Natural Language Processing (NLP)

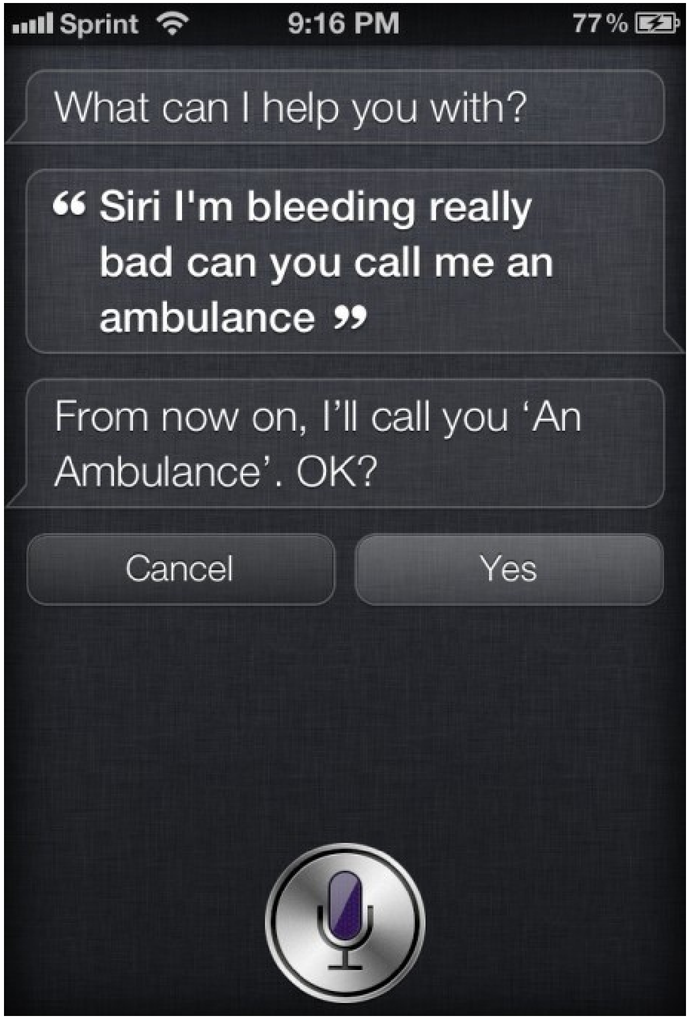
Goal: Understand the meaning of natural language

Applications

- Information retrieval
- Machine translation
- Dialogue systems

Example: IBM Watson in quiz show

NLP is difficult



What can I help you with?

“ Siri I'm bleeding really bad can you call me an ambulance ”

From now on, I’ll call you ‘An Ambulance’. OK?

Overview

- Relations for knowledge representation
- Context free grammars
- Prolog

tuProlog

tuProlog runs on Java 7 & .NET

To install:

- Go to http://code.google.com/p/tuprolog/downloads/list
- Download and unzip 2p-2.7.0.zip (for Java)

  or 2p.NET-2.6.0.zip (for .NET)
- Optional: Install Eclipse Indigo plugin from

  http://tuprolog.googlecode.com/svn/2p-plugin/trunk/

  alice.tuprologx.eclipse.updatesite/

  (see http://apice.unibo.it/xwiki/bin/view/Tuprolog/

  EclipsePluginInstructions)

- Add tuProlog-directory to classpath
- Invoke GUI with

  java -jar 2p.jar

  (or java -cp <tuProlog dir> -jar 2p.jar)

Motivation

How to analyze

“bring me the book”?

```
if(word1 == “bring” && word4 == “book”) ...
```

“get me the keys”

```
if( word1 == “bring” || word1 == “get” ) {
    if(word4 == “book”) …
    else if(word4 == “keys”) …
    ...
```

What do do with

- “bring me the blue book”?
- “please bring me the book on the table”?
- “bring me the book and my glasses”?
- “bring my cat the food”?

