PROBABILISTIC PARSING

Martin Rajman

Martin.Rajman@epfl.ch

et

Jean-Cédric Chappelier

Jean-Cedric.Chappelier@epfl.ch

Laboratoire d'Intelligence Artificielle

Objectives of this lecture

➥ Present SCFGs, the extension of formal grammars to deal with more difficult problems

Contents

Parsing: probabilisitic approach

WHY probabilities?

Linguistic resources needed for semantic/pragmatic models, even for more sophisiticated syntactic models, are hard to obtain/create

☞ **Extension** of (simple) standard syntactic models

☞ to be able to **make choices** among sentences/structures (in case of ambiguity)

☞ Automatic **Learning** of models from corpora

Parsing: probabilisitic approach (2)

What does it mean to ”probabilize”?

☞ Implicitly represent the linguistic constraints that we do not want to or do not know how to integrate into the models:

Set of linguistic phenomena that **cannot** or are **hard to express** in operational terms but that still are **possible to evaluate** (on corpora)

The probability is then a measure of the quality of the adequation between the sentence/structure and the underlying model

Parsing: probabilisitic approach (3)

WHAT is ”probabilized”?

☞ The point of view is different depending on whether the syntactic model is used as a **recognizer** or as an **analyzer**

- A *recognizer* in only able to tell whether the input sentence is correct or not
- An *analyzer* is more complex and produces additional information for the correct sentences: a structure representing the syntactic organization of the words.

Parsing: probabilisitic approach (4)

	recognizer	analyzer
what is probabilized?	sentences	parse trees associated to a given sentence
meaning of the probabilities	adequation of a sentence to the model	adequation of a structure (tree) to the model
	$P(w_1^n)$	$P(T\|w_1^n)$
example	N-grams	SCFG

Notice: Although in principle probabilities have no reason to depend on the formal description of the language they are associated with, their operational definition in practice can hardly be build independently of the generative model defining the language (i.e. the grammar)

Parsing: probabilisitic approach (5)

General scheme of realization of probabilistic model:

☞ Identify the probability to estimate: $P(W_1 ... W_n)$ or $P(A|W_1 ... W_n)$

☞ on the basis of linguistic hypotheses, express this probability by <u>restricted</u> number of parameters: $P = f(p_1 ... p_k)$

☞ On the basis of a well defined corpora, estimate retained parameters in order to be able to compute probabilities

N-grams

One possible probabilization of a language: estimate probabilities of sequences of words by their occurrence frequencies in a reference corpus

❢ For an accurate estimation, **huge** amounts of data are required

☞ reducing the number of parameters: estimate probabilities of fixed-size sequences (N-grams) and then approximate the probabilities of a longer sequence on the basis of these parameters:

$$P(w_1, ..., w_n) = P(w_1, ..., w_{N-1}) \cdot \prod_{i=N}^{n} P(w_i | w_{i-N+1}, ..., w_{i-1})$$

Example: ($N = 2$)

the cat ate a mouse	ate mouse a cat the
(the cat) (cat ate) (ate a) (a mouse)	(ate mouse) (mouse a) (a cat) (cat the)

Contents

SCFG: Summary

a Stochastic Context-Free Grammar is

- a CFG for which
- each rule R is associated with a stochastic coefficient $p(R)$ such that
  - $0 \leq p(R) \leq 1$
  - $\displaystyle\sum_{R':\text{left}(R')=\text{left}(R)} p(R') = 1$
- $P(T = R_0 \circ ... \circ R_n) = \displaystyle\prod_{i=0}^{n} p(R_i)$

Maximization or

consistent grammars

Notations

For a context-free grammar $\mathcal{G}$ we will use the following notations:

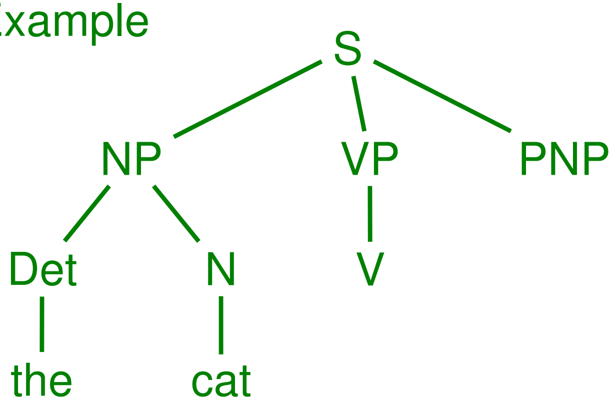
$\mathcal{L}(\mathcal{G})$ the language recognized by $\mathcal{G}$ $\qquad$ $\mathcal{R}(\mathcal{G})$ the set of rules of $\mathcal{G}$

$\mathcal{A}(\mathcal{G})$ the set of **partial** trees of $\mathcal{G}$ (with root S)

$\mathcal{T}(\mathcal{G})$ the set of complete trees of $\mathcal{G}$ $\qquad$ ($\mathcal{T}(\mathcal{G}) \subset \mathcal{A}(\mathcal{G})$)

For a tree T of $\mathcal{A}(\mathcal{G})$, $r(T)$ will denote its root, $F(T)$ the ordered sequences of its leaves and lmnt(T) the lest-most non-terminal leave of T. If T does not have any non-terminal leave, lmnt$(T) = \varepsilon$

Example



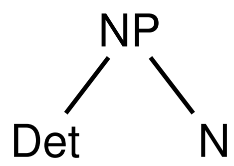
$F(T) = \{$the, cat, V, PNP$\}$

and lmnt$(T) = $ V

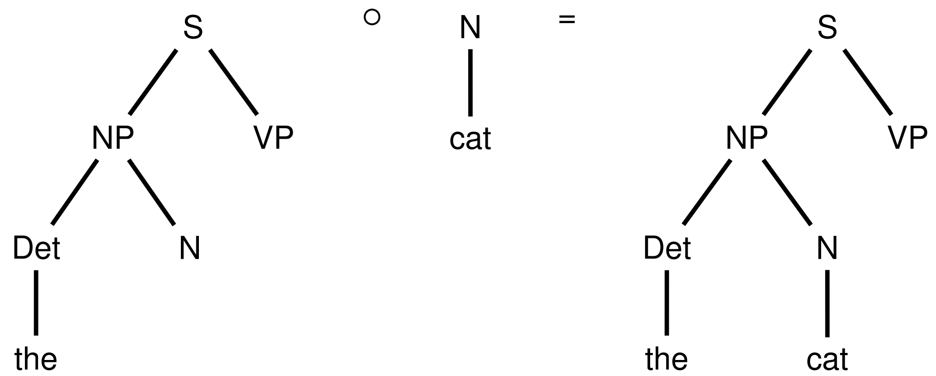
Notations (2)

Furthermore, the same notation R will be used for both the rule and the corresponding elementary tree:

NP $\rightarrow$ Det N



The symbol $\circ$ denotes the internal composition rule on $\mathcal{A}(\mathcal{G})$ that returns the tree resulting from the substitution of the left-most non-terminal leave of the left tree by the right tree when it is possible, and ε if not.



For a rule R of $\mathcal{R}(\mathcal{G})$, left(R) denotes the left-hand side of R

SCFG

Desambiguation: Let $\mathcal{G}$ be a Stochastic CFG and $W = w_1^n$ a sentence with several interpretations $T_1, ..., T_k$ according to $\mathcal{G}$. The goal is to choose among the T_is

In a standard approach, such a choice is made on semantic/pragmatic criteria

In the probabilistic approach, the choice is made according to the probabilities of the T_i trees. In other terms, we are looking for:

$$T = \underset{T_i \supset W}{\text{Argmax}}\, P(T_i|W)$$

But $P(T_i|W) = \frac{P(T_i,W)}{P(W)} = \frac{P(T_i)}{P(W)}$ since T_i precisely is a tree that analyses W

We are therefore looking for $T = \underset{T_i \supset W}{\text{Argmax}}\, P(T_i)$

SCFG: formalization

T_i is interpreted as the result of a given (unknown) stochastic process ξ

☞ because of the one-to-one mapping that exists in CFG between trees and derivations (sequences of rules), ξ is supposed to be a stochastic process on **rules**, i.e a random sequence in $\mathcal{R}(\mathcal{G})$

