# Shallow Analysis: Light Parsing, NER & Finite State Transducers

Stephan Busemann
LT lab, DFKI

(based on slides by
Günter Neumann, Steven Bird, Karin Haenelt)

---

| LT Components | Text → Meaning | Technologies |
|---|---|---|
| Lexical / Morphological Analysis | | OCR |
| Tagging | | Spelling Error Correction |
| Chunking | | Grammar Checking |
| Syntactic Analysis | | Information retrieval |
| Grammatical Relation Finding | | Document Classification |
| Named Entity Recognition | | Information Extraction |
| Word Sense Disambiguation | | Summarization |
| Semantic Analysis | | Question Answering |
| Reference Resolution | | Ontology Extraction and Refinement |
| Discourse Analysis | | Dialogue Systems |
| | | Machine Translation |

| LT Components | Text | Technologies |
|---|---|---|
| Lexical / Morphological Analysis | | OCR |
| | | Spelling Error Correction |
| Shallow Parsing | | Grammar Checking |
| | | Information retrieval |
| Named Entity Recognition | | Document Classification |
| | | Information Extraction |
| Word Sense Disambiguation | | Summarization |
| | | Question Answering |
| Semantic Analysis | | Ontology Extraction and Refinement |
| Reference Resolution | | Dialogue Systems |
| Discourse Analysis | | Machine Translation |
| | Meaning | |

# From POS tagging to Information Extraction

- POS tagging
  The/Det woman/NN will/MD give/VB Mary/NNP a/Det book/NN
- NP chunking
  The/NP1 woman/NP1 will/VP1 give/VP1 Mary/NP2 a/NP3 book/NP3
- Relation Finding
  [NP1-SUBJ the woman ] [VP1 will give ] [NP2-OBJ1 Mary]
  [NP3-OBJ2 a book ]]
- Semantic Tagging = Information Extraction
  [GIVER the woman][will give][GIVEE Mary][GIVEN a book]
- Semantic Tagging = Question Answering
  Who will give Mary a book?
  [GIVER ?][will give][GIVEE Mary][GIVEN a book]

## Parsing of unrestricted text

- Complexity of parsing of unrestricted text
  - Robustness
  - Large sentences
  - Large data sources
  - Input texts are not simply sequences of word forms
    - Textual structure (e.g., enumeration, spacing, etc.)
    - Combined with structual annotation (e.g., XML tags)

## Motivations for Parsing

- Why parse sentences in the first place?
- Parsing is usually an intermediate stage
  - Builds structures that are used by later stages of processing
- Full Parsing is a sufficient but not necessary intermediate stage for many NLP tasks.
- Parsing often provides more information than we need.

## Light Parsing

Goal: assign a partial structure to a sentence.

- Simpler solution space
- Local context
- Non-recursive
- Restricted (local) domain

## Output from Light Parsing

- What kind of partial structures should light parsing construct?
- Different structures useful for different tasks:
  - Partial constituent structure
    [NP I] [VP saw [NP a tall man in the park]].
  - Prosodic segments
    [I saw] [a tall man] [in the park].
  - Content word groups
    [I] [saw] [a tall man] [in the park].

## Chunk Parsing

Goal: divide a sentence into a sequence of chunks.

- Chunks are non-overlapping regions of a text

  [I] saw [a tall man] in [the park]

- Chunks are non-recursive
  - A chunk can not contain other chunks
- Chunks are non-exhaustive
  - Not all words are included in the chunks

## Chunk Parsing Examples

- Noun-phrase chunking:
  - [I] saw [a tall man] in [the park].
- Verb-phrase chunking:
  - The man who [was in the park] [saw me].
- Prosodic chunking:
  - [I saw] [a tall man] [in the park].

## Chunks and Constituency

Constituents: [[a tall man] [ in [the park]]].
Chunks: [a tall man] in [the park].

- A constituent is part of some higher unit in the hierarchical syntactic parse
- Chunks are not constituents
  - Constituents are recursive
- But, chunks are typically sub-sequences of constituents
  - Chunks do not cross major constituent boundaries

1. [$_{NP}$ [$_{NP}$ G.K. Chesterton ], [$_{NP}$ [$_{NP}$ author ] of [$_{NP}$ [$_{NP}$ The Man ] who was [$_{NP}$ Thursday ] ] ] ]
2. [$_{NP}$ G.K. Chesterton ], [$_{NP}$ author ] of [$_{NP}$ The Man ] who was [$_{NP}$ Thursday ]

## Chunk Parsing: Accuracy

- Chunk parsing achieves higher accuracy than full parsing
- Smaller solution space
- Less word-order flexibility within chunks than between chunks
  - Fewer long-range dependencies
  - Less contextual dependence
- Better locality
- No need to resolve ambiguity
- Less error propagation

## Chunk Parsing: Efficiency

Chunk parsing is more efficient than full parsing

- Smaller solution space
- Relevant context is small and local
- Chunks are non-recursive
- Chunk parsing can be implemented with a finite state machine
  - Fast (linear)
  - Low memory requirement (no stacks)
- Chunk parsing can be applied to a very large text sources (e.g., the web)