Context free grammars

Grammar: Set of replacement rules

- Nonterminals: Can be replaced
- Terminals: Can't be replaced
- Rules: *Nonterminal* → *Replacement*

Example

- Terminals = {a, the, cat, sleeps, eats}
- Nonterminals = {S, A, N, V}
- Rules:

  S → A N V

  A → a | the

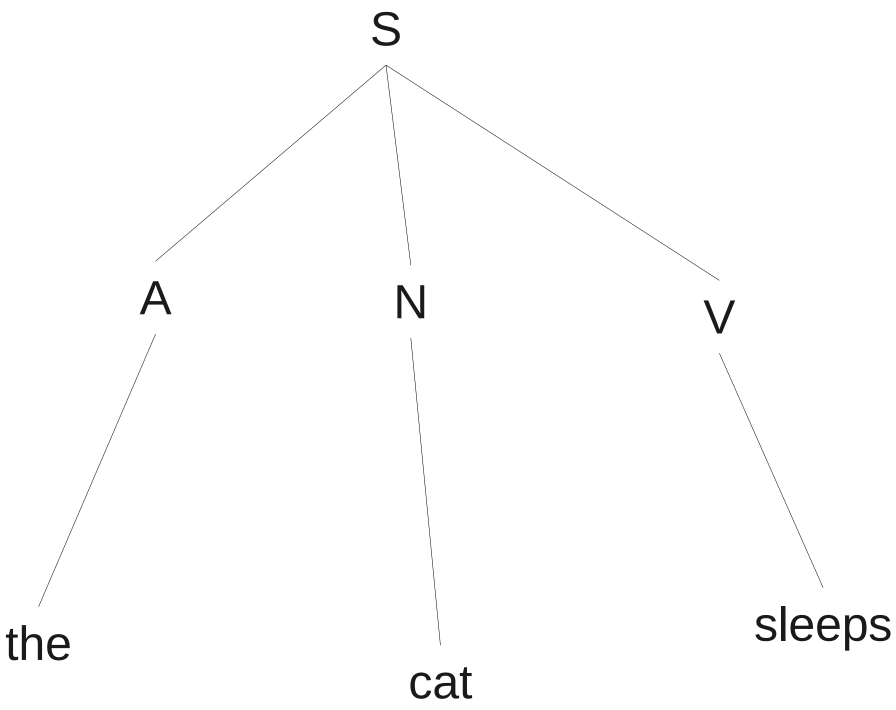
  N → cat

  V → sleeps | eats

We can derive

- S => A N V => a N V => a cat V => a cat sleeps
- S => A N V => the N V => the cat V => the cat eats

Syntax tree:



Exercise

What do the following grammars produce?

- S → a S b | a b
- S → S S | (S) | ( )

