Grammatical inference: an introduction

Colin de la Higuera
University of Nantes

Nantes


ROYAUME-UNI
BELGIQUE
ALLEMAGNE
SUISSE
ITALIE
ESPAGNE
ANDORRE
MANCHE
OCÉAN ATLANTIQUE
MER MÉDITERRANÉE
MASSIF ARMORICAIN
BASSIN PARISIEN
MASSIF CENTRAL
BASSIN AQUITAIN
Nantes
Paris
Bordeaux
Toulouse
Lyon
Marseille
Carte de France - French map
Voyagevirtuel

Acknowledgements

- Laurent Miclet, Jose Oncina, Tim Oates, Anne-Muriel Arigon, Leo Becerra-Bonache, Rafael Carrasco, Paco Casacuberta, Pierre Dupont, Rémi Eyraud, Philippe Ezequel, Henning Fernau, Jean-Christophe Janodet, Satoshi Kobayachi, Thierry Murgue, Frédéric Tantini, Franck Thollard, Enrique Vidal, Menno van Zaanen,...

http://pagesperso.lina.univ-nantes.fr/~cdlh/

http://videolectures.net/colin_de_la_higuera/

What we are going to talk about

1. Introduction, validation issues
2. Learning automata from an informant
3. Learning automata from text
4. Learning PFA
5. Learning context free grammars
6. Active learning

What we are not going to be talking about

- Transducers
- Setting parameters (EM, Inside outside,…)
- Complex classes of grammars

Outline (of this talk)

1. What is grammatical inference about?
2. A (detailed) introductory example
3. Validation issues
4. Some criteria

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)

