Natural Language Parsing

Foundations of Language Science and Technology (WS 2008/2009)

Yi Zhang

Language Technology Lab, DFKI GmbH
Cluster of Excellence - Multimodal Computing and Interaction
Department of Computational Linguistics, Saarland University

Language & Grammar

- Language
  - Structural
  - Productive
  - Ambiguous, yet efficient in human-human communication

- Grammar
  - Generalization of regularities in language structures
  - Morphology & syntax, often complemented by phonetics, phonology, semantics, and pragmatics.

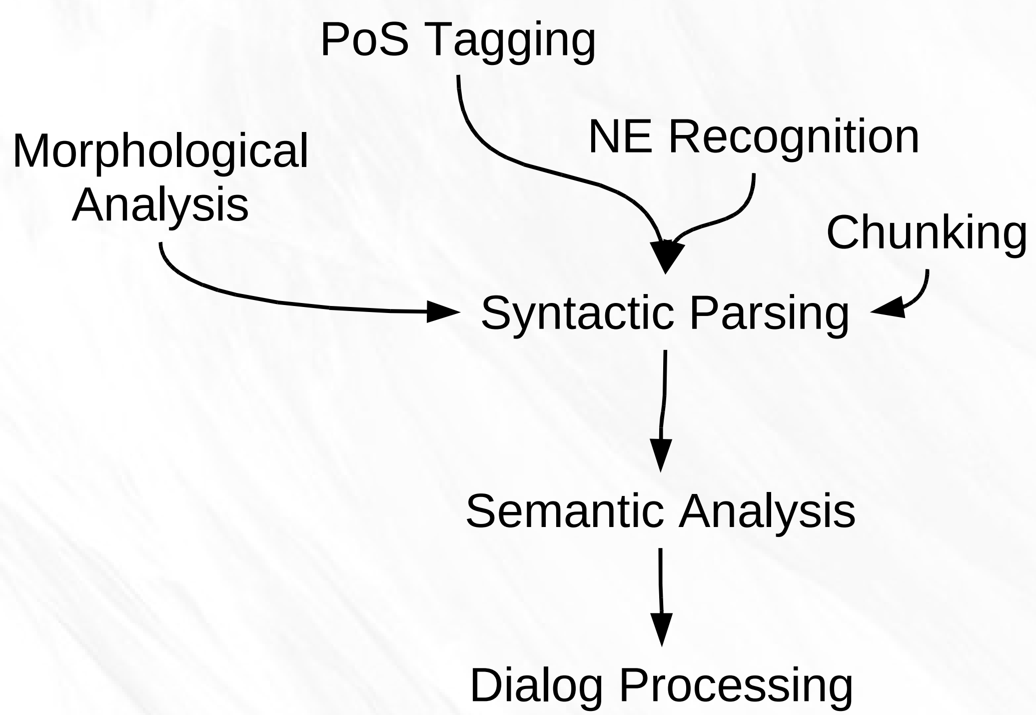
Grammar Frameworks

- Formalism
  - Mathematical rigor
  - Facilitates the development of linguistic theory
- Formal linguistic theory
  - Formalized description of language phenomena using the formalism
- A grammar framework does NOT correspond to parsing/generation algorithms. But a well-designed framework should bear processing steps in mind