Grammar of natural language

Sentence → Nounphrase Verbphrase

Nounphrase → Article Noun

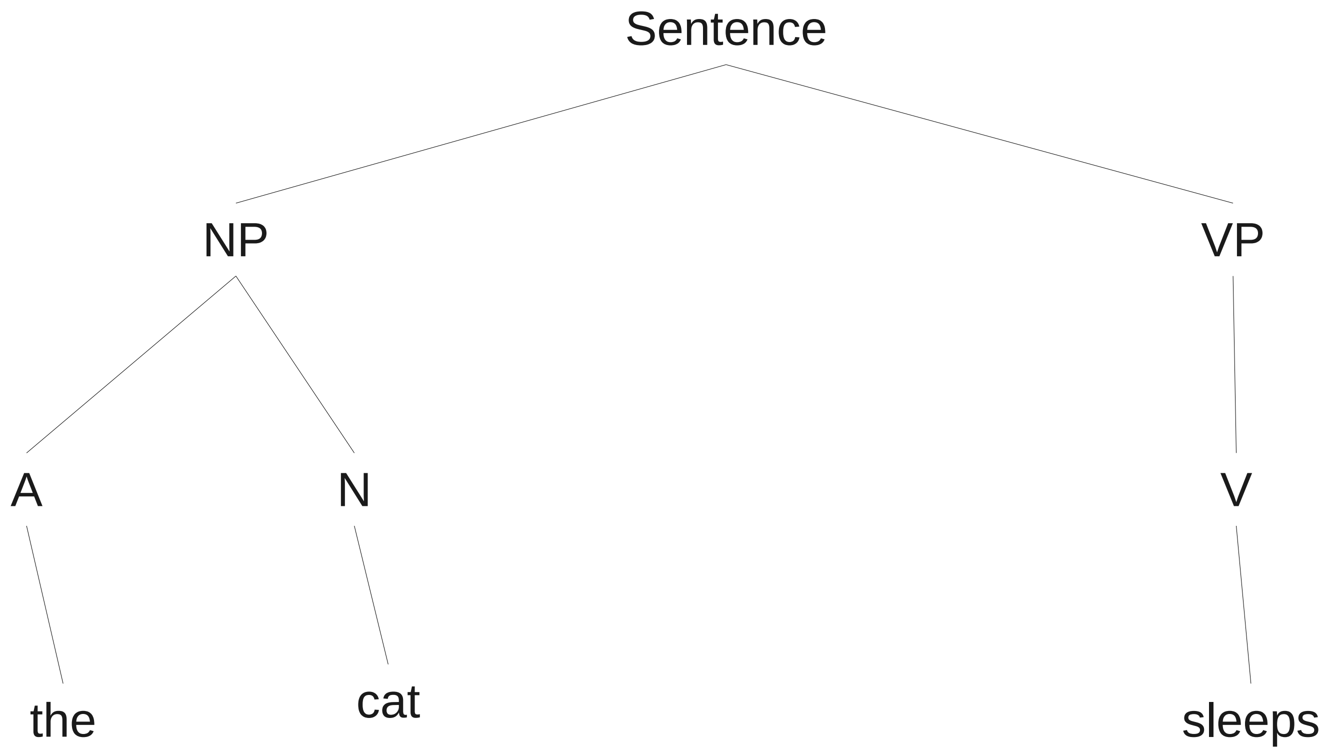
Article → a | the

Noun → cat | mouse | bird

Verbphrase → Verb

Verb → sleeps | eats

Syntax tree:



Verb valency

Intransitve verbs have no object:

- The cat sleeps
- A bird flies

Transitve verbs need an object:

- The cat eats a mouse
- The birds sees the cat

Extending the grammar

Sentence → Nounphrase Verbphrase

Nounphrase → Article Noun

Article → a | the

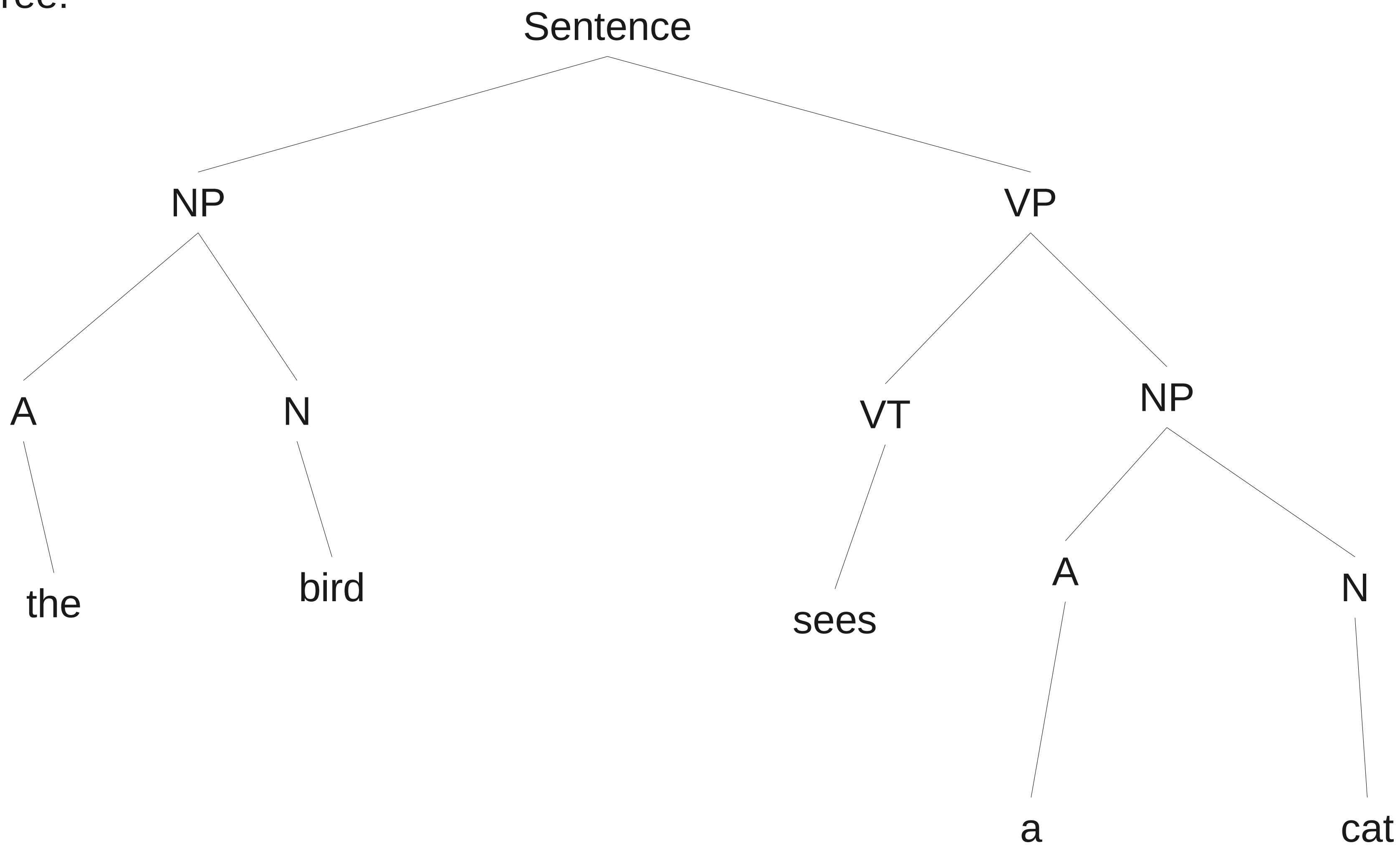
Noun → cat | mouse | bird

Verbphrase → VerbIt | VerbT Nounphrase

VerbIt → sleeps | eats

VerbT → eats | sees

Syntax tree:



Grammars in tuProlog

dcg1.pl

Exercise

Extend dcg1.pl with adjectives

Check that you can derive

- [the, cat, sleeps]
- [the, black, cat, sleeps]

