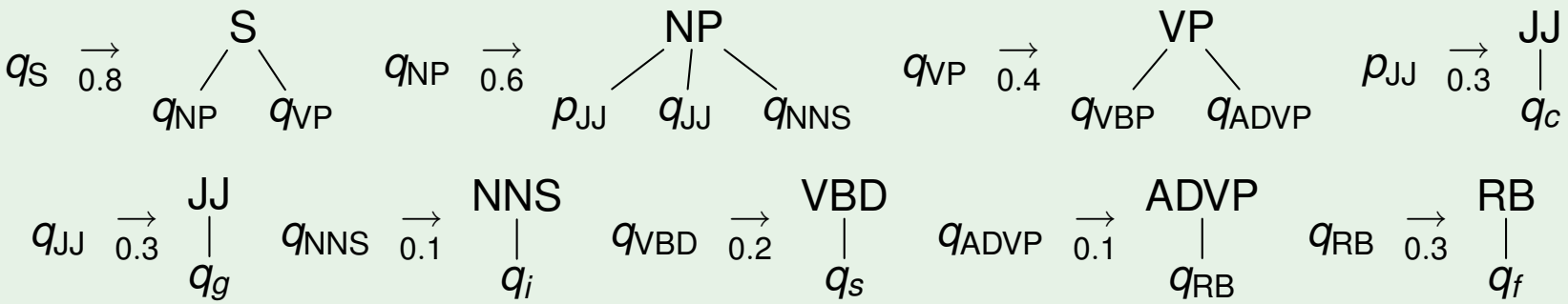
Definition

A regular tree grammar is a grammar with rules of the form



The weights are handled in the same manner as for XTOP (multiply in a derivation & sum different derivation weights)

Example



State Metrics of a Weighted Tree Grammar

Given weighted tree grammar $G = (Q, \Sigma, I, R)$

D_M^q: derivations in M starting from state q

Definition

Inside weight of $q \in Q$:

$$\text{in}(q) = \sum_{d \in D_M^q} \text{wt}(d)$$

Outside weight of $q \in Q$:

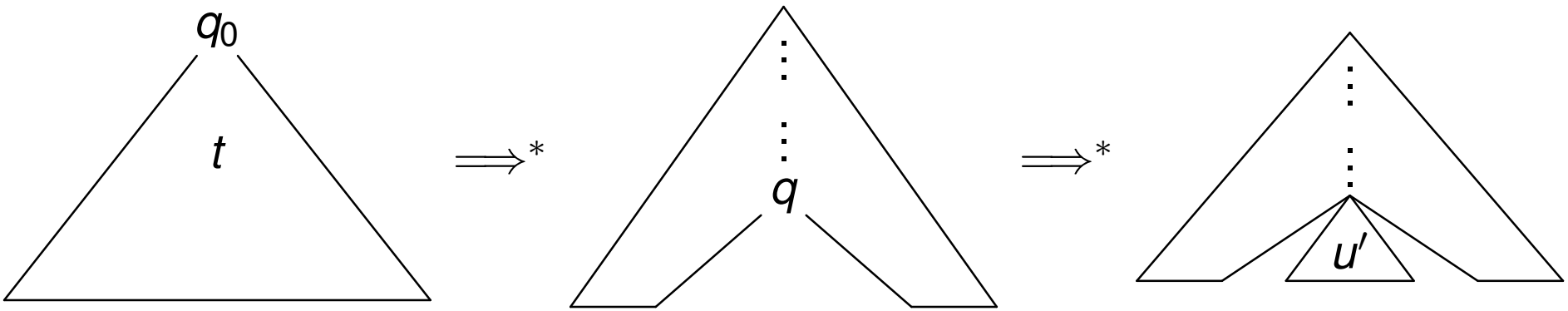
$$\text{out}(q) = \sum_{\substack{t \in C_\Sigma(\{q\}) \\ d \in D_M(t)}} \text{wt}(d)$$

Question

Trivia

Show the relevant part for the

- inside weight of q
- outside weight of q



Expectation Step

Perform the following steps for each input/output pair $(t, u) \in T$ with weight c