## Chunk Parsing Techniques

- Chunk parsers usually ignore lexical content
- Only need to look at part-of-speech tags
- Techniques for implementing chunk parsing
  - Regular expression matching
  - Chinking
  - Cascaded Finite state transducers

## Regular Expression Matching

- Define a regular expression that matches the sequences of tags in a chunk
  - A simple noun phrase chunk regexp:
    - <DT> ? <JJ> * <NN.?>
- Chunk all matching subsequences:
    - In:
      The /DT little /JJ cat /NN sat /VBD on /IN the /DT mat /NN
    - Out:
      [The /DT little /JJ cat /NN] sat /VBD on /IN [the /DT mat /NN]
- If matching subsequences overlap, the first one gets priority
- Regular expressions can be cascaded

## Chinking

- A chink is a subsequence of the text that is not a chunk.
- Define a regular expression that matches the sequences of tags in a chink.
  - A simple chink regexp for finding NP chunks:
    (<VB.?> | <IN>)+
- Chunk anything that is not a matching subsequence:

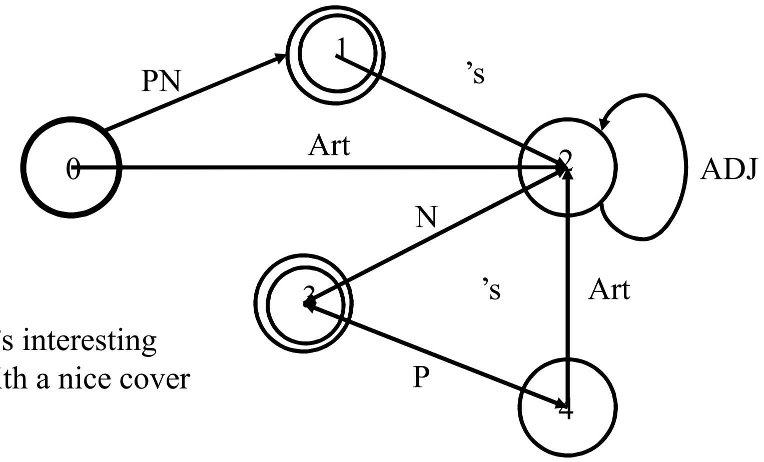
  the/DT little/JJ cat/NN sat/VBD on /IN the /DT mat/NN

  [the/DT little/JJ cat/NN] <u>sat/VBD on /IN</u> [the /DT mat/NN]

