Grammatical inference: an introduction

Module X3IT040, Colin de la Higuera, Nantes & Le Mans, 2020

UNIVERSITÉ DE NANTES

Statistical and symbolic language modeling

ATAL

LS2N LABORATOIRE DES SCIENCES DU NUMÉRIQUE DE NANTES

Outline (of this first talk)

1. What is grammatical inference about?
2. Why is it a difficult task?
3. Why is it a useful task?
4. Validation issues
5. Some criteria

1. What is grammatical inference about?

1. Grammatical inference

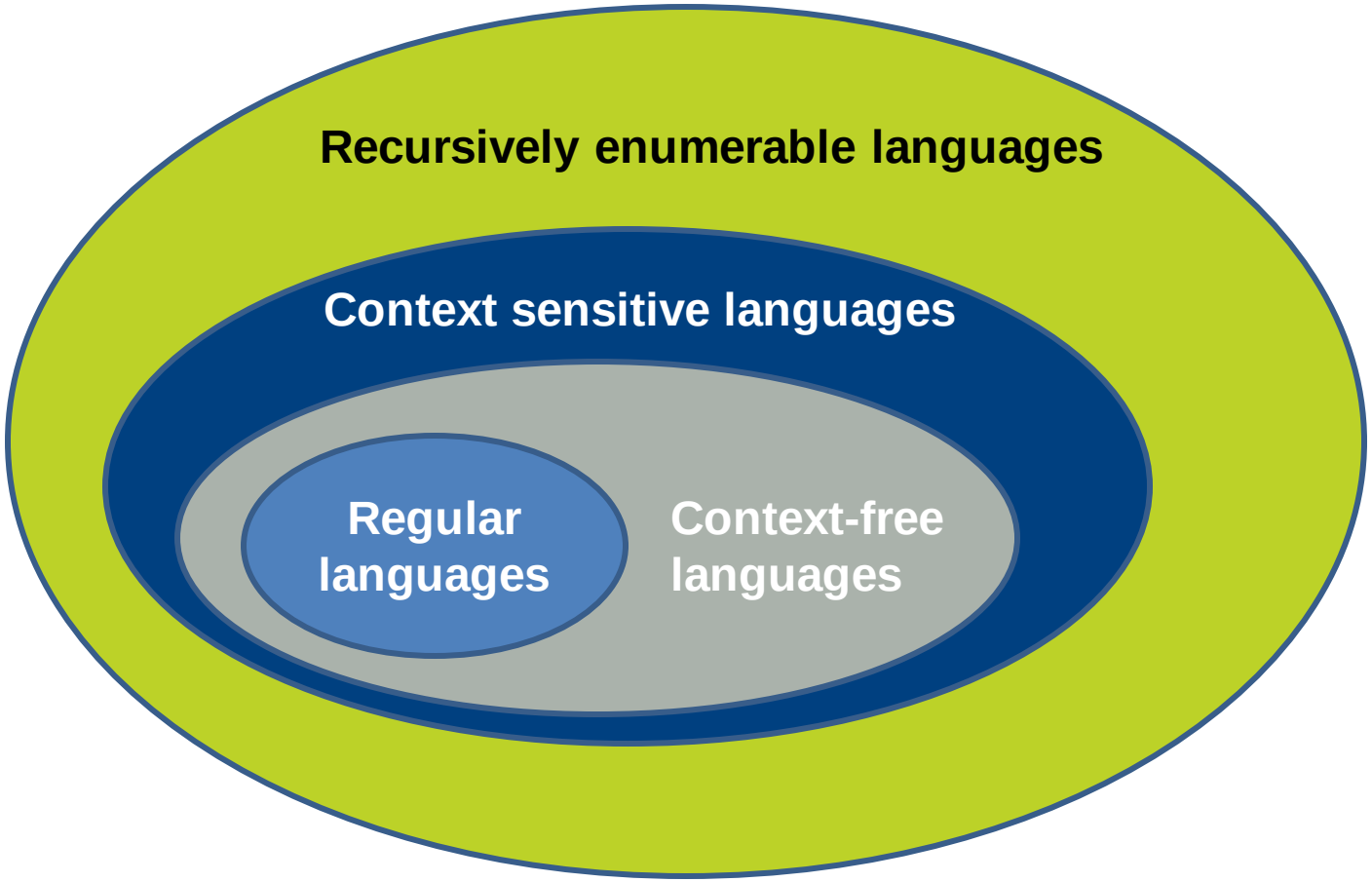
is about learning a grammar given information about a language

