Formal Syntax and Grammar Engineering

Stephan Oepen

Universitetet i Oslo & CSLI Stanford

oe@csli.stanford.edu

Lilja Øvrelid

Göteborgs Universitet

lilja.ovrelid@svenska.gu.se

http://www.delph-in.net/courses/04/fs/

So, What is Computational Linguistics?

... teaching computers our language. (Alien Researcher)

... the scientific study of human language—specifically of the system of rules and the ways in which they are used in communication—using mathematical models and formal procedures that can be realized and validated using computers; a cross-over of many disciplines. (Stanford Professor)

... a cornerstone of our pioneering .NET initiative and the operating systems of the future; innovative technology that will change our world. (President of US-Based Software Company)

... a sub-discipline of our Artificial Intelligence programmes.

(CMU Professor)

What About Formal Syntax Then?

Grammaticality

- *Kim was happy because ____ passed the exam.*
- *Kim was happy because ____ final grade was VG.*
- *Kim was happy when she saw ____ on television.*

Meaning

- *Kim gave Sandy a book.*
- *Kim gave a book to Sandy.*
- *Sandy was given a book by Kim.*

Ambiguity

- *I saw the astronomer with the telescope.*
- *Have her report on my desk immediately!*

What We Are About to Do (and Why)

Course Outline

- Develop understanding of (natural) language as a system of rules;
- learn how to *formalize* grammars through typed feature structures;
- adapt and develop sequence of trivial HPSG grammars in LKB;
- solve daily excercises: immediate gratification (risk of late hours).

Why Computational Grammars

- **research** formalize linguistic theories with complex interactions of language phenomena; identify cross-language generalizations;
- **education** teach frameworks or analyses in formal morphology, syntax, and semantics; support student experimentation;
- **applications** embed grammar-based natural language analysis in research prototypes and commercial applications.

Student Experimentation — Immediate Gratification

Example: Norwegian – English Machine Translation

Some Areas of Descriptive Grammar

Phonetics	***The study of speech sounds.***
Phonology	***The study of sound systems.***
Morphology	***The study of word structure.***
Syntax	***The study of sentence structure.***
Semantics	***The study of language meaning.***
Prgamatics	***The study of language use.***

Grammar Engineering from a CS Perspective

Implementation Goals

- Translate linguistic constraints into specific formalism $\rightarrow$ formal model;
- computational grammar provides mapping between form and meaning;
- assign correct analyses to grammatical, reject ungrammatical inputs;
- parsing and generation algorithms: apply mapping in either direction.

Analogy to (Object-Oriented) Programming

- Computational system with observable behavior: immediately testable;
- typed feature structures as a specialized (OO) programming language;
- make sure that all the pieces fit together; revise – test – revise – test ...

Course Organization

Comments on Background Literature

Formal Syntax

- Sag, Ivan A. Tom Wasow, and Emily M. Bender: *Syntactic Theory. A Formal Introduction (2nd Edition).* Stanford, CA: CSLI Publications (2003);
- Pollard, Carl and Sag, Ivan: *Head-Driven Phrase Structure Grammar.* Chicago, IL and London, UK: University of Chicago Press (1994).
- Shieber, Stuart: *An Introduction to Unification-Based Approaches to Grammar.* Stanford, CA: CSLI Publications (1986).

The Linguistic Knowledge Builder

- Copestake, Ann: *Implementing Typed Feature Structure Grammars.* Stanford, CA: CSLI Publications (2001).

Candidate Theories of Grammar (1 of 3)

Language as a Set of Strings

The dog barks.
The angry dog barks.
The fierce dog barks.
The fierce angry dog barks.
The angry fierce dog barks.
The dog chased a cat.
A dog chased the cat.
The dog chased a black cat.
The dog chased a young cat.
The dog of my neighbours chased a cat.
A dog chased the cat of my neighbours.
The cat of my neighbours was chased by a dog.
...

Candidate Theories of Grammar (2 of 3)

Language as a Sequence of Words

a, *the*, *my*, *that*, ...	**determiner (D)**
cat, *dog*, *neighbours*, ...	**noun (N)**
fierce, *angry*, *black*, *young*, ...	**adjective (A)**
barks, *chased*, *was*, ...	**verb (v)**
of, *by*, *on*, *at*, *under*, ...	**preposition (P)**

Regular Expressions

$$X^+ \equiv \{\, X \mid XX \mid XXX \mid XXXX \mid \ldots \,\}$$
$$X^* \equiv \{\, \_ \mid X \mid XX \mid XXX \mid XXXX \mid \ldots \,\}$$

$$\mathrm{D}\ \mathrm{A}^*\ \mathrm{N}^+\ \mathrm{V}\ \left(\mathrm{D}\ \mathrm{A}^*\ \mathrm{N}^+\right)^?$$

Candidate Theories of Grammar (3 of 3)

Review: Context-Free Grammars

- Formally, a *context-free grammar* (CFG) is a quadruple: $\langle C, \Sigma, P, S \rangle$
- C is the set of categories (aka *non-terminals*), e.g. $\{$S, NP, VP, V$\}$;
- Σ is the vocabulary (aka *terminals*), e.g. $\{$kim, snow, adores$\}$;
- P is a set of category rewrite rules (aka *productions*), e.g.