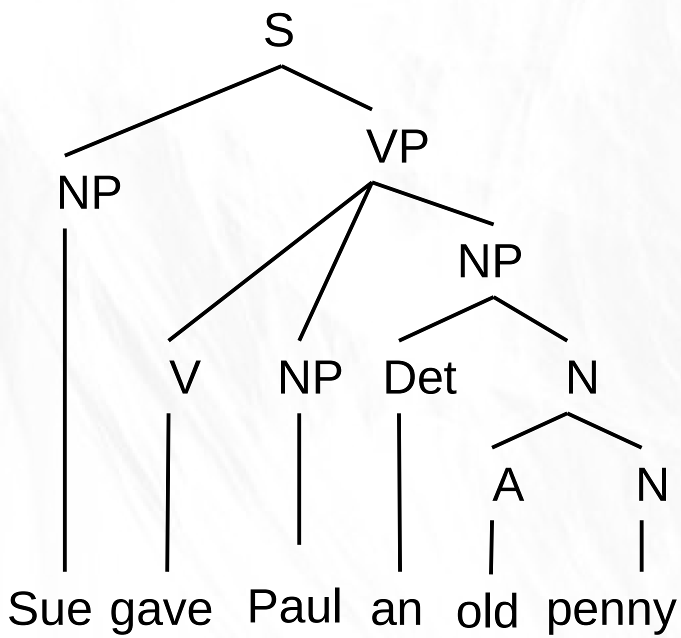
Ambiguity

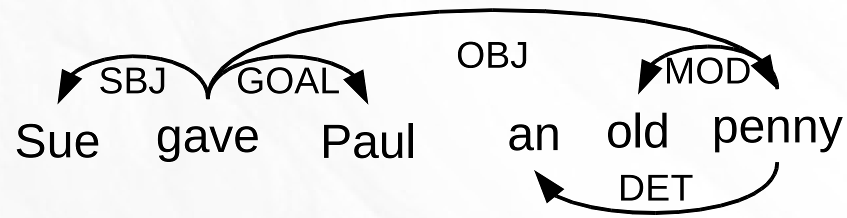
- Human languages are ambiguous on almost every layer
- Grammar frameworks are designed to represent necessary ambiguities, and eliminate unnecessary ones
- Parsing models are responsible of efficiently apply grammar descriptions and retrieve analyses

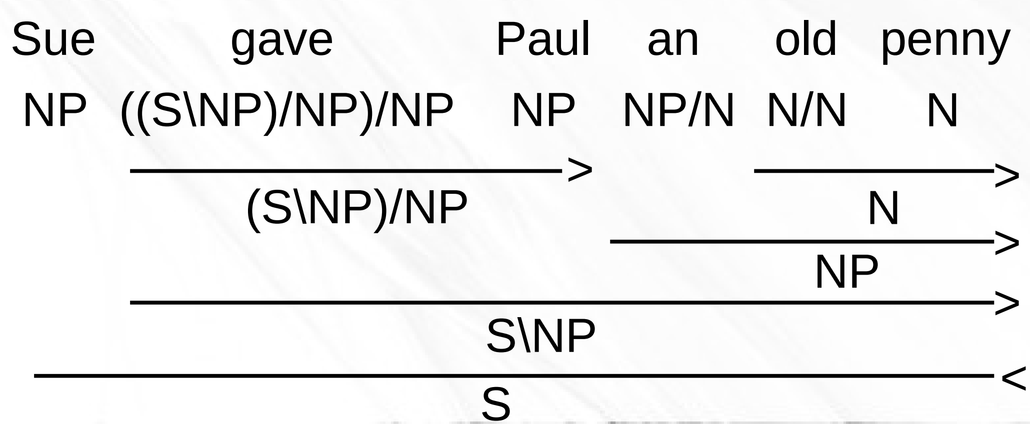
Syntactic Parser as NLP Component



Trees (or not)


S
VP
NP
NP
V
NP
Det
N
A
N
Sue gave Paul an old penny


OBJ
SBJ
GOAL
MOD
Sue gave Paul an old penny
DET


Sue gave Paul an old penny
NP ((S\NP)/NP)/NP NP NP/N N/N N
>
(S\NP)/NP
>
N
>
NP
>
S\NP
<
S

Chomsky Hierarchy

- Type 0 (unrestricted rewriting system)

$$\alpha \rightarrow \beta$$

- Type 1 (context sensitive grammars)

$$\phi A \omega \rightarrow \phi \beta \omega , A \in V_N , \beta \neq \epsilon$$