- Information is strings, trees or graphs
- Information can be (typically)
  - Text: only positive information
  - Informant: labelled data
  - Actively sought (query learning, teaching)

1.1 The functions/goals

- Languages and grammars from the Chomsky hierarchy
- Probabilistic automata and context-free grammars
- Hidden Markov Models
- Patterns
- Transducers

1.2 The Chomsky hierarchy



1.3 The Chomsky hierarchy revisited

Regular languages

Recognized by DFA, NFA

Generated by regular grammars

Described by regular expressions

Context-free languages

Generated by CF grammars

Recognized by stack automata

Context-sensitive languages

CS grammars

(parsing is not in P)

RE languages (all Turing machines)

Parsing is undecidable

1.4 Other formalisms

- Topological formalisms
  - Semilinear languages
  - Hyperplanes
  - Balls of strings

1.5 Distributions of strings

- A probabilistic automaton defines a distribution over the strings

1.6 Fuzzy automata

- An automaton will say that string w belongs to the language with probability p
- The difference with the probabilistic automata is that
  - The total sum of probabilities may be different than 1 (may even be infinite)
  - The fuzzy automaton cannot be used as a generator of strings

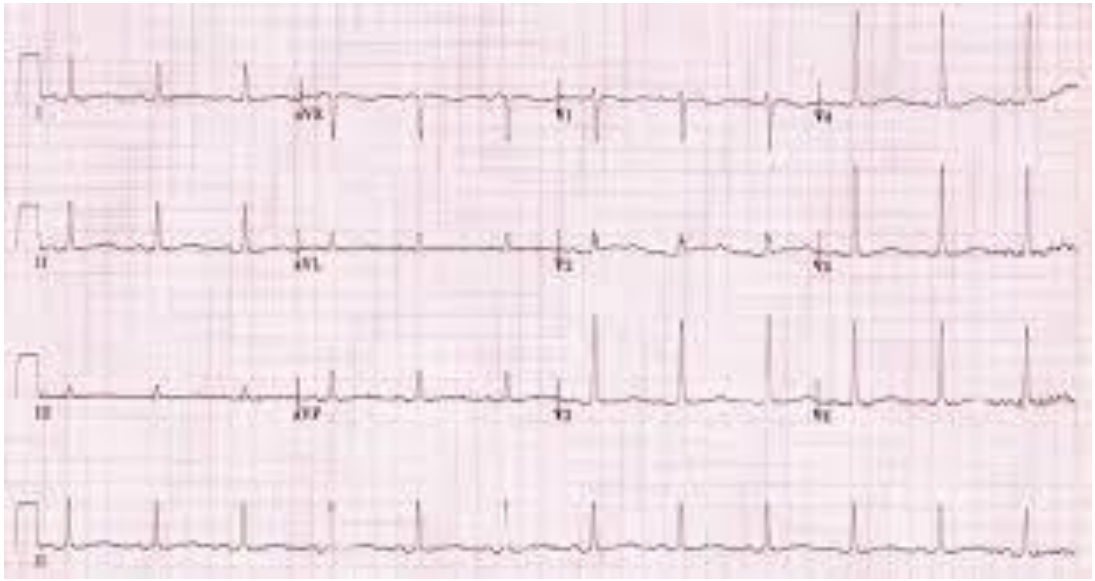
1.7 The data: examples of strings

A string in Gaelic and its translation to English:

- *Tha thu cho duaichnidh ri èarr àirde de a' coisich deas damh*

- *You are as ugly as the north end of a southward traveling ox*

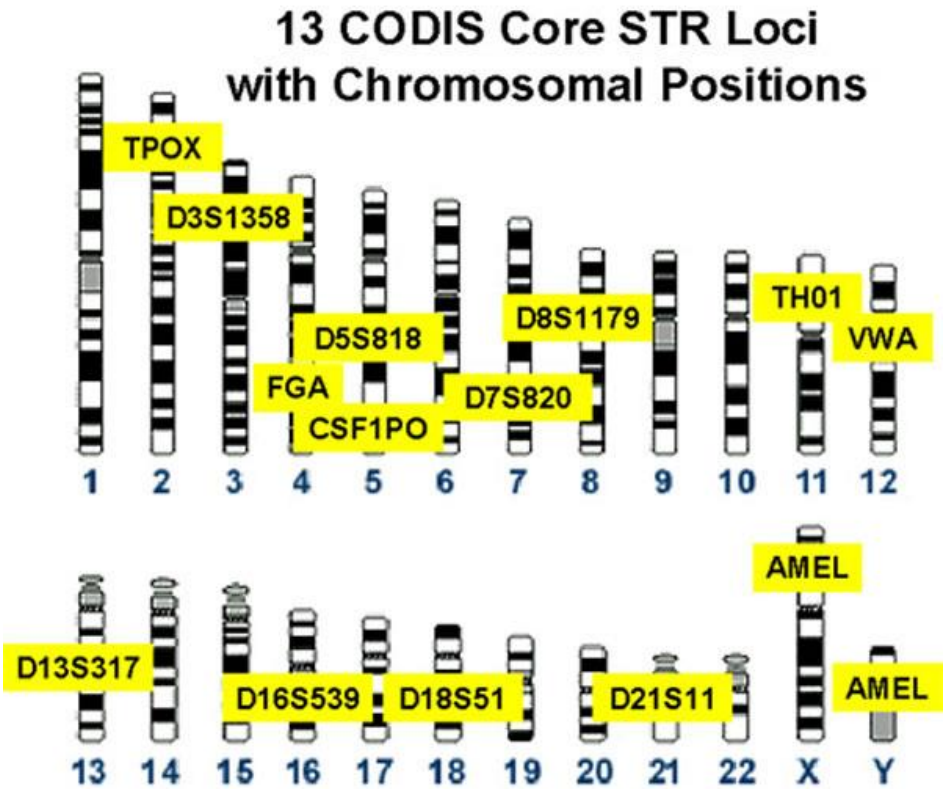
1.7 The data: examples of strings



Sinus rhythm with acquired long QT, work found via Flickr, by Popfossa, CC BY 2.0

- Time series pose the problem of the alphabet:
  - An infinite alphabet?
  - Discretizing?
  - An ordered alphabet