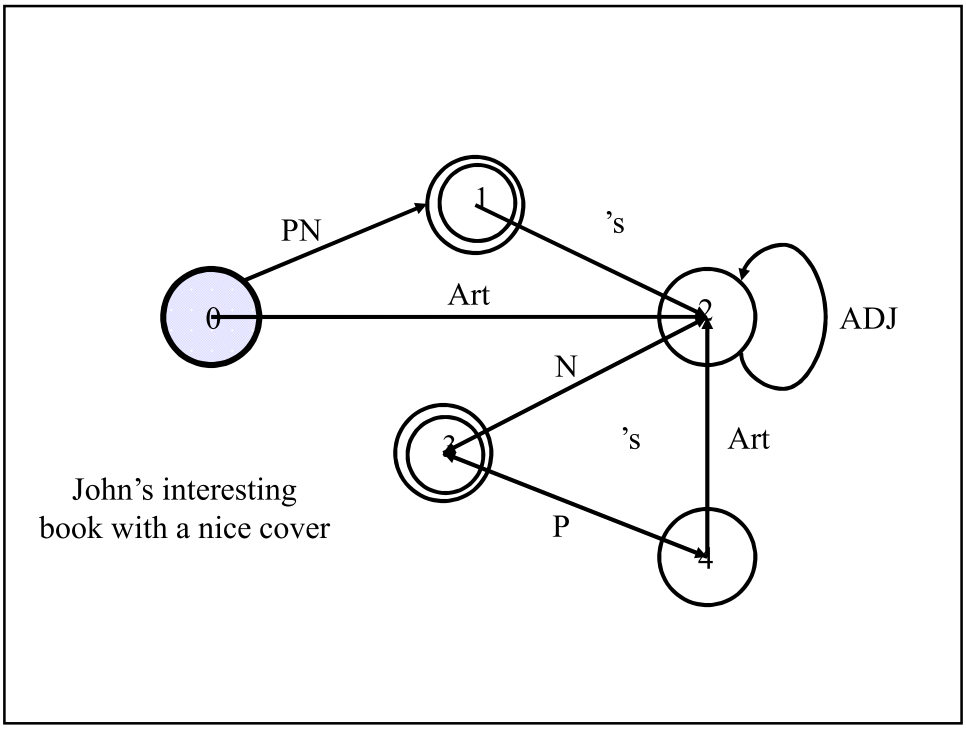
  chunk chink chunk

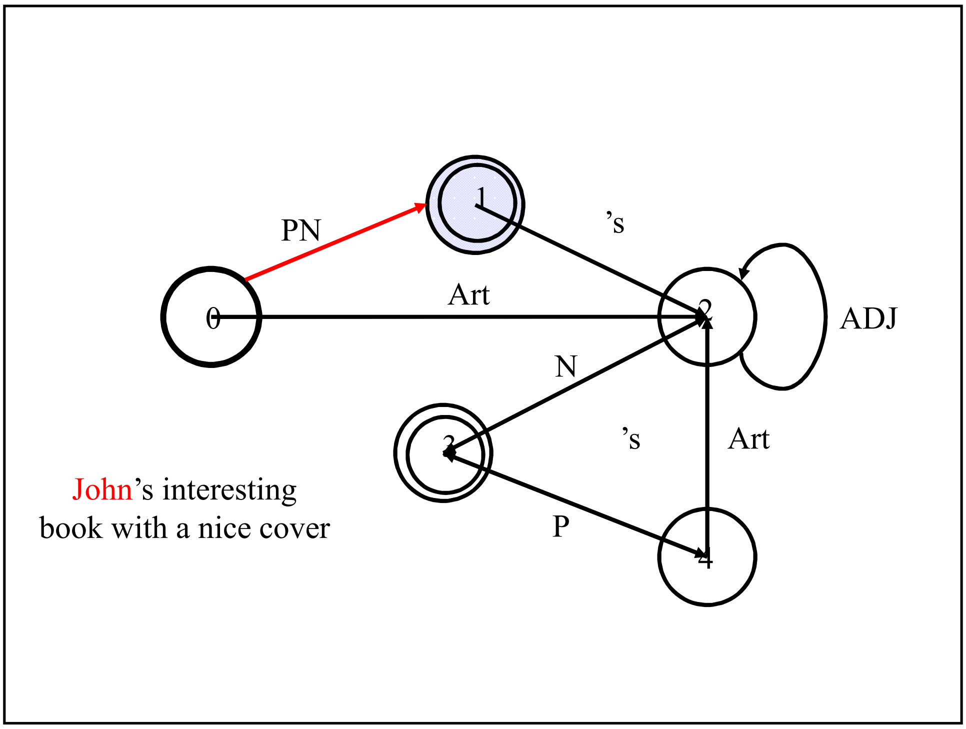
# Syntactic Structure: Partial Parsing Approaches

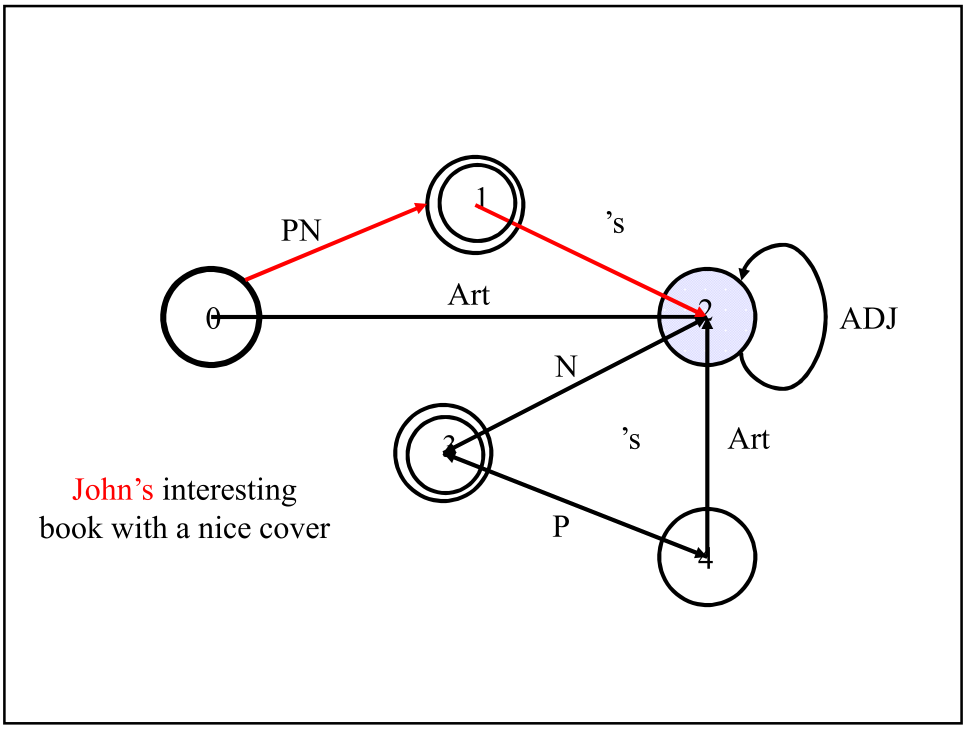
- Finite-state approximation of sentence structures (Abney 1995)
  - finite-state cascades: sequences of levels of regular expressions
  - recognition approximation: tail-recursion replaced by iteration
  - interpretation approximation: embedding replaced by fixed levels
- Finite-state approximation of phrase structure grammars (Pereira/Wright 1997)
  - flattening of shift-reduce-recogniser
  - no interpretation structure (acceptor only)
  - used in speech recognition where syntactic parsing serves to rank hypotheses for acoustic sequences
- Finite-state approximation (Sproat 2002)
  - bounding of centre embedding
  - reduction of recognition capacity
  - flattening of interpretation structure