Above lists are not limitative

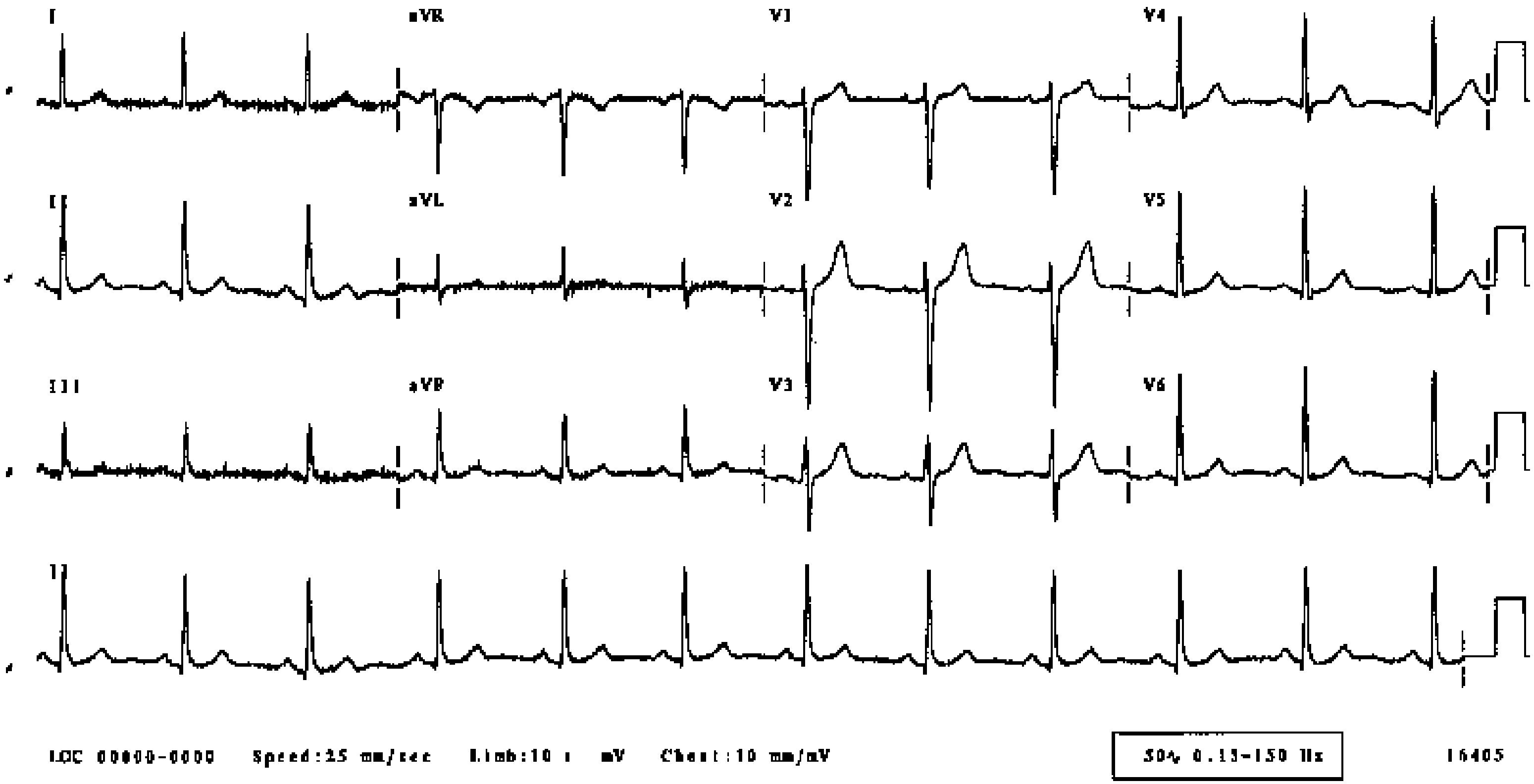
The functions/goals

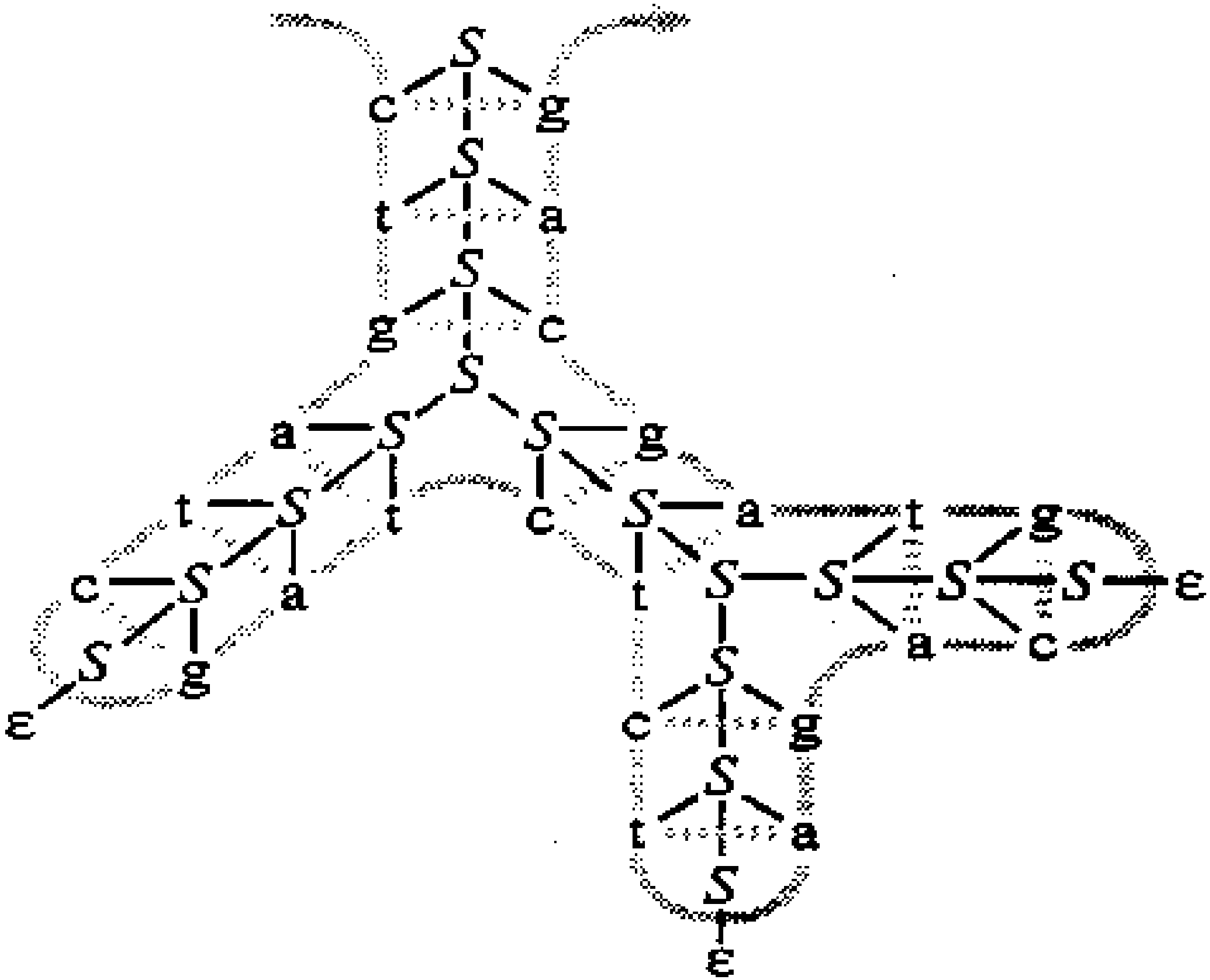
- Languages and grammars from the Chomsky hierarchy
- Probabilistic automata and context-free grammars
- Hidden Markov Models
- Patterns
- Transducers
- …