1.7 The data: examples of strings

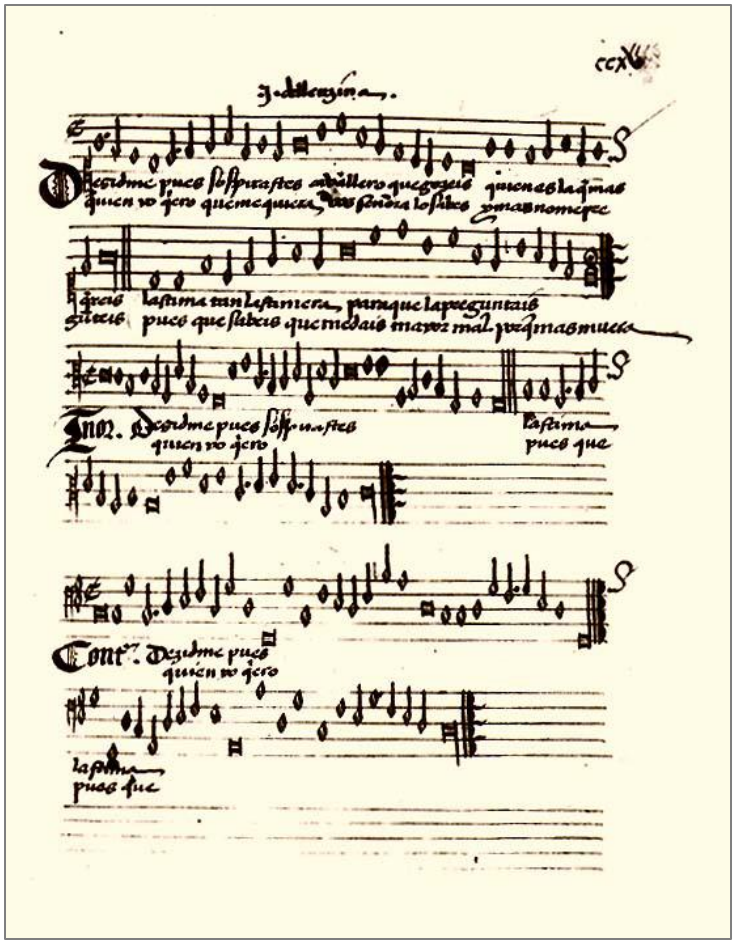


Codis profile, Chemical Science & Technology Laboratory, National Institute of Standards and Technology, work found via Wikipedia, CC BY-SA 3.0

1.7 The data: examples of strings

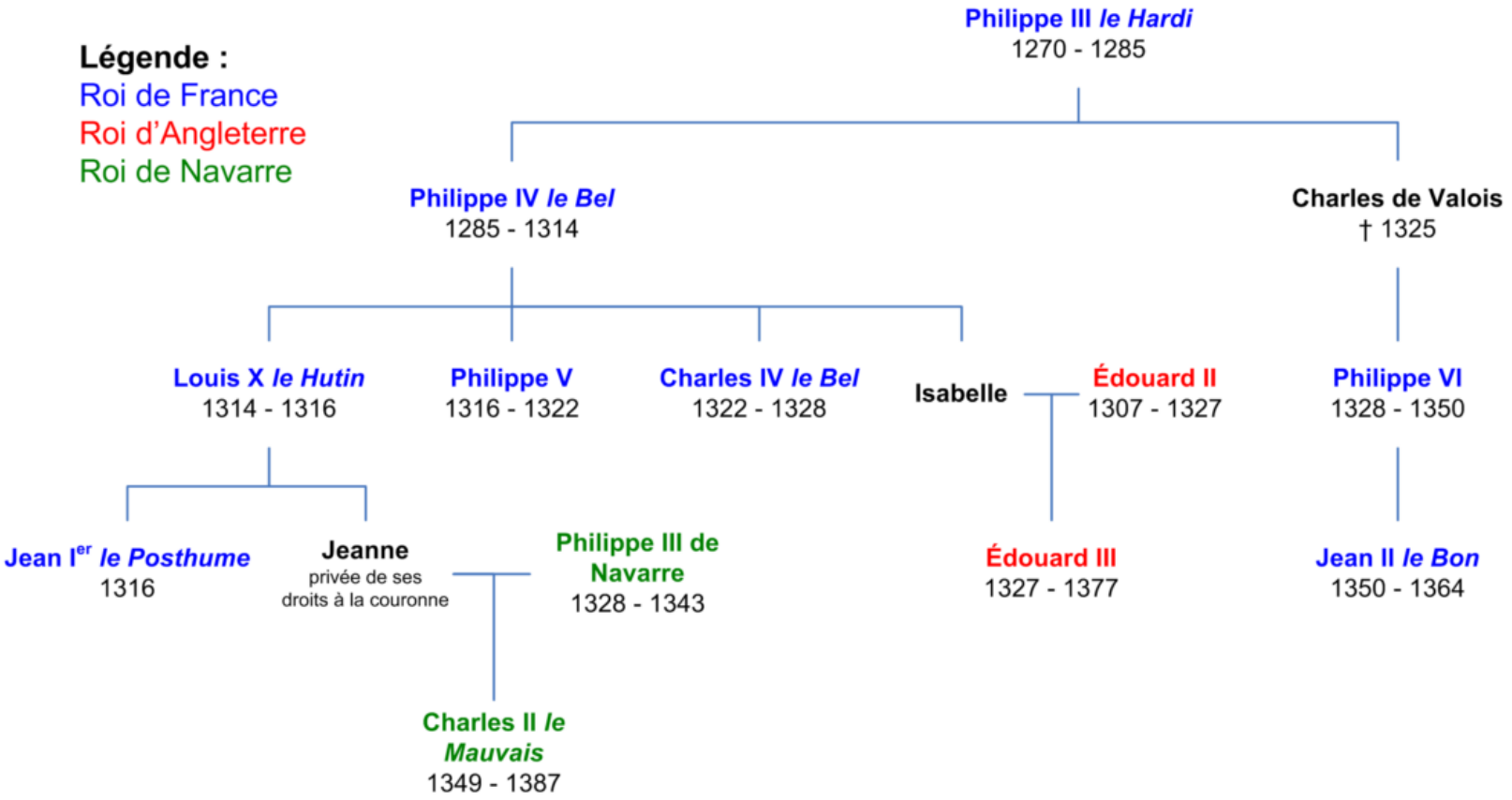
```
>A BAC=41M14 LIBRARY=CITB_978_SKB
AAGCTTATTCAATAGTTTATTAAACAGCTTCTTAAATAGGATATAAGGCAGTGCCATGTA
GTGGATAAAAGTAATAATCATTATAATATTAAGAACTAATACATACTGAACACTTTCAAT
GGCACTTTACATGCACGGTCCCTTTAATCCTGAAAAAATGCTATTGCCATCTTTATTTCA
GAGACCAGGGTGCTAAGGCTTGAGAGTGAAGCCACTTTCCCCAAGCTCACACAGCAAAGA
CACGGGGACACCAGGACTCCATCTACTGCAGGTTGTCTGACTGGGAACCCCCATGCACCT
GGCAGGTGACAGAAATAGGAGGCATGTGCTGGGTTTGGAAGAGACACCTGGTGGGAGAGG
GCCCTGTGGAGCCAGATGGGGCTGAAAACAAATGTTGAATGCAAGAAAAGTCGAGTTCCA
GGGGCATTACATGCAGCAGGATATGCTTTTTAGAAAAAGTCCAAAAACACTAAACTTCAA
CAATATGTTCTTTTGGCTTGCATTTGTGTATAACCGTAATTAAAAAGCAAGGGGACAACA
CACAGTAGATTCAGGATAGGGGTCCCCTCTAGAAAGAAGGAGAAGGGGCAGGAGACAGGA
TGGGGAGGAGCACATAAGTAGATGTAAATTGCTGCTAATTTTTCTAGTCCTTGGTTTGAA
TGATAGGTTCATCAAGGGTCCATTACAAAAACATGTGTTAAGTTTTTTAAAAATATAATA
AAGGAGCCAGGTGTAGTTTGTCTTGAACCACAGTTATGAAAAAAATTCCAACTTTGTGCA
TCCAAGGACCAGATTTTTTTTAAAATAAAGGATAAAAGGAATAAGAAATGAACAGCCAAG
TATTCACTATCAAATTTGAGGAATAATAGCCTGGCCAACATGGTGAAACTCCATCTCTAC
TAAAAATACAAAAATTAGCCAGGTGTGGTGGCTCATGCCTGTAGTCCCAGCTACTTGCGA
GGCTGAGGCAGGCTGAGAATCTCTTGAACCCAGGAAGTAGAGGTTGCAGTAGGCCAAGAT
GGCGCCACTGCACTCCAGCCTGGGTGACAGAGCAAGACCCTATGTCCAAAAAAAAAAAAA
AAAAAAAGGAAAAGAAAAAGAAAGAAAACAGTGTATATATAGTATATAGCTGAAGCTCCC
TGTGTACCCATCCCCAATTCCATTTCCCTTTTTTGTCCCAGAGAACACCCCATTCCTGAC
TAGTGTTTTATGTTCCTTTGCTTCTCTTTTTAAAAACTTCAATGCACACATATGCATCCA
TGAACAACAGATAGTGGTTTTTGCATGACCTGAAACATTAATGAAATTGTATGATTCTAT
```