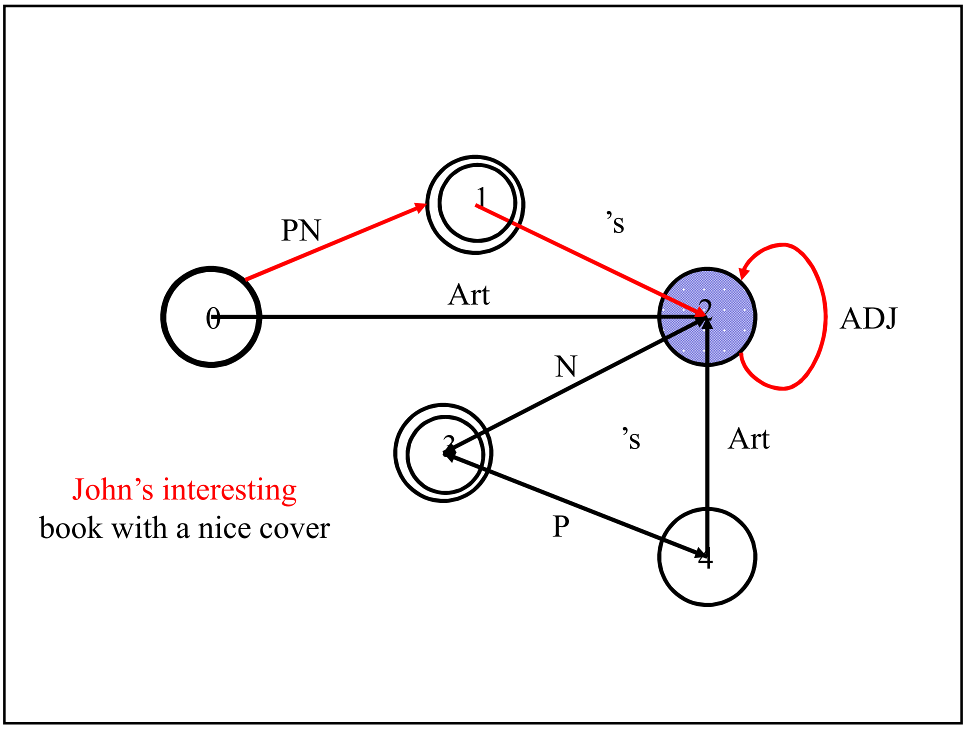


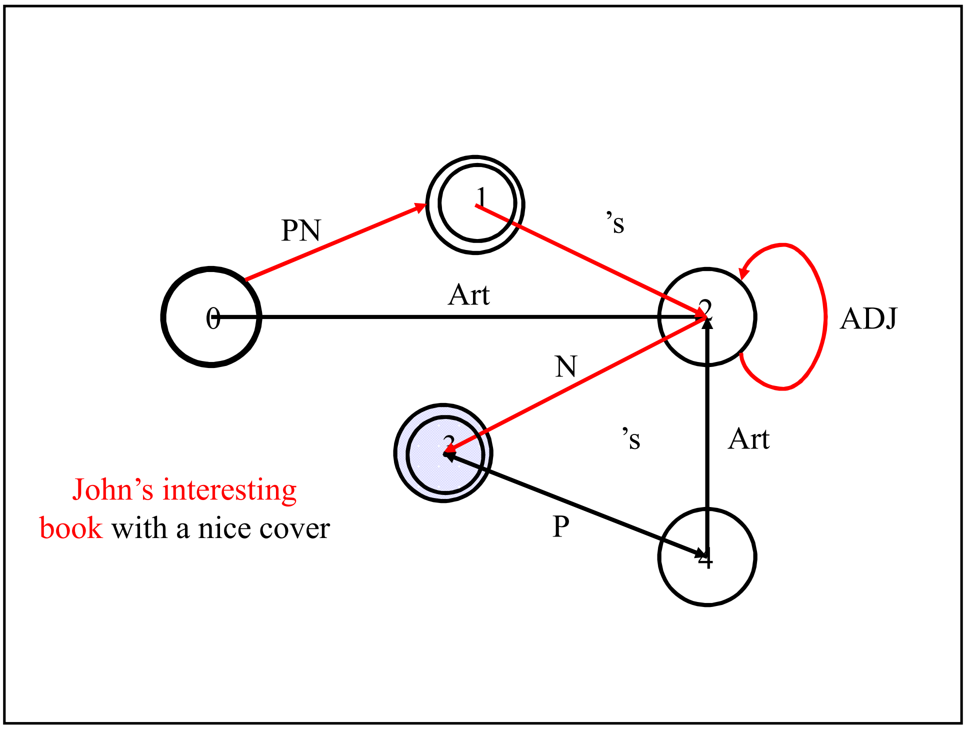

John's interesting
book with a nice cover

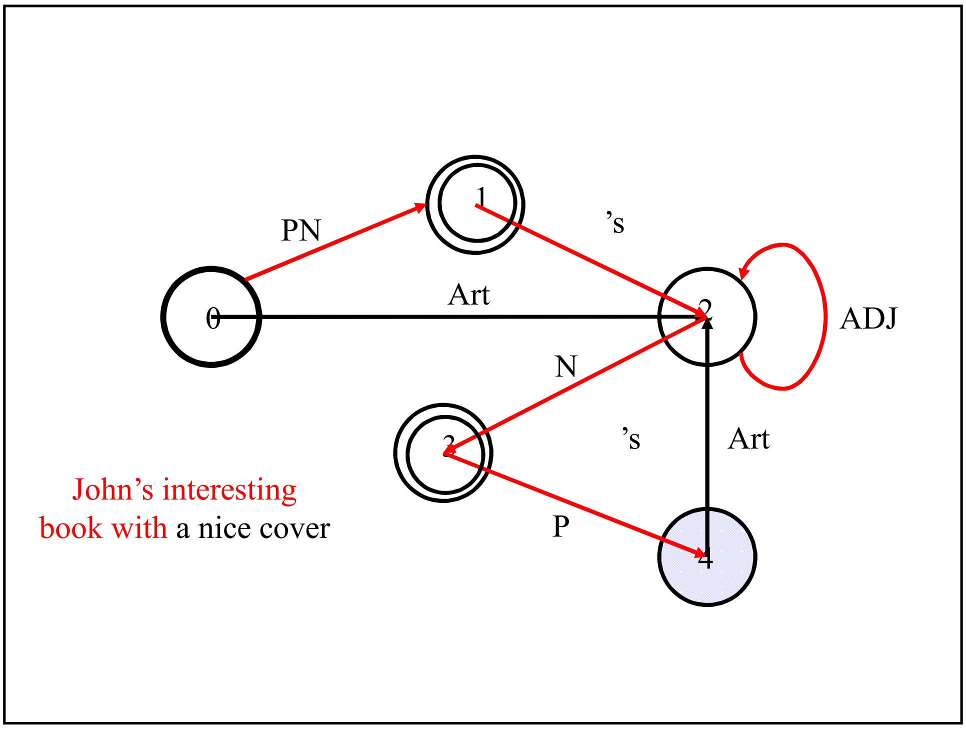

PN
1
's
0
Art
2
ADJ
N
3
's
Art
P
4
John's interesting
book with a nice cover

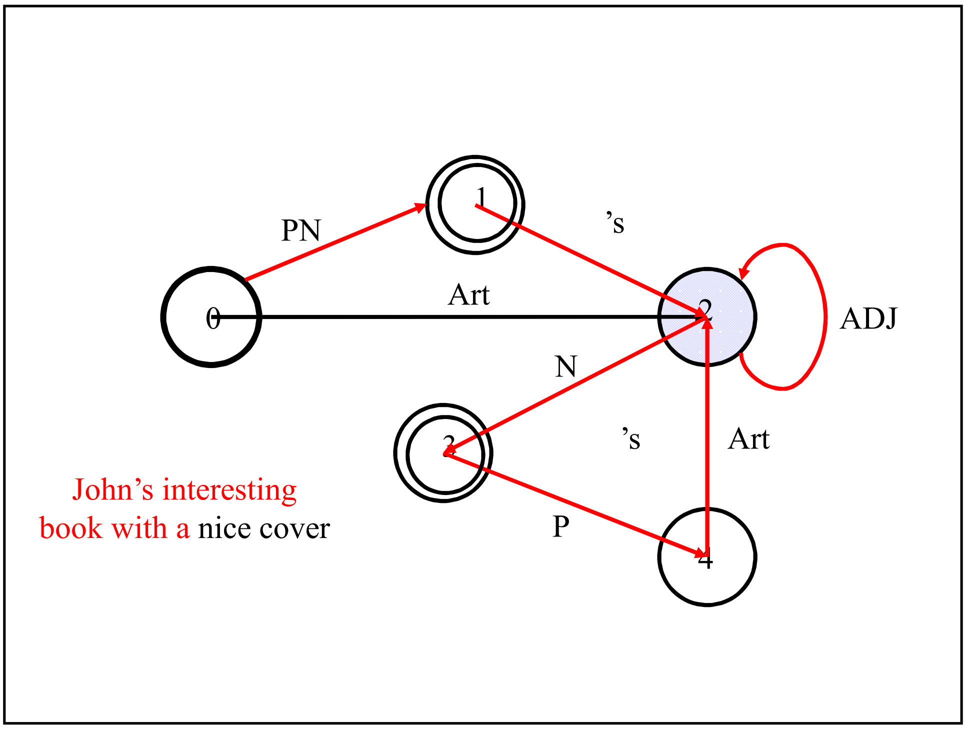

PN
1
's
0
Art
2
ADJ
N
3
's
Art
P
4
John's interesting
book with a nice cover

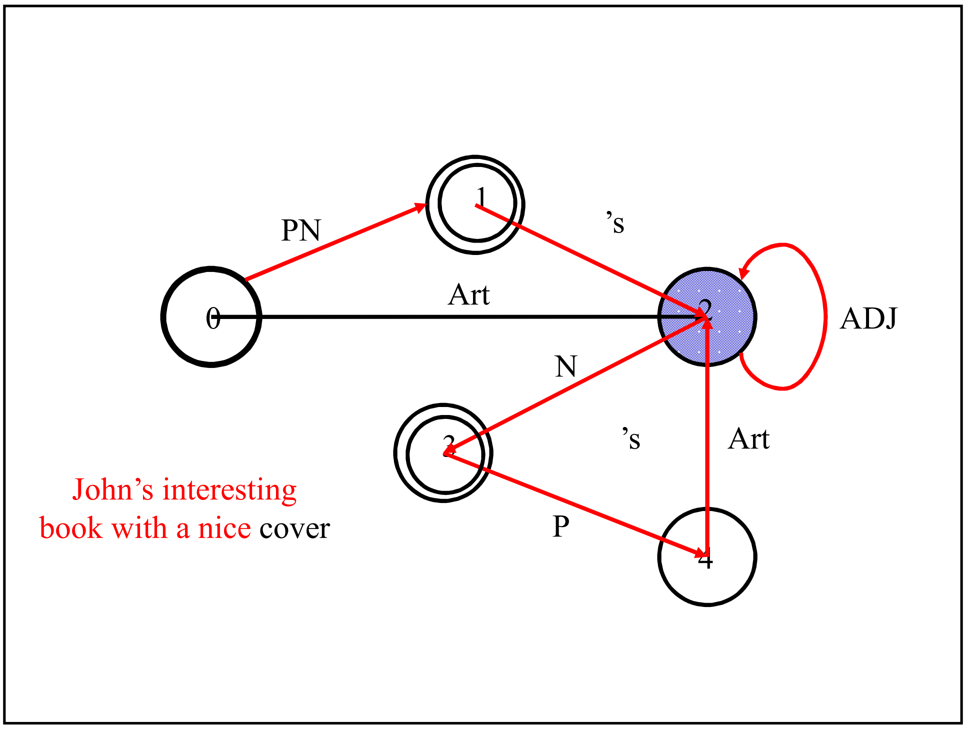

PN
1
's
0
Art
2
ADJ
N
3
's
Art
P
4
John's interesting
book with a nice cover

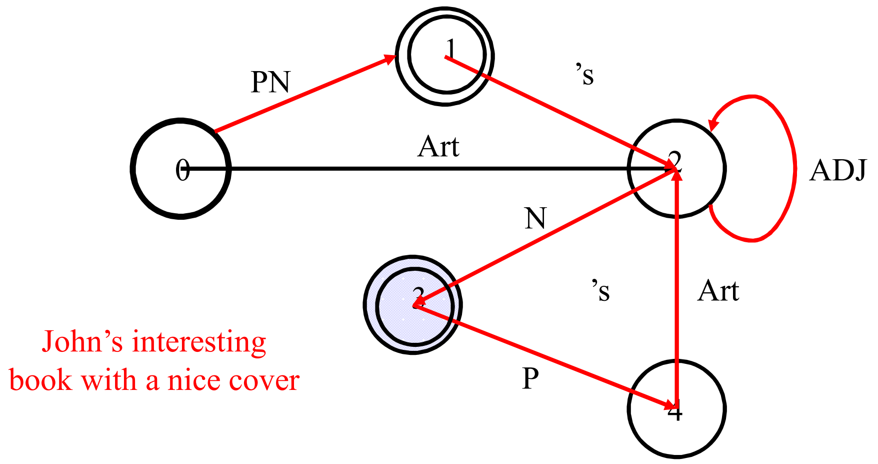

PN
1
's
0
Art
2
ADJ
N
3
's
Art
P
4
John's interesting
book with a nice cover


PN
1
's
0
Art
2
ADJ
N
3
's
Art
John's interesting
book with a nice cover
P
4


PN
1
's
0
Art
2
ADJ
N
3
's
Art
John's interesting
book with a nice cover
P
4

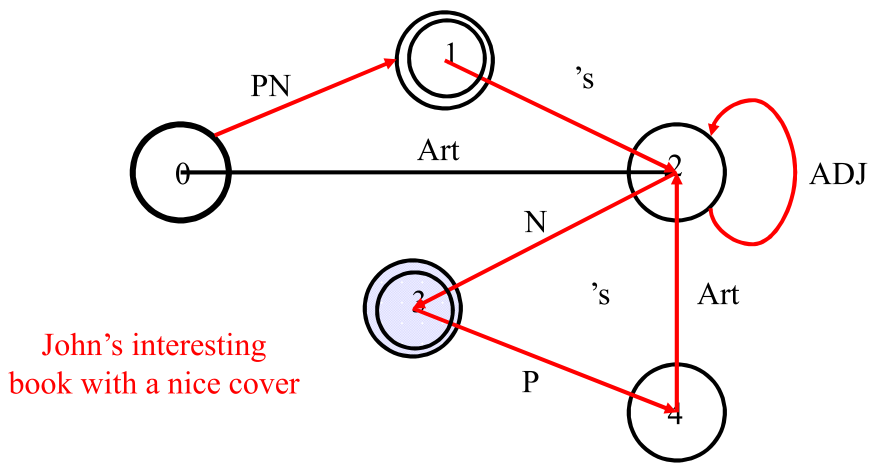

PN
1
's
0
Art
2
ADJ
N
2
's
Art
John's interesting
book with a nice cover
P
4


PN
1
's
0
Art
2
ADJ
N
2
's
Art
John's interesting
book with a nice cover
P
4



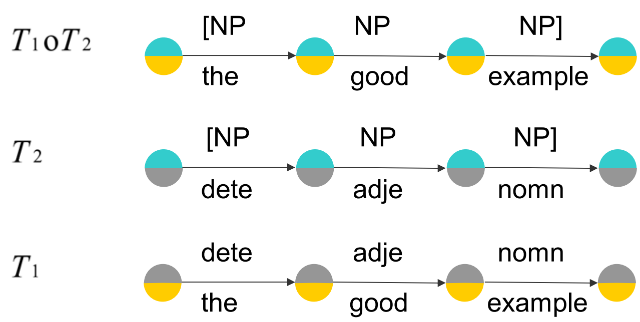

Pattern-maching