- Type 2 (context free grammars)

$$A \rightarrow \beta , \beta \neq \epsilon$$

- Type 3 (regular grammars)

$$A \rightarrow xB \vee A \rightarrow x , x \neq \epsilon$$

Context-Free Grammar

- $\langle V_T, V_N, \wp, S \rangle$
  - V_T : Terminals
  - V_N : Non-Terminals
  - $\wp$: Productions
    - $A \rightarrow \beta,\ A \in V_N, \beta \in (V_N \cup V_T)*$
  - S : Start symbol $S \in V_N$

Context-Free Phrase Structure Grammar

- S -> NP VP
- NP -> Det N
- NP -> Adj NP
- VP -> V
- VP -> V NP
- VP -> Adv VP

- N -> dog | cat
- Det -> the | a
- V -> chases | sleeps
- Adj -> gray | lazy
- Adv -> fiercely

CFG Derivation

- If $\phi = \beta A \gamma, \omega = \beta \alpha \gamma$ and $A \rightarrow \alpha \in \wp$
  then ω *follows* ϕ , $\phi \Rightarrow \omega$
- A sequence of strings $\phi_{1,} \phi_{2,} \cdots, \phi_m$
  where for all i $1 \le i \le m-1$, $\phi_i \Rightarrow \phi_{i+1}$
  then $\phi_{1,} \phi_{2,} \cdots, \phi_m$ is a derivation from ϕ_1 to ϕ_m
- ``*derivable*'' relation: transitive, reflexive

Earley's Algorithm

- Input: ${}_0 w_1 {}_1 w_2 {}_2 \cdots {}_{n-1} w_n {}_n$
- Chart: a set of items $\langle h, i, A \rightarrow \alpha . \beta \rangle$
  - h, i : positions in the input $0 \leqslant h \leqslant i \leqslant n$
  - $A \rightarrow \alpha . \beta$: dotted rule $A \rightarrow \alpha \beta \in \wp$
    - α : rhs prefix that has already been applied to input from h to i
    - β : rhs suffix yet to be found

Earley's Algorithm

- **Initialize**

foreach $S \rightarrow \alpha \in \wp$
 $\mathbb{C} \Leftarrow \langle 0, 0, S \rightarrow . \alpha \rangle$

- **Scan(i)**

if $w_i = a \wedge \langle h, i-1, A \rightarrow \alpha . a \beta \rangle \in \mathbb{C}$
 $\mathbb{C} \Leftarrow \langle h, i, S \rightarrow \alpha a . \beta \rangle$

- **Complete(i)**

foreach $\langle h, i, A \rightarrow \alpha . \rangle \in \mathbb{C}$
 foreach $\langle k, h, B \rightarrow \beta . A \gamma \rangle \in \mathbb{C}$
  $\mathbb{C} \Leftarrow \langle k, i, B \rightarrow \beta A . \gamma \rangle$

- **Predict(i)**

foreach $\langle h, i, A \rightarrow \alpha . B \beta \rangle \in \mathbb{C}$
 foreach $B \rightarrow \gamma$
  $\mathbb{C} \Leftarrow \langle i, i, B \rightarrow . \gamma \rangle$

- **Parse**

Initialize
for $i = \langle 1 \cdots n \rangle$
 Predict(i-1)
 Scan(i)
 Complete(i)
if $\exists \langle 0, n, S \rightarrow \alpha . \rangle \in \mathbb{C}$
 return *success*
else
 return *failed*

An Example

The Det dog N chases V a Det cat N

	0	1	2	3	4	5
0	S->.NP VP NP->.Det N					
1	NP->Det .N					
2	NP->Det N. S->NP .VP		VP->.V VP->.V NP			
3	S->NP VP.		VP->V. VP->V .NP	NP->.Det N		
4				NP->Det .N		
5	S->NP VP.		VP->V NP.	NP->Det N.		