The data: examples of strings

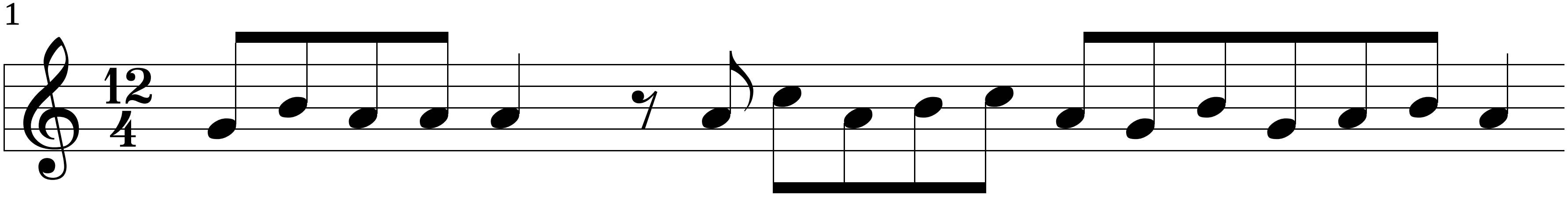
A string in Gaelic and its translation to English:

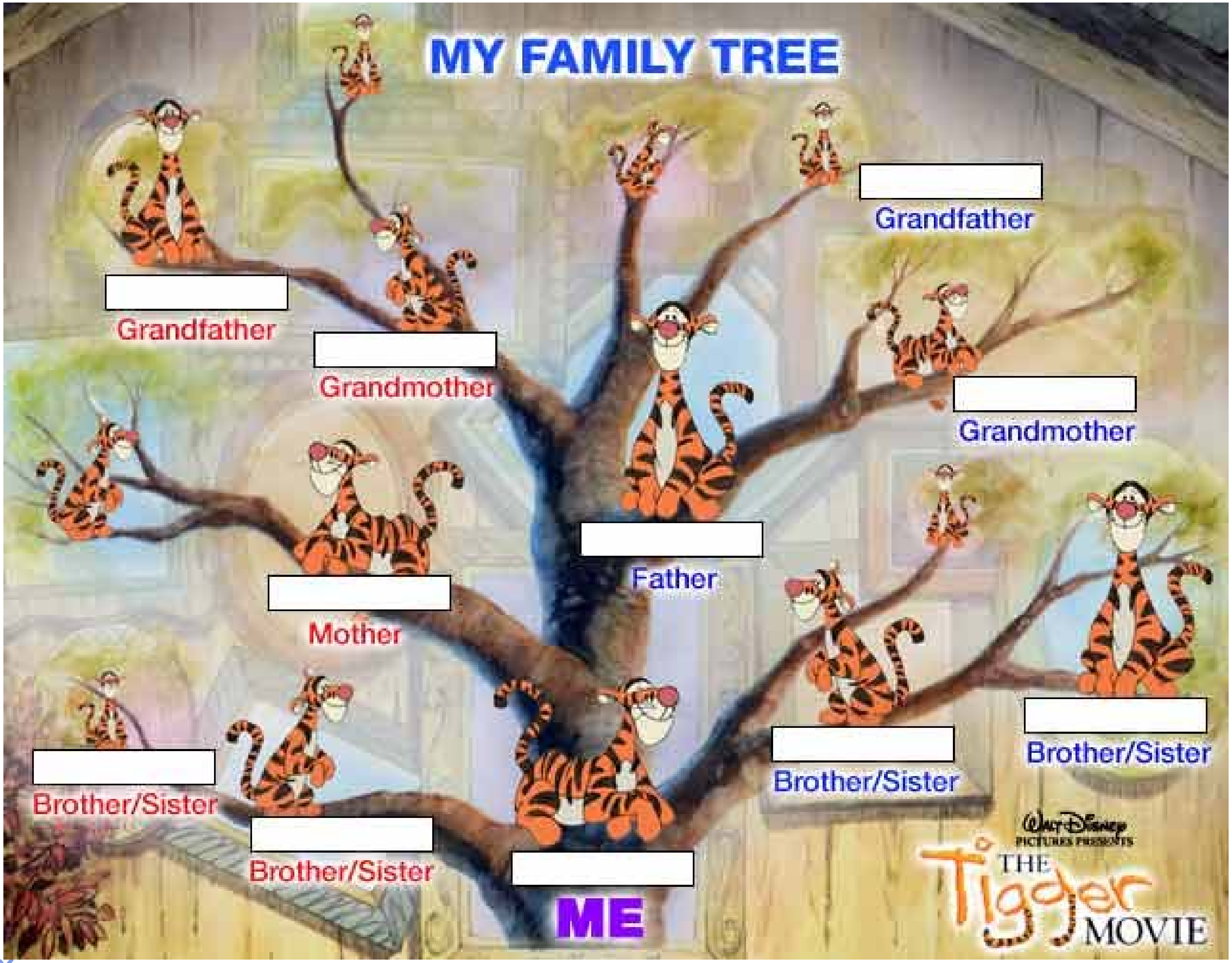
- *Tha thu cho duaichnidh ri èarr àirde de a' coisich deas damh*
- *You are as ugly as the north end of a southward traveling ox*