PN 's (ADJ)* N P Art (ADJ)* N



# Syntactic Structure: Finite State Cascades

- functionally equivalent to composition of transducers,
  - but without intermediate structure output
  - the individual transducers are considerably smaller than a composed transducer




# Syntactic Structure: Finite-State Cascades (Abney)

Finite-State Cascade

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $L_3$ ---- | S | | | | | | | S | | |
| $T_3$ | | | | | | | | | | |
| $L_2$ ---- | NP | | PP | | | | VP | NP | VP | |
| $T_2$ | | | | | | | | | | |
| $L_1$ ---- | NP | | P | NP | | | VP | NP | VP | |
| $T_1$ | | | | | | | | | | |
| $L_0$ ---- | D | N | P | D | N | N | V-tns | Pron | Aux | V-ing |
| | the | woman | in | the | lab | coat | thought | you | were | sleeping |

Regular-Expression Grammar

$$L_1 : \left\{ \begin{array}{l} NP \rightarrow D?\,N*\,N \\ VP \rightarrow V - tns \mid Aux\,V\text{-}ing \end{array} \right\}$$

$$L_2 : \{PP \rightarrow P\ NP\}$$

$$L_3 : \{S \rightarrow NP\ PP*\ VP\ PP*\}$$

# Syntactic Structure:
# Finite-State Cascades (Abney)

- A cascade consists of a sequence of levels
- Phrases at one level are built on phrases at the previous level
- No recursion:
  - phrases never contain same level or higher level phrases
- Two levels of special importance
  - chunks: non-recursive cores (NX, VX) of major phrases (NP, VP)
  - simplex clauses: embedded clauses as siblings
- Patterns:
  - reliable indicators of bits of syntactic structure

# An alternative FST cascade for German (free word order), Neumann et al.

## Most partial parsing approaches following a bottom-up strategy:

Major steps

lexical processing

including morphological analysis, POS-tagging, Named Entity recognition

phrase recognition

general nominal and prepositional phrases, verb groups

clause recognition via domain-specific templates

templates triggered by domain-specific predicates attached to relevant verbs;

expressing domain-specific selectional restrictions for possible argument fillers

Bottom-up chunk parsing

perform clause recognition after phrase recognition is completed

## However a bottom-up strategy showed to be problematic in case of German free text processing

### Crucial properties of German

1. highly ambiguous morphology (e.g., case for nouns, tense for verbs)
2. free word/phrase order
3. splitting of verb groups into separated parts into which arbitrary phrases an clauses can be spliced in (e.g., Der Termin findet morgen statt. The date takes place tomorrow.)

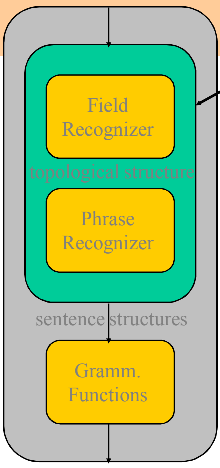
Main problem in case of a bottom-up parsing approach:
Even recognition of simple sentence structure
depends heavily on performance of phrase recognition.

NP ist gängige Praxis.
[NP Die vom Bundesgerichtshof und den Wettbewerbern als Verstoß gegen das Kartellverbot gegeißelte zentrale TV-Vermarktung] ist gängige Praxis.

NP ist gängige Praxis.
[NP Central television marketing criticized by the German Federal High Court and the guards against unfair competition as being an infringement of anti-cartel legislation] is common practice.