☞ We will therefore characterize $P(T)$ using $P(\xi = R_0, ..., R_n)$

$$P(\xi = R_0, ..., R_n) = P(R_0) \cdot \prod_{i=1}^{n} P(R_i | R_1, ..., R_{i-1})$$

Definition of ξ

To fully define ξ we need the definition of $P(R_0)$ and $P(R_i|R_1, ..., R_{i-1})$:

- R_0 is the *constant* "random" variable S (null-depth tree with root S, the start-symbol)

  Therefore $P(R_0 = \text{S}) = 1$
- $P(R_i|R_0, ..., R_{i-1})$ is null if left$(R_i) \neq$ lmnt$(R_0 \circ ... \circ R_{i-1})$

☞ What value for the probability when it is not zero?

Value for $P(R_i|R_0, ..., R_{i-1})$

As up to now, this probability is conditioned by $\text{left}(R_i) = \text{lmnt}(R_0 \circ ... \circ R_{i-1})$

If we make the assumption that it is conditioned **ONLY** by this, then

$$P(R_i|R_0, ..., R_{i-1}) = P(R_i|\text{lmnt}(R_0 \circ ... \circ R_{i-1})) = P(R_i|\text{left}(R_i))$$

which therefore only depends on R_i and will be denoted by $p(R_i)$. It is called the "*stochastic coefficient*" of the rule R_i

☞ $p(R_i)$ is a **parameter** of the processus ξ and, by construction, we have:

$$\forall R \in \mathcal{R}(\mathcal{G}) \quad \sum_{R' \in \mathcal{R}(\mathcal{G}):\text{left}(R')=\text{left}(R)} p(R') = 1$$

Notice that limiting $P(R_i|R_0...R_{i-1})$ to the conditioning by $P(R_i|\text{lmnt}(R_0 \circ ... \circ R_{i-1}))$ only is a **strongly restrictive hypothesis** on the processus

.

Probability of a tree?

Finaly, the probability of a (valid) sequence of rules is:

$$P(R_0, ..., R_n) = \prod_{i=1}^{n} p(R_i)$$

Each T in $\mathcal{T}(\mathcal{G})$ corresponds to a unique (valid) sequence of rules, therefore

$$P(T) = P(R_0, R_1, ..., R_k) = \prod_{i=1}^{k} p(R_i)$$

In short: For SCFGs, the probability of a tree is the product of the stochastic coefficient associated to its rules

Probability of a tree? (2)

BUT... is it really a probabilty on $T(\mathcal{G})$?...

What is $\displaystyle\sum_{T \in T(\mathcal{G})} P(T)$?

- It converges
- towards a limit lower or equal to 1
- But that can be < 1

  Example: S $\rightarrow$ S S (p) S $\rightarrow$ a (1-p)

Therefore the correct probabilization is: $$\widehat{P}(T) = \frac{P(T)}{\displaystyle\sum_{T \in T(\mathcal{G})} P(T)}$$

In the case where the grammar is **consistent** (i.e. $\sum P(A) = 1$) (or in the case where only the maximum probability is considered), the two approches are equivalent. The only problematic case here is when one deals simultaneously with several not consistent grammars.

Probability of a sentence $P(W)$

The probability of a sentence is defined by:

$$P(W) = \sum_{\substack{T \in T(\mathcal{G}): \\ F(T)=W}} \widehat{P}(T)$$

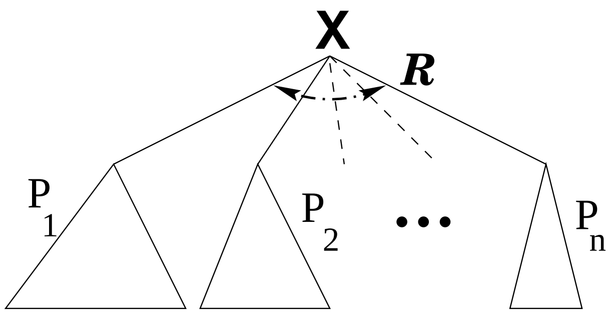
Notice that $P(T,W) = \widehat{P}(T) \cdot \delta(W = F(T))$ (Kronecker notation) which justifies the formulas used at the beginning of the course