١. S -> NP VP

٢. VP -> V NP

٣. VP-> V

٤. NP -> Det N

Probabilistic Context-Free Grammar

- Each rule is augmented by a probability

$$\forall A \in V_N \sum_{\forall \alpha, A \rightarrow \alpha \in \wp} P(A \rightarrow \alpha) = 1$$

- The probability of a derivation is the product of rule probabilities of each derivation step

$$P(t) = \prod_{A \rightarrow \alpha \in t} P(A \rightarrow \alpha)$$

More Probabilities

- String probability $P(X \Rightarrow^* x)$
  Sum of the probabilities of all left-most derivations producing x from X

- Sentence probability $P(S \Rightarrow^* x)$
  Sum of the probabilities of all left-most derivations producing x from start symbol S

- Prefix probability $P(S \Rightarrow_L^* x)$
  Sum of the probabilities of all sentences having x as prefix

- Structured language model

Parsing with PCFG

- Earley's algorithm can be adapted to carry probabilities
  - Predict $\langle h, i, A \rightarrow \alpha . B\beta \rangle [x, y] \Rightarrow \langle i, i, B \rightarrow . \beta \rangle [+x * P(B \rightarrow \beta), P(B \rightarrow \beta)]$
  - Scan $\langle h, i-1, A \rightarrow \alpha . a\beta \rangle [x, y] \Rightarrow \langle h, i, A \rightarrow \alpha a . \beta \rangle [x, y]$
  - Complete $\langle h, i, A \rightarrow \alpha . \rangle [x_{1,} y_1] \wedge \langle k, hB \rightarrow \beta . A\gamma \rangle [x_{2,} y_2]$
    $\Rightarrow \langle k, h, B \rightarrow \beta A . \gamma \rangle [+y_1 * x_2, +y_1 * y_2]$
- Inside probability: $\beta_A(p, q)$
- Best-first parsing with Viterbi Algorithm

Statistical Constituent Parsers

- Collins' parser [Collins 1997]
- Charniak's parser [Charniak 2000]
- Reranking model [Collins et al. 2005]
- Self-training [McClosky 2006]

Statistical Dependency Parsing

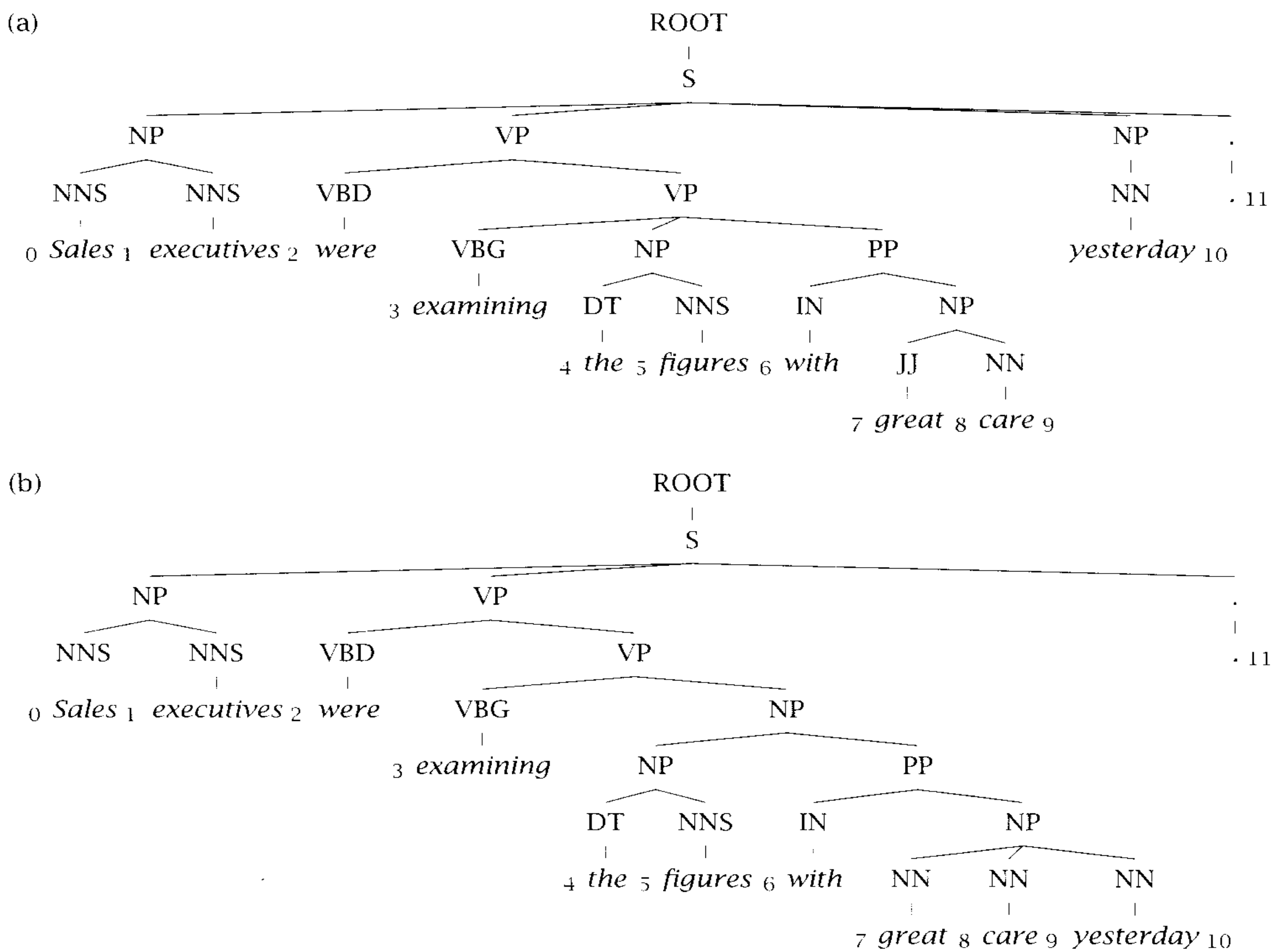
- Graph-based approach
  [Eisner 1996] [McDonald et al. 2005]
  - Edge-factorized scoring model
  - Efficient algorithms to find maximal spanning tree
  - Allows non-projective dependency structures
- Transition-based approach
  [Covington 2001] [Nivre et al. 2007]
  - (Near) deterministic parsing
  - Projective/pseudo-projective

Parsing with Richer Formalisms

- TAG
  [Schabes et al. 1990] [Xia 2001]
- CCG
  [Hockenmaier et al. 2007] [Clark et al. 2007]
- LFG
  [Riezler et al. 2002] [Cahill et al. 2004]
- HPSG

  [van Noord 2006] [Miyao et al. 2008] [Callmeier 2001]

Evaluation -- PARSEVAL



(c) Brackets in gold standard tree (a.):
S-(0:11), **NP-(0:2)**, VP-(2:9), VP-(3:9), **NP-(4:6)**, PP-(6-9), NP-(7,9), *NP-(9:10)

(d) Brackets in candidate parse (b.):
S-(0:11), **NP-(0:2)**, VP-(2:10), VP-(3:10), NP-(4:10), **NP-(4:6)**, PP-(6-10), NP-(7,10)

(e)

Precision:	3/8 = 37.5%	Crossing Brackets:	0
Recall:	3/8 = 37.5%	Crossing Accuracy:	100%
Labeled Precision:	3/8 = 37.5%	Tagging Accuracy:	10/11 = 90.9%
Labeled Recall:	3/8 = 37.5%		

Domain Adaptability and Multilinguality

- Statistical parsing models usually performs well in in-domain tests and suffer accuracy drop when tested out-domain (typically 6~8% performance drop when train on WSJ and test on Brown)
- Typological differences between languages require different parsing models (morphology, word order, projectivity, etc.)

Open Questions

- How much linguistics is required for parsing?
- How do we evaluate a parser?
- How to make trade-offs between adequacy, accuracy and efficiency?
- ... ...