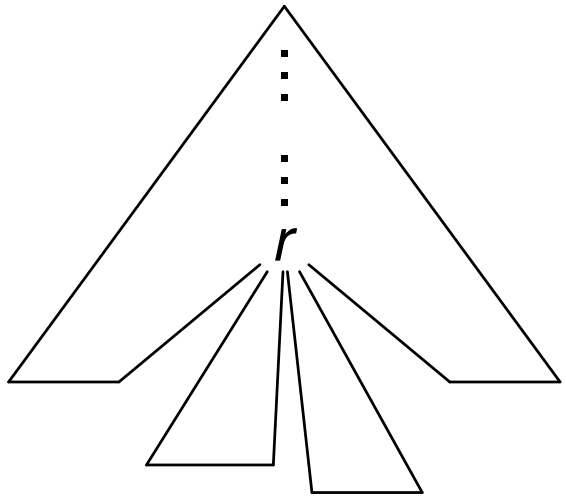
Normalized counts

1. let (Q, R, I, P) be the derivation WTG of M for (t, u)
2. for each production $q \xrightarrow[c']{} r(q_1, \ldots, q_k) \in P$ do

$$\text{count}(r) \leftarrow \text{count}(r) + c \cdot \frac{\text{out}(q) \cdot c' \cdot \prod_{i=1}^{k} \text{in}(q_i)}{M(t, u)}$$

Expectation Step

$$\text{count}(r) \leftarrow \text{count}(r) + c \cdot \frac{\text{out}(q) \cdot c' \cdot \prod_{i=1}^{k} \text{in}(q_i)}{M(t,u)}$$

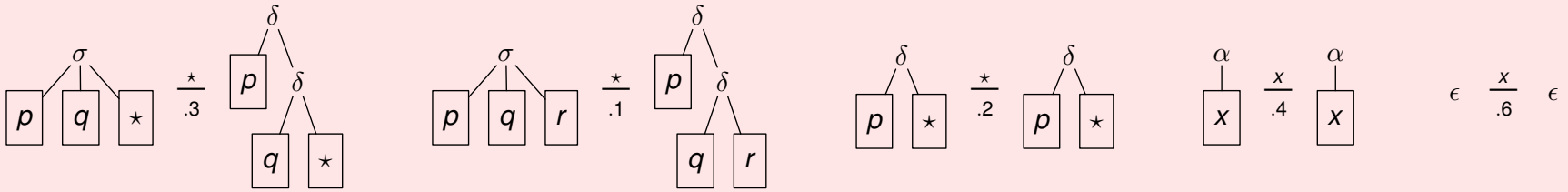


- $\text{out}(q) \cdot c' \cdot \prod_{i=1}^{k} \text{in}(q_i)$: contribution of the rule r
- normalized by $M(t,u)$, the current weight of (t,u)
- this part of the corpus weight c is attributed to r and added to its count

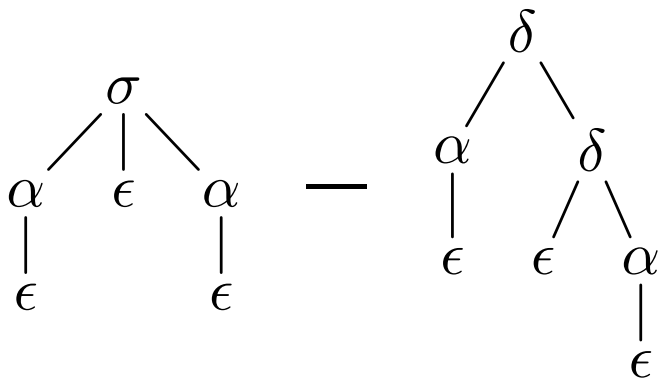
Final Question

Task

Build the derivation WTG for the XTOP with the rules



for the input $\sigma(\alpha(\epsilon), \epsilon, \alpha(\epsilon))$
and the output $\delta(\alpha(\epsilon), \delta(\epsilon, \alpha(\epsilon)))$



References

- ARNOLD, DAUCHET: *Bi-transductions de forêts.* ICALP 1976
- BERSTEL, REUTENAUER: *Recognizable formal power series on trees.* Theor. Comput. Sci. 18, 1982
- ENGELFRIET: *Top-down tree transducers with regular look-ahead.* Math. Systems Theory 10, 1977
- GALLEY, HOPKINS, KNIGHT, MARCU: *What's in a translation rule?* HLT-NAACL 2004
- GRAEHL, KNIGHT: *Training tree transducers.* HLT-NAACL 2004
- MAY, KNIGHT: TIBURON — *a weighted tree automata toolkit.* CIAA 2006
- ROUNDS: *Mappings and grammars on trees.* Math. Systems Theory 4, 1970
- THATCHER *Generalized*^{2} *sequential machine maps.* J. Comput. System Sci. 4, 1970