SCFG: Implementation

It is possible to compute $\mathrm{Argmax}\, P(T_i)$ and/or $P(W) = \sum P(T_i)$ during the bottom-up phase of the CYK analysis, using dynamic programming

For a given element in a cell, a value v_i representing the maximum (or the sum) of the probabilities of its interpretations is stored

Notice:



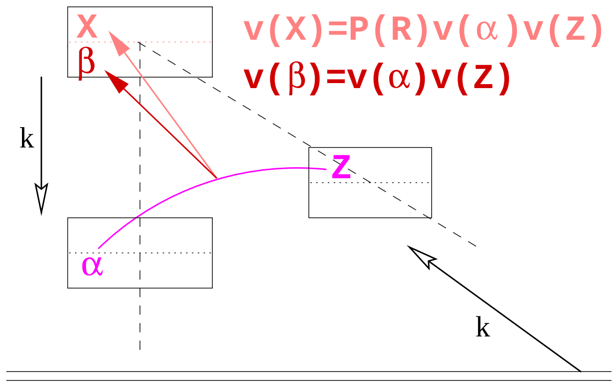
$$
\begin{aligned}
P(X) &= \prod p(R_i) \\
&= p(R) \cdot P_1 \cdots P_n
\end{aligned}
$$

SCFG: Implementation (2)

When a new interpretation of element i is build (by composition of elements j and k), the value v_i is updated according to:

$$v_i = \max(v_i, v_j\, v_k\, \rho_i)$$

(or) $$v_i = v_i + v_j\, v_k\, \rho_i$$



with $\rho_i = 1$ if element i is a item $[\alpha \bullet \ldots]$

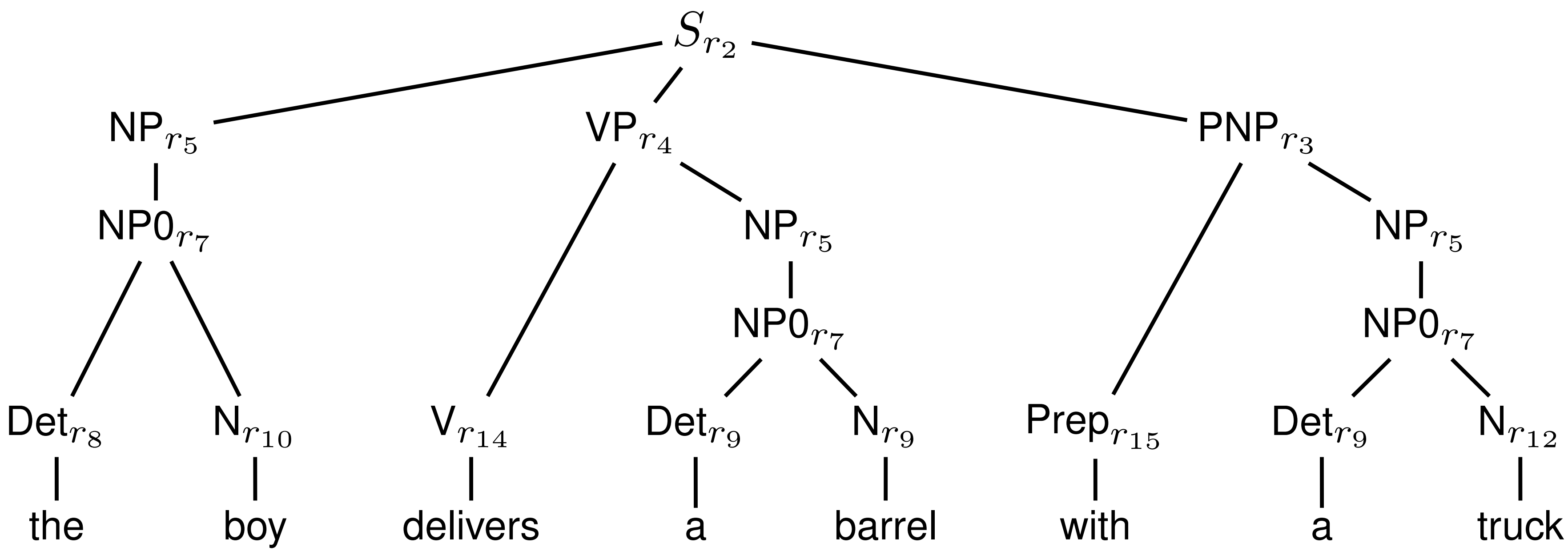
and $\rho_i = p(R_k)$ if element i is a non-terminal obtained by applying rule R_k