I
aVR
V1
V4
II
aVL
V2
V5
III
aVF
V3
V6
II
LOC 00000-0000
Speed:25 mm/sec
Limb:10 mm/mV
Chest:10 mm/mV
50~ 0.15-150 Hz
16405

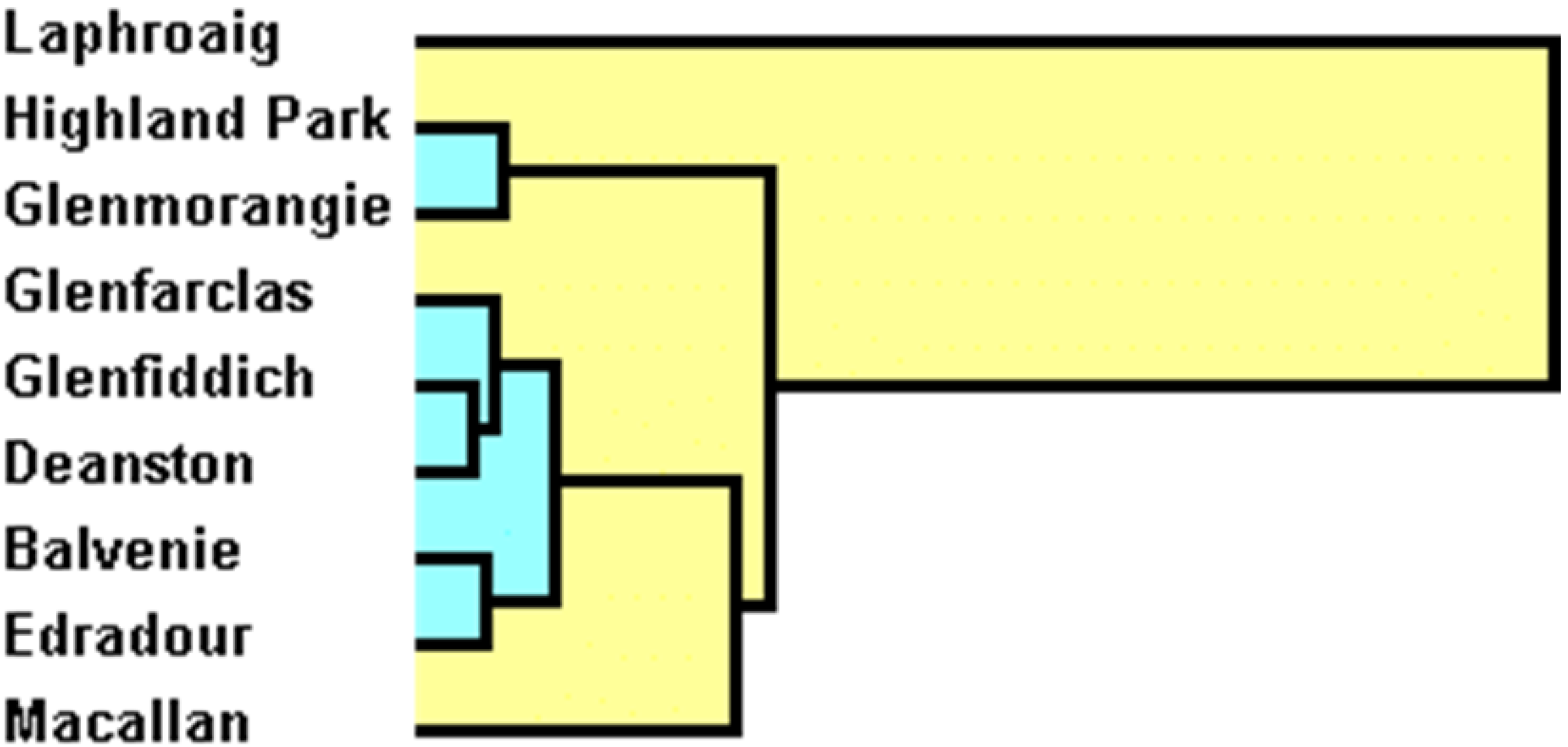


>A BAC=41M14 LIBRARY=CITB_978_SKB
AAGCTTATTCAATAGTTTATTAAACAGCTTCTTAAATAGGATATAAGGCAGTGCCATGTA
GTGGATAAAAGTAATAATCATTATAATATTAAGAACTAATACATACTGAACACTTTCAAT
GGCACTTTACATGCACGGTCCCTTTAATCCTGAAAAAATGCTATTGCCATCTTTATTTCA
GAGACCAGGGTGCTAAGGCTTGAGAGTGAAGCCACTTTCCCCAAGCTCACACAGCAAAGA
CACGGGGACACCAGGACTCCATCTACTGCAGGTTGTCTGACTGGGAACCCCCATGCACCT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


1

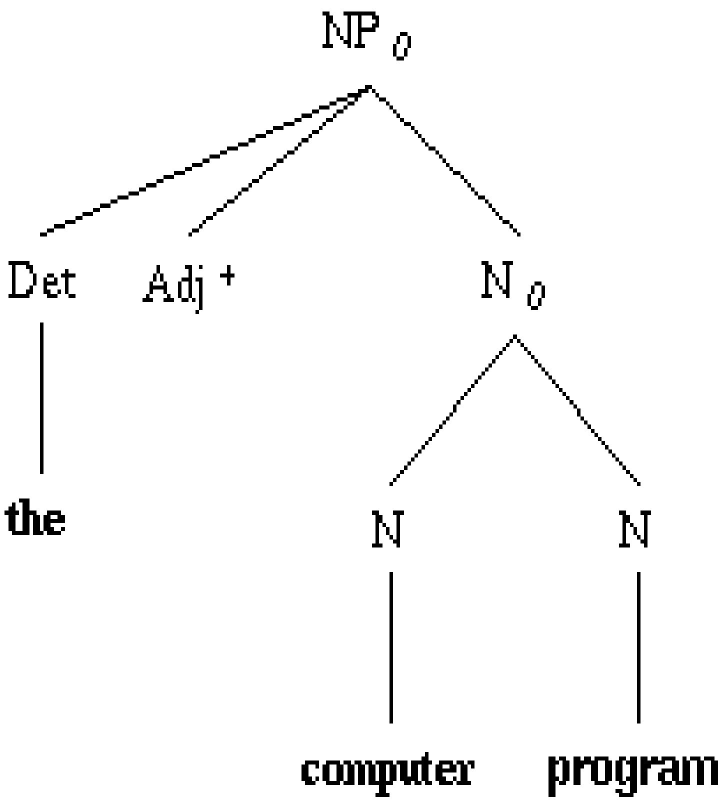

MY FAMILY TREE
Grandfather
Grandmother
Grandfather
Grandmother
Father
Mother
Brother/Sister
Brother/Sister
Brother/Sister
Brother/Sister
ME
Walt Disney
PICTURES PRESENTS
THE
Tigger
MOVIE

Cdlh 2010


Laphroaig
Highland Park
Glenmorangie
Glenfarclas
Glenfiddich
Deanston
Balvenie
Edradour
Macallan

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

```xml
<?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>
```



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

And also

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

2 An introductory example

- D. Carmel and S. Markovitch. Model-based learning of interaction strategies in multi-agent systems. *Journal of Experimental and Theoretical Artificial Intelligence*, 10(3):309–332, 1998
- D. Carmel and S. Markovitch. Exploration strategies for model-based learning in multiagent systems. *Autonomous Agents and Multi-agent Systems*, 2(2):141–172, 1999

