Chomsky Hierarchy

Background for NLP

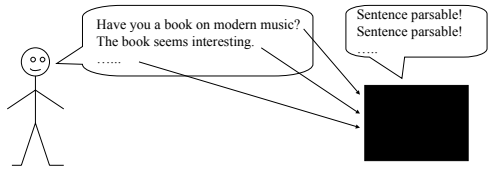
- Questions brought up by N. Chomsky in the 1950'ies:
  - Can a **natural language** like English be described ("parsed", "compiled") with the same methods as used for **formal/artificial** (programming) **languages** in computer science?
  - Can we use simple finite state grammars or context-free grammars for the description of English?
  - Or does linguistics need to invent an own and more powerful grammar type for the description of natural languages?
- Offshoots: "The Chomsky Hierarchy of Grammars", "Natural Language Processing", "Generative Transformational Grammar",

Chomsky: Grammar Theory 0

- Some key extracts/quotations from "Syntactic Structures"
  - A language is a (infinite) set of sentences, each finite in length and constructed out of a finite set of elements.
  - A grammar is a device that separate the grammatical sequences from the ungrammatical sentences and generates the structures of the grammatical ones.
  - A grammar is a reconstruction of the native speaker's competence, his ability to generate (produce and understand) an infinite number of sentences
  - A grammar is a **theory** of a language. It must comply with the empiristic axioms: The theory must be **adequate** and **simple**.

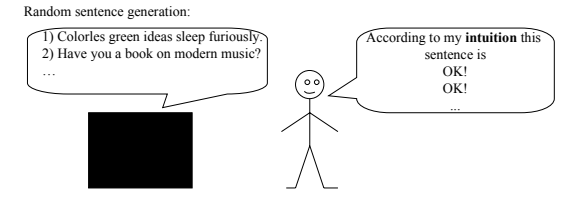
Chomsky: Grammar Theory 1

The grammar must generate ("parse") **ALL** sentences acceptable to the native speaker and ….



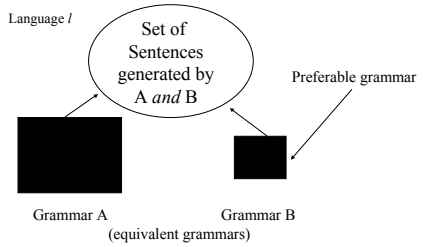
Chomsky: Grammar Theory 2

… the grammar must generate **NOTHING BUT** sentences acceptable to the native speaker and ...



Chomsky: Grammar Theory 3

… the grammar must be as **SIMPLE** (e.g. "small") as possible



Chomsky: Grammar Theory 4

What's in the "Black Box"? What *type* of grammar can "generate" a natural language like English?

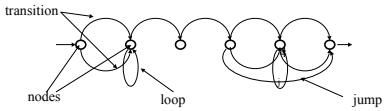
- A **Finite State Grammar** without/with loops?
  - (No! "Syntactic Structures" pp. 18 ff.)
- A **Phrase Structure Grammar**?
  - (No! "Syntactic Structures" pp. 26 ff.)
- A **Transformational Grammar**?
  - (Yes/Maybe! According to "Syntactic Structures" pp 34 ff. ***BUT*** *"Generative Transformational Grammar" has turned out to be a "blind alley" in computational linguistics)*

Chomsky: Hierarchy of Grammars

- Type 3: Regular Grammars
  - Equivalent to finite state automata, **finite state transition networks**, Markov models (probabilistic type).
- Type 2: Context free Grammars
  - E.g. recursive transition networks (RTNs), **phrase structure grammars** (PSGs). Unification grammars where attributes take values drawn from a finite table.
- Type 1: Context sensitive Grammars
  - Augmented transition networks (ATNs), **transformational grammars**, some unification grammars
- Type 0: Unrestricted Grammars

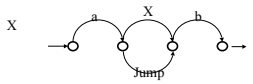
Finite State Grammar

- Structure:
  - Directed Graph/Transition Network structure
  - All transitions are **terminals**
  - The terminal symbols are either words or POS (word class) names like Noun, Verb, Pronoun.
  - The network structure may involve loops (iterations) and "empty" transitions (jumps, skips)



Recursive Transition Network Grammar

- Structure:
  - A **SET** of named Directed Graph/Transition Network structure**S**
  - Transitions are terminals or **NON-TERMINALS**
  - Terminal symbols/loops/jumps -> see FSN -slide
  - A non-terminal symbol is the name of a network in the set included in the RTN