The initial value for the v_is is 0

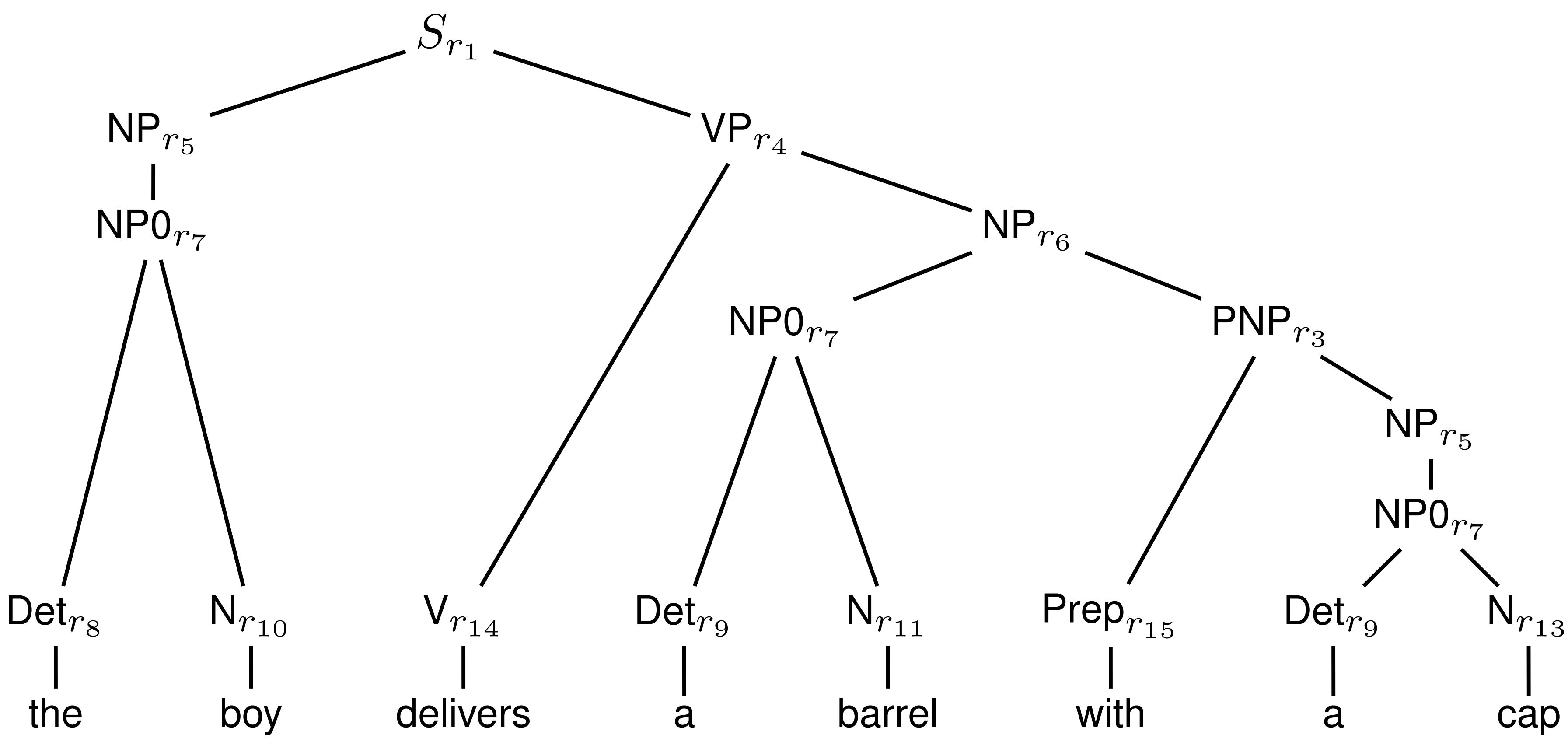
Grammar extraction from a treebank

Let us consider that a treebank made of the following parse trees is available:

T_1:



T_2:


S_{r_1}
NP$_{r_5}$
VP$_{r_4}$
NP0$_{r_7}$
NP$_{r_6}$
NP0$_{r_7}$
PNP$_{r_3}$
NP$_{r_5}$
NP0$_{r_7}$
Det$_{r_8}$
N$_{r_{10}}$
V$_{r_{14}}$
Det$_{r_9}$
N$_{r_{11}}$
Prep$_{r_{15}}$
Det$_{r_9}$
N$_{r_{13}}$
the
boy
delivers
a
barrel
with
a
cap

Grammar extraction (2)

From the trees present in the corpus, we can extract the context-free grammar G, made of the following 15 rules:

rule	p_i
r_1: `S -> NP VP`	p_1
r_2: `S -> NP NP PNP`	p_2
r_3: `PNP -> Prep NP`	p_3
r_4: `VP -> V NP`	p_4
r_5: `NP -> NP0`	p_5
r_6: `NP -> NP0 PNP`	p_6
r_7: `NP0 -> Det N`	p_7

rule	p_i
r_8: `Det -> the`	p_8
r_9: `Det -> a`	p_9
r_{10}: `N -> boy`	p_{10}
r_{11}: `N -> barrel`	p_{11}
r_{12}: `N -> truck`	p_{12}
r_{13}: `N -> cap`	p_{13}
r_{14}: `V -> delivers`	p_{14}
r_{15}: `Prep -> with`	p_{15}

where the p_i denote the probabilities associated with each of the rules

☞ How can we estimate them?

Estimating the probabilities

supervised learning: When a tree-bank (annotated corpus) is available, stochastic coefficients are estimated by the relative frequencies (maximum likelihood estimation):

$$p(R) = \frac{\text{nb. occurrences of } R}{R' \text{ such that left}(R') = \text{left}(R)}$$

unsupervised learning: When only text is available (**and** also a grammar) : EM estimation of the coefficients : inside-outside algorithm

- iterative algorithm
- converges towards a local minimum
- highly sensitive to initial values

hybrid approaches: using a (small) tree-bank and a (large) corpus of text

Estimating the probabilities (2)

In our case (supervised learning), we get:

rule	p_i
r_1: `S -> NP VP`	1/2
r_2: `S -> NP NP PNP`	1/2
r_3: `PNP -> Prep NP`	1
r_4: `VP -> V NP`	1
r_5: `NP -> NP0`	5/6
r_6: `NP -> NP0 PNP`	1/6
r_7: `NP0 -> Det N`	1

rule	p_i
r_8: `Det -> the`	1/3
r_9: `Det -> a`	2/3
r_{10}: `N -> boy`	1/3
r_{11}: `N -> barrel`	1/3
r_{12}: `N -> truck`	1/6
r_{13}: `N -> cap`	1/6
r_{14}: `V -> delivers`	1
r_{15}: `Prep -> with`	1

Keypoints

- Probabilities of SCFGs are implicit linguistic constraints serving as measures of the adequation between the sentence and the model
- The role of probabilities is to identify the correctness of the sentence and eventually to choose one interpretation among several
- Calculation of probabilities of syntactic interpretations of sentences
- Estimation of probabilities of SCFGs from training corpora

References

[1] C. D. Manning, H. Schütze, *Foundations of Statistical Natural Language Processing*, ch. 11, 12, MIT, 1999.

[3] D. Jurafsky & J. H. Martin, *Speech and Language Processing*, ch. 12, Prentice Hall, 2000.

[4] R. Dale, H. Moisl & H. Sommers, *Handbook of Natural Language Processing*, ch. 22, Dekker, 2000.