## In order to overcome these problems we propose the following two phase divide-and-conquer strategy

Text (morph. analysed)




Fct. descriptions

Divide-and-conquer strategy

1. Recognize verb groups and topological structure (fields) of sentence domain-independently;

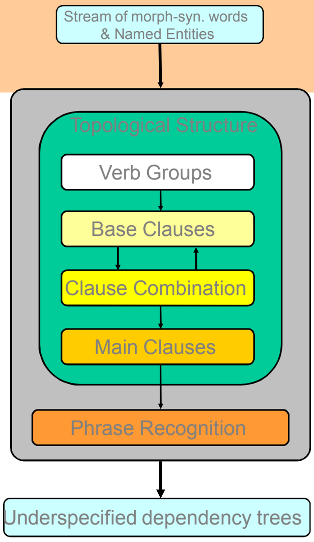
FrontField Vfin MiddleField Vinfin RestField

2. Apply general as well as domain-dependent phrasal grammars to the identified fields of the main and sub-clauses

[CoordS [CSent Diese Angaben konnte der Bundesgrenzschutz aber nicht bestätigen], [CSent Kinkel sprach von Horrorzahlen, [Relcl denen er keinen Glauben schenke]]].

This information couldn't be verified by the Border Police, Kinkel spoke of horrible figures that he didn't believe.

## The divide-and-conquer parser is realized by means of a cascade of finite state grammars




Weil die Siemens GmbH, die vom Export lebt, Verluste erlitt, mußte sie Aktien verkaufen.

[Because the Siemens Corp which strongly depends on exports suffered from losses they had to sell some shares.]

Weil die Siemens GmbH, die vom Export Verb-FIN, Verluste Verb-FIN, Modv-FIN sie Aktien FV-Inf.

Weil die Siemens GmbH, Rel-Clause Verluste Verb-FIN, Modv-FIN sie Aktien FV-Inf.

Subconj-Clause,
Modv-FIN sie Aktien FV-Inf.

Clause

## Named Entity Extraction – The who, where, when & how much in a sentence

- The task: identify lexical and phrasal information in text which express references to named entities NE, e.g.,
  - person names
  - company/organization names
  - locations
  - dates&times
  - percentages
  - monetary amounts
- Determination of an NE
  - Specific type according to some taxonomy
  - Canonical representation (template structure)

## Example of NE-annotated text

Delimit the named entities in a text and tag them with NE types:

<ENAMEX TYPE=„LOCATION“>Italy</ENAMEX>‘s business world was rocked by the announcement <TIMEX TYPE=„DATE“>last Thursday</TIMEX> that Mr. <ENAMEX TYPE=„PERSON“>Verdi</ENAMEX> would leave his job as vice-president of <ENAMEX TYPE=„ORGANIZATION“>Music Masters of Milan, Inc</ENAMEX> to become operations director of <ENAMEX TYPE=„ORGANIZATION“>Arthur Andersen</ENAMEX>.

- „Milan“ is part of organization name
- „Arthur Andersen“ is a company
- „Italy“ is sentence-initial – capitalization useless

## NE and Question-Answering

- Often, the expected answer type of a question is a NE
  - What was the name of the first Russian astronaut to do a spacewalk?
    - Expected answer type is PERSON
  - Name the five most important software companies!
    - Expected answer type is a list of COMPANY
  - Where is does the ESSLLI 2004 take place?
    - Expected answer type is LOCATION (subtype COUNTRY or TOWN)
  - When will be the next talk?
    - Expected answer type is DATE

## Difficulties of Automatic NEE

- NEs can't be enumerated in order to include them in dictionaries/Gazetteers
- Names are changing constantly
- Names vary in form
- Subsequent occurrences of names might be abbreviated

➢ list search/matching does not perform well
➢ context based pattern matching needed

## Difficulties for Pattern Matching Approach

Whether a phrase is a named entity, and what name class it has, depends on

- Internal structure:
  „Mr. Brandon“
- Context:
  „The new company, SafeTek, will make air bags.“
  „Feiyu Xu, researcher at DFKI, Berlin”

# NEE is an interesting problem

- Productivity of name creation requires lexicon free pattern recognition
- NE ambiguity requires resolution methods
- Fine-grained NE classification needs fined-grained decision making methods
  - Taxonomy learning
- Multi-linguality
  - A text might contain NE expressions from different languages, e.g., output of IdentiFinder™
    Pilot challenge in ACE'2007
    - Extract all NEs mentioned in a Mandarin/Arabic text
    - Translate them into English

# Statistical and Rule-Based NEE

- Identify a type of NE from charpos S to charpos E
  - Giuseppe Verdi, Italian composer of „Aida"
    <NE TYPE=„PERSON">Giuseppe Verdi</NE>
- Recognize structured entities with rule-based systems
  - Prof. Dr. Wolfgang Wahlster, CEO of DFKI GmbH

```
<NE TYPE=„PERSON“>
  <ELTS> <NE TYPE=„TITLE“>Prof. Dr. </NE>
         <NE TYPE=„FIRSTNAME“>Wolfgang</NE>
         <NE TYPE=„LASTNAME“>Wahlster</NE>
         <NE TYPE=„FUNCTION“>CEO</NE>
         <NE TYPE=„COMPANY“>
           <ELTS> <NE TYPE „NAME“>DFKI</NE>
                  <NE TYPE „DGNR“>GmbH</NE>
           </ELTS></NE>
  </ELTS> </NE>
```