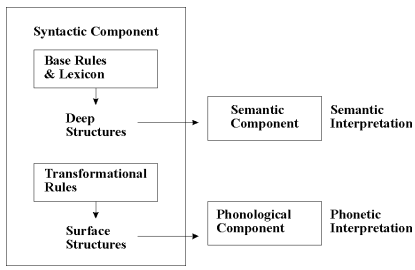
Equivalent BNF/PSG
X -> a b
X -> a X b

A^nB^n-problem, "Syntactic Structures", p. 30

What's wrong with FSNs & RTNs according to Chomsky?

- **FSNs without loops** can only generate a *finite* set of sentences. English is an *infinite* set
- **FSNs with loops** generate infinite sets of sentences but cannot describe A^nB^n sequences found in constructions with "respectively".
- **RTNs (PSGs/BNFs)** generate infinite sets of sentences, can describe A^nB^n sequences, but applied to English a huge number of symbols is required (conflict with simplicity)

Generative Transformational Grammar



Word Classes

Open classes

- nouns = *navneord*
  - common name = *fællesnavn*
  - proper name) = *egenavn*
- verbs = *udsagnsord*
- adjectives = *tillægsord*
- adverbs = *biord*

Closed classes

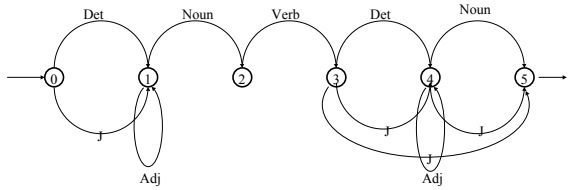
- prepositions = *forholdsord*
- determiners = *kendeord*
- pronouns = *stedord*
- conjunctions = *bindord*
- auxiliary verbs = *hjælpeverber/mådesudsagnsord*
- particles = *(verbal-)partikler*
- numerals = *talord*

Ambiguity

Lexical Ambiguity

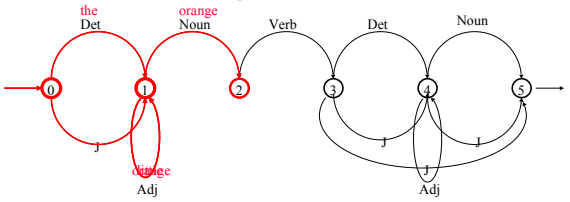
- Definition: a word belongs to two or more word ("part of speech") classes
- Example: the *round* table (adjective), to *round* the corner (verb), dance in a *round* (noun), come *round* and see us (adverb), he walked *round* the room (preposition)
- Finite state grammars can be used for resolving lexical ambiguity

Grammar: Simple Finite State Network



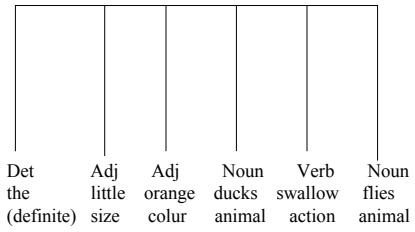
Lexicon:				
	the:	Det (definite)	ducks:	Noun (animal)
	little:	Adj (size)		Verb (action)
	orange:	Adj (colour)	swallow:	Noun (animal)
		Noun (fruit)		Verb (action)
			flies:	Noun (animal)
				Verb (action)

Grammar: Simple Finite State Network



Lexicon:				
	the:	Det (definite)	ducks:	Noun (animal)
	little:	Adj (size)		Verb (action)
	orange:	Adj (colour)	swallow:	Noun (animal)
		Noun (fruit)		Verb (action)
			flies:	Noun (animal)
				Verb (action)

Parsing with Finite State Grammar
"parse tree"

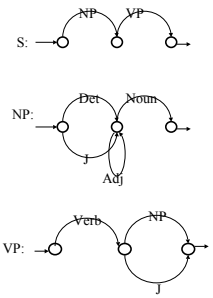


Finite State Grammar: Conclusion

- FSNs can resolve lexical ambiguities
- FSNs cannot assign actual structural descriptions to sentences. The generated structures are "flat" describing simple paths through the network.
- FSNs only describe legal sequences of terminal symbols

(Note: In NLP, syntactic parsing is sometimes preceded by "POS-tagging" (or "Constraint-grammars), a preprocessor that resolves many lexical ambiguities. This speeds up syntactic parsing. POS-tagging is normally based on trainable finite state machines).