1.7 The data: examples of strings



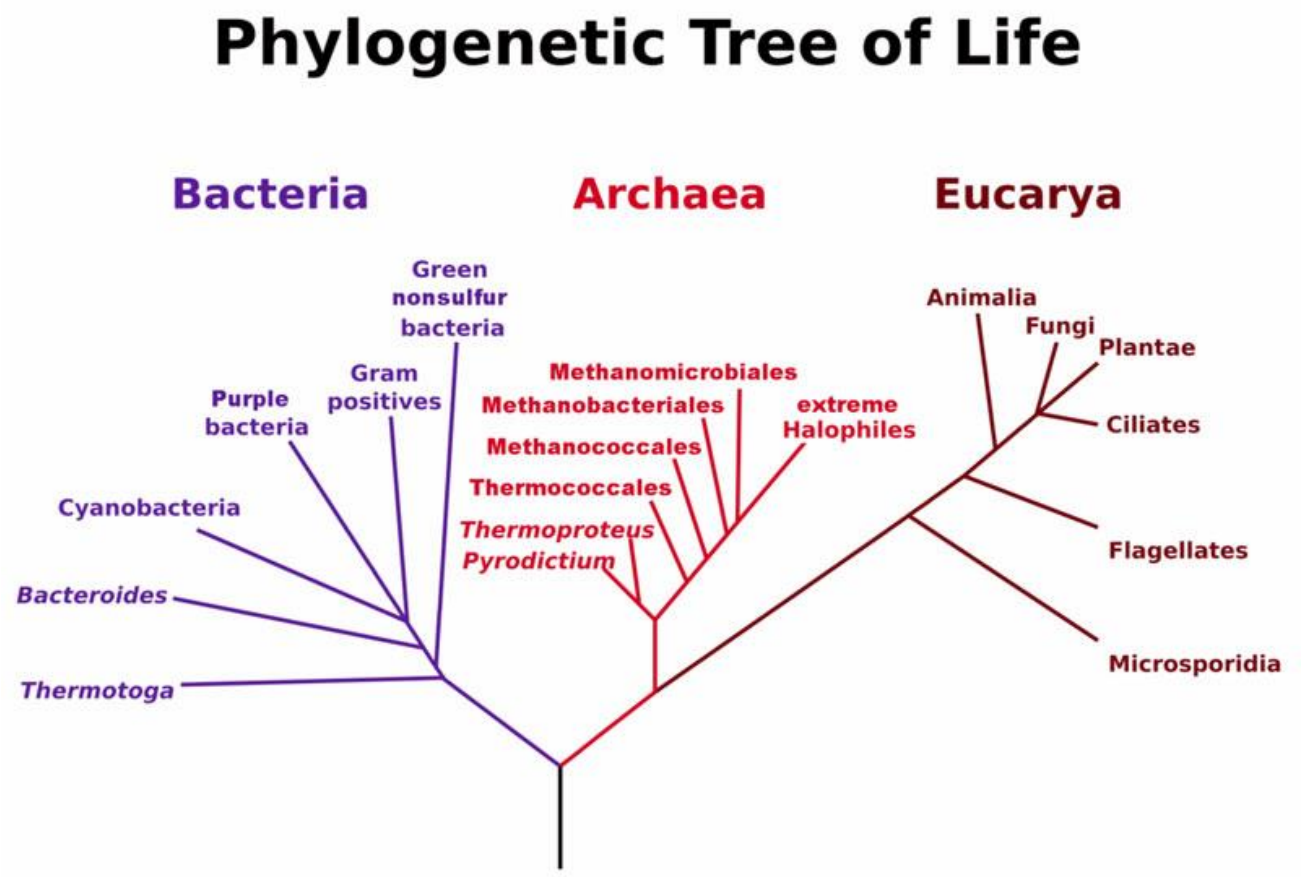
Cancionero de Palacio, work found via Wikipedia, CC BY-SA 3.0

1.7 The data: examples of strings



Généalogie2 Guerre de Cent Ans, work found via Wikipedia, CC BY-SA 3.0

1.7 The data: examples of strings



Phylogenetic Tree, Woese 1990, Maulucioni, work found via Wikipedia, CC BY-SA 3.0

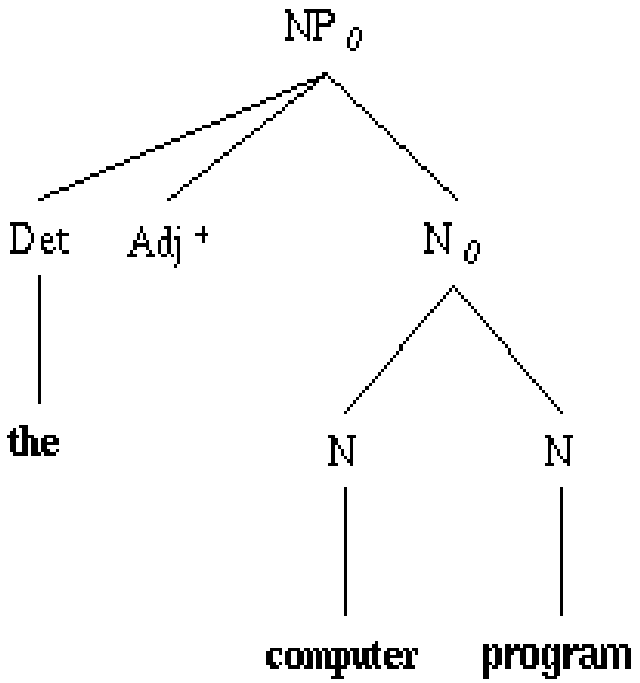
1.7 The data: examples of strings

```
<book>
    <part>
      <chapter>
        <sect1/>
        <sect1>
          <orderedlist numeration="arabic">
            <listitem/>
            <f:fragbody/>
          </orderedlist>
        </sect1>
      </chapter>
    </part>
  </book>
```

1.7 The data: examples of strings

```
<?xml version="1.0"?>
<?xml-stylesheet href="carmen.xsl" type="text/xsl"?>
<?cocoon-process type="xslt"?>
<!DOCTYPE pagina [
<!ELEMENT pagina (titulus?, poema)>
<!ELEMENT titulus (#PCDATA)>
<!ELEMENT auctor (praenomen, cognomen, nomen)>
<!ELEMENT praenomen (#PCDATA)>
<!ELEMENT nomen (#PCDATA)>
<!ELEMENT cognomen (#PCDATA)>
<!ELEMENT poema (versus+)>
<!ELEMENT versus (#PCDATA)>
]>
<pagina>
<titulus>Catullus II</titulus>
<auctor>
<praenomen>Gaius</praenomen>
<nomen>Valerius</nomen>
<cognomen>Catullus</cognomen>
</auctor>
```

1.7 The data: examples of strings




```
[NP {subs 0}
  [Det [{bold the}]]
  [Adj {sups 8 +}]
  [{norm12 N}{subs 0}
     [N [{bold computer}]]
     [N [{sans program}]]]]
```

1.7 And also

- Business processes
- Bird songs
- Images (contours and shapes)
- Robot moves
- Web services
- Malware
- …

2. What does learning mean?

What does learning mean?

- Suppose we write a program that can build a grammar from some data… are we done?
- A first question is: “why bother?”
- If my programme works, why do something more about it?
- Why should we do something when other researchers in Machine Learning are not?

Motivating reflection #1

```
print(17); //
```

```
x=rand(10000;
print(x); //
```

- Is 17 a random number?
- Is 01101101101101010110000111101 a random sequence?

Is grammar *G* the correct grammar for a given sample *S*?

Motivating reflection #2

- In the case of languages, learning is an ongoing process

- Is there a moment where we can say we have learnt a language?

Motivating reflection #3

- Statement “I have learnt” does not make sense
- Statement “I am learning” makes sense

What usually is called “having learnt”

- That the grammar / automaton is the smallest, best (re a score) → Combinatorial characterisation
- That some optimisation problem has been solved
- That the “learning” algorithm has converged (EM)

What is not said

- That having solved some complex combinatorial question we have an Occam, Compression, MDL, Kolmogorov complexity like argument which gives us some guarantee with respect to the future

- Computational learning theory has got such results

Why should we bother and those working in *statistical machine learning* not?

- Whether with numerical functions or with symbolic functions, we are all trying to do some sort of optimisation

- The difference is (perhaps) that numerical optimisation works much better than combinatorial optimisation!

[they actually do bother, only differently]