S $\rightarrow$ NP VP
VP $\rightarrow$ V NP
NP $\rightarrow$ kim
NP $\rightarrow$ snow
V $\rightarrow$ adores

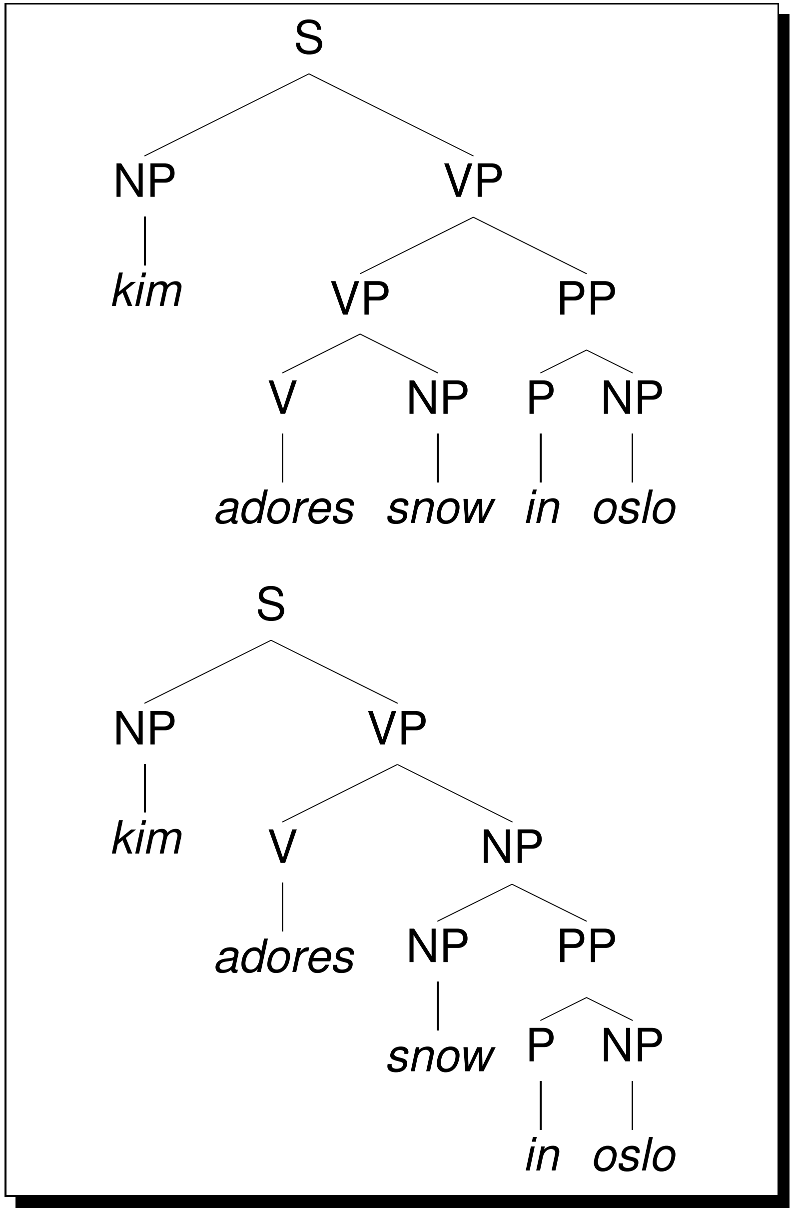
- $S \in C$ is the *start symbol*, a filter on complete (aka 'sentential') results;
- for each rule '$\alpha \rightarrow \beta_1, \beta_2, ..., \beta_n$' $\in P$: $\alpha \in C$ and $\beta_i \in C \cup \Sigma$; $1 \leq i \leq n$.

Recognizing the Language of a Grammar

S → NP VP
VP → V NP
VP → VP PP
NP → NP PP
PP → P NP
NP → kim | snow | oslo
V → snores | adores
P → in

All Complete Derivations

- are rooted in the start symbol S;
- label internal nodes with categories $\in C$, leafs with words $\in \Sigma$;
- instantiate a grammar rule $\in P$ at each local subtree of depth one.



Limitations of Context-Free Grammar

Agreement and Valency (For Example)

That dog barks.

**That dogs barks.*

**Those dogs barks.*

The dog chased a cat.

**The dog barked a cat.*

**The dog chased.*

**The dog chased a cat my neighbours.*

The cat was chased by a dog.

**The cat was chased of a dog.*

...