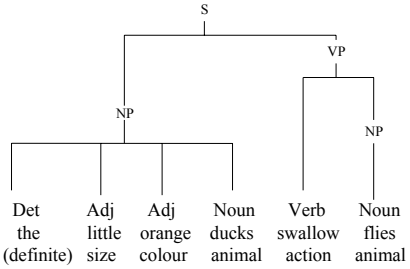
Grammar: Simple Recursive Transition Network
(equivalent to fsn)



Equivalent BNF:

S -> NP VP
NP ->Noun
NP -> Det Noun
NP -> AdjP Noun
NP -> Det AdjP Noun
AdjP -> Adj
AdjP-> adj AdjP
VP ->Verb
VP -> Verb NP

Parsing with Recursive Transition Network Grammar
Phrase Structure ("parse tree")



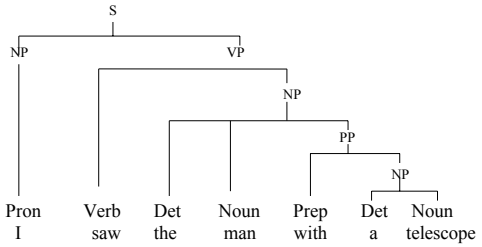
Structural Ambiguity

- Definition: a context-free grammar can assign two or more phrase structures ("parse trees") to one and the same sequence of terminal symbols (words or word classes).
- In formal language theory often referred to as the *grammar* being ambiguous (ambiguous vs. unambiguous grammars)
- Examples:
  - old men and women
  - time flies like an arrow
  - I saw the man with the telescope

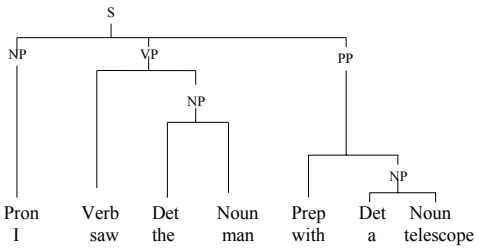
Structural Ambiguity 1
(ambiguous context-free grammars)

- S-> NP VP (PP) — Subject+Predicate+a facultative prepositional phrase describing e.g. instrument/time/place of the Subject-Predicate relation
- NP->Pron — I/me/him...
- NP->(Det) Noun (PP) — (the/a) man (in England/round the corner/with a hat)
- VP->Verb (NP) — eat (sth.)/see (sth.)
- PP-> Prep NP — in England/round the corner/with a hat/with a telescope

Structural Ambiguity 2
Phrase Structure 1.



Structural Ambiguity 3
Phrase Structure 2.



Context-free Grammar: Conclusion

- Like FSNs, RTNs/BNFs can resolve lexical ambiguities
- Additionally, RTNs/BNFs can assign actual structural descriptions to sentences. The generated structures analyses the sentence into constituents.
- A proper constituent analysis is a vital step towards an actual *semantic interpretation* of the sentence

In NLP, unification-based context-free grammars are often preferred because they can be used with a number of efficient parsing algorithms developed in formal-language theory. In general, such unification-grammars presuppose

- *general* parsing algorithms (no restrictions as regards left-recursion/right-recursion etc.)
- *exhaustive* parsing algorithms (because of ambiguities)

Widespread are algorithms derived from the *Earley chart parsing algorithm* (cf. J. Earley). Example at http://www.sil.org/pcpatr/

More about Ambiguity 1.

- How do humans resolve ambiguities?
  - Pragmatics:
    - Understanding intentions
    - World-knowledge
- Example:
  - "Denmark will have a distinguished visit next year. *The Russian president Boris Jeltsin and the American president Bill Clinton will attend a meeting on social problems in Copenhagen"*

  Translated from Danish:

  "Danmark får fornemt besøg næste år. Den russiske præsident Boris Jeltsin og den amerikanske præsident Bill Clinton skal til møde om sociale problemer i København"

More about Ambiguity 2.

- How are ambiguities resolved in NLP applications?
  - Implementation of less ambiguous domain-specific *sub-grammars*
  - Application of *preference rules*.
  - In spoken dialogue systems:
    - Implementation of *system-directed* dialogues (system-prompts)
    - Clarification sub-dialogues