Prepositional phrases

Preposition: on, with, at, …

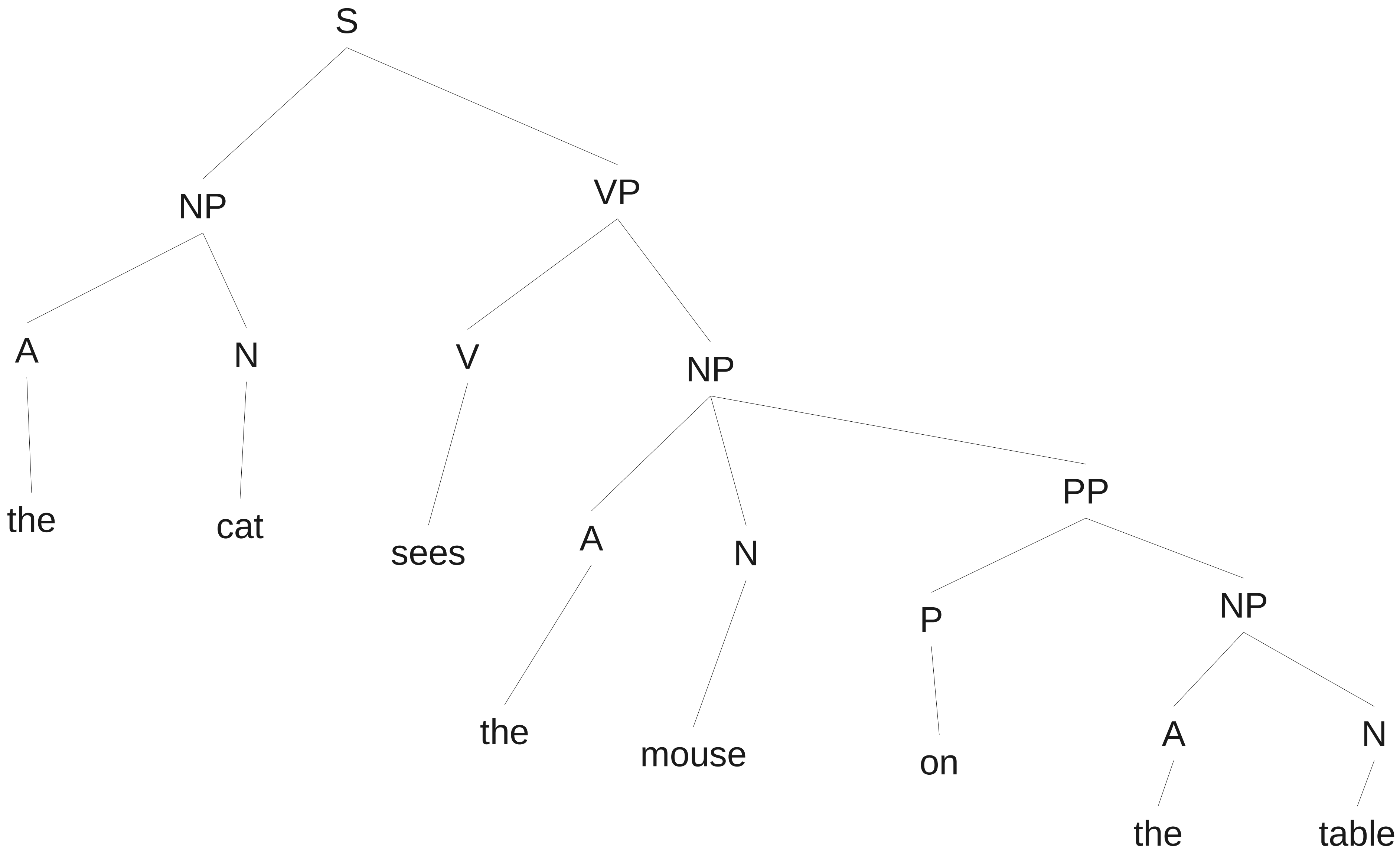
- Structure:

prepositional phrase →

preposition noun_phrase

- A noun phrase can be followed by a prepositional phrase

Example



Exercise

Augment dcg1.pl with prepositional phrases

Derive

[the, cat, sees, the, mouse, on, the, table]

Variables

- Start with uppercase letter
- Are bound to values in a query

Example:

- phrase(sentence, [the, X, sleeps]).
- phrase(sentence, X).

Identifying parts of sentences

How to identify the

- Subject (who is acting)
- Action (what is the subject doing)
- Object, if any?

Parameters of grammars

dcg2.pl

dcg3.pl

Imperative sentences

Word order as in declarative sentence with subject removed

Example:

- The cat hunts the mouse.
- Hunt the mouse!

Question sentences

Usually start with **wh**

Example:

- The cat hunts the mouse.
- **Wh**at does the cat hunt?

Running tuProlog from Java

Dcg1.java

Dcg2.java

Knowledge representation

How can we represent

- Tim studies engineering
- Reni is a cat
- Reni likes sheba

Relations in Prolog

We can define relations (or predicates):

- studies(tim, engineering).
- cat(reni).
- likes(reni, sheba).

Querying the knowledge base

- Yes/no question:

  cat(reni).

  → Yes.

- Searching a solution:

  cat(X).

  → X / reni.

Rules

Operator :- (“is implied by”)

Examples:

- Every cat is an animal

```
animal(X) :- cat(X).
```

- All cats like Sheba

```
likes(X, sheba) :- cat(X).
```

- If it meows and has four legs, then it's a cat.

```
cat(X) :- meows(X), four_legged(X).
```

Exercise

Represent the facts

- Reni, Mimi, Momo are cats
- Whiskas, Sheba is cat food
- All cats eat cat food

Verify that Reni eats Sheba

Simple dialogue system

studies.pl