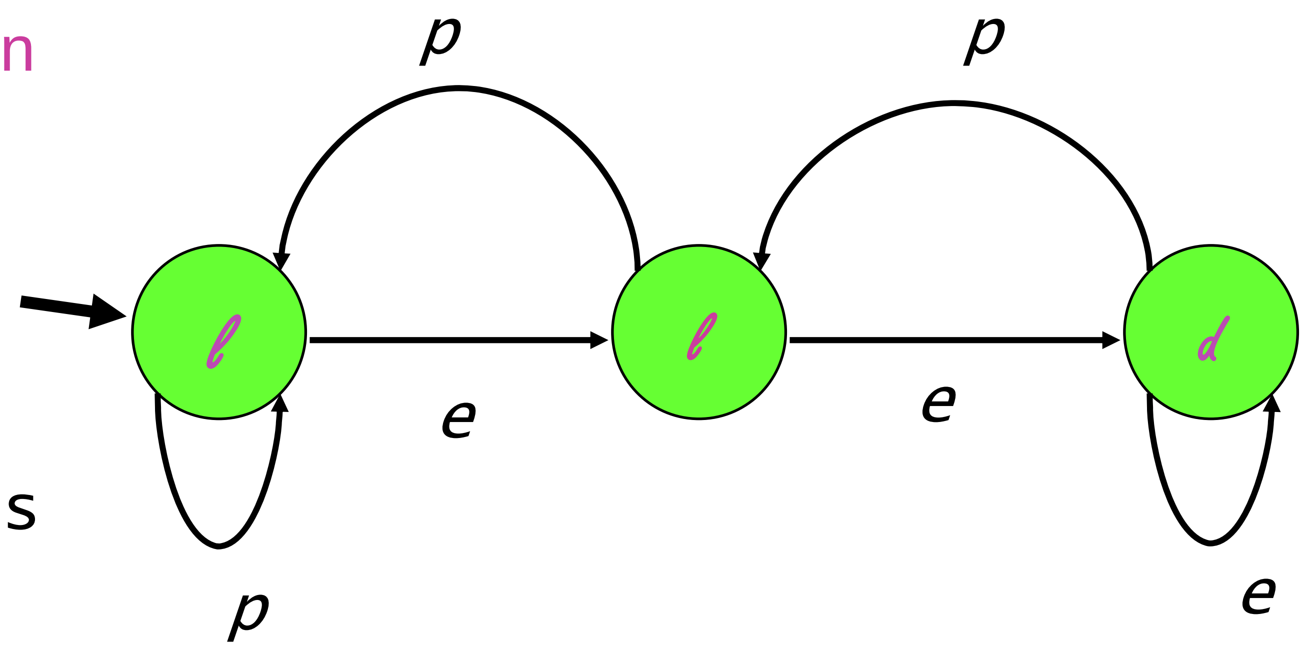
The problem:

- An agent must take cooperative decisions in a multi-agent world
- His decisions will depend:
  - on what he hopes to win or lose
  - on the actions of other agents

Hypothesis: the opponent follows a rational strategy (given by a *DFA*/Moore machine):

You: listen or doze

Me: equations or pictures



e e p e p	→ ℓ
e e e	→ d

Example: (the prisoner's dilemma)

- Each prisoner can admit (*a*) or stay silent (*s*)

- If both admit: 3 years (prison) each
- If A admits but not B: A=0 years, B=5 years
- If B admits but not A: B=0 years, A=5 years
- If neither admits: 1 year each

A \ B	a	s
a	-3, -3	0, -5
s	-5, 0	-1, -1

- Here an iterated version against an opponent who follows a rational strategy
- Gain Function: limit of means (average over a very long series of moves)

The general problem

- We suppose that the strategy of the opponent is given by a deterministic finite automaton
- Can we imagine an optimal strategy?

Suppose we know the opponent's strategy:

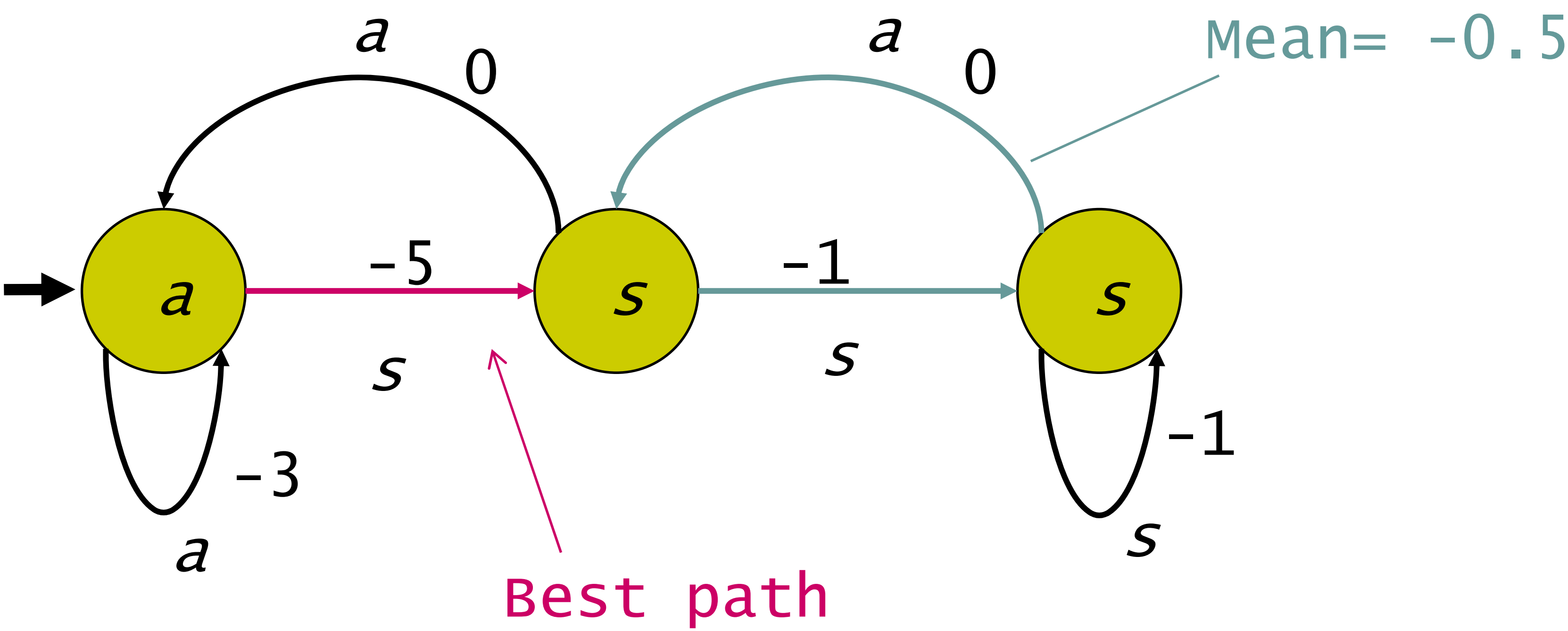
- Then (game theory):
- Consider the opponent's graph in which we value the edges by *our own gain*

1 Find the cycle of maximum mean weight

2 Find the best path leading to this cycle of maximum mean weight

3 Follow the path and stay in the cycle

	a	s
a	-3	0
s	-5	-1



Question

- Can we play a game against this opponent and…

- can we then reconstruct his strategy ?

Data *(him, me)*

HIM	ME
a	*a*
a	*s*
s	*a*
a	*a*
a	*s*
s	*s*
s	*s*
s	*a*
s	*a*

I play *asa*,
his move is *a*

$\lambda \rightarrow a$

$a \rightarrow a$

$as \rightarrow s$

$asa \rightarrow a$

$asaa \rightarrow a$

$asaas \rightarrow s$

$asaass \rightarrow s$

Logic of the algorithm

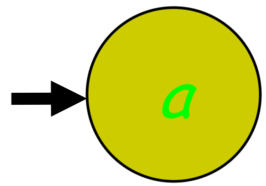
- Goal is to be able to parse ant to have a partial solution consistent with the data
- Algorithm is loosely inspired by a number of grammatical inference algorithms
- It is greedy

First decision:

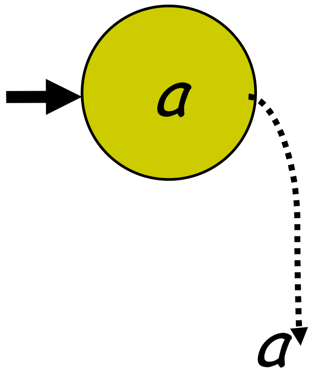
$\lambda \rightarrow a$

$a \rightarrow ?$

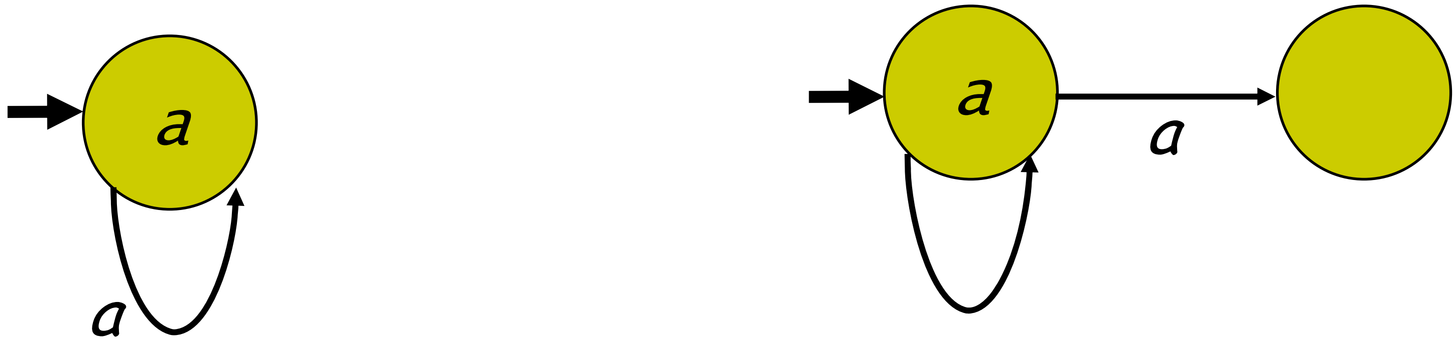
Sure:



Have to deal with:



Candidates



Occam's razor

Entia non sunt multiplicanda praeter necessitatem

"Entities should not be multiplied unnecessarily."

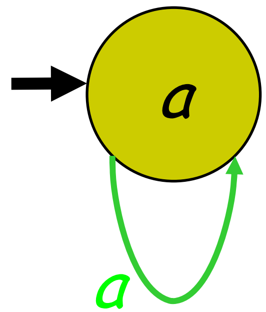
Second decision:

$\lambda \rightarrow a$

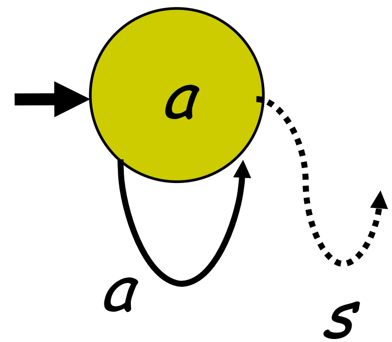
$a \rightarrow a$

$as \rightarrow ?$

Accept:



Have to deal with:



Third decision:

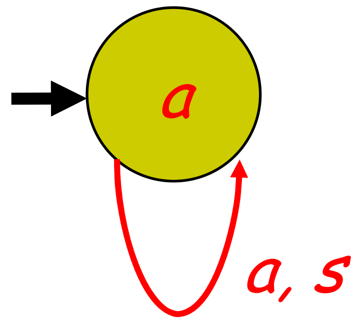
$\lambda \rightarrow a$

$a \rightarrow a$

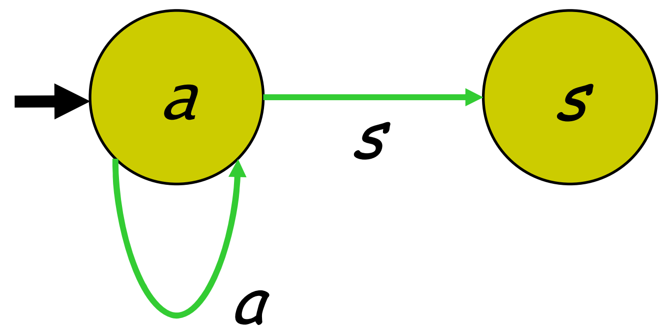
$as \rightarrow s$

$asa \rightarrow ?$

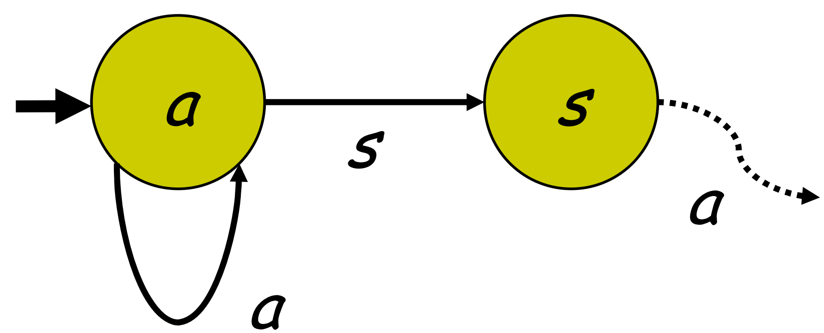
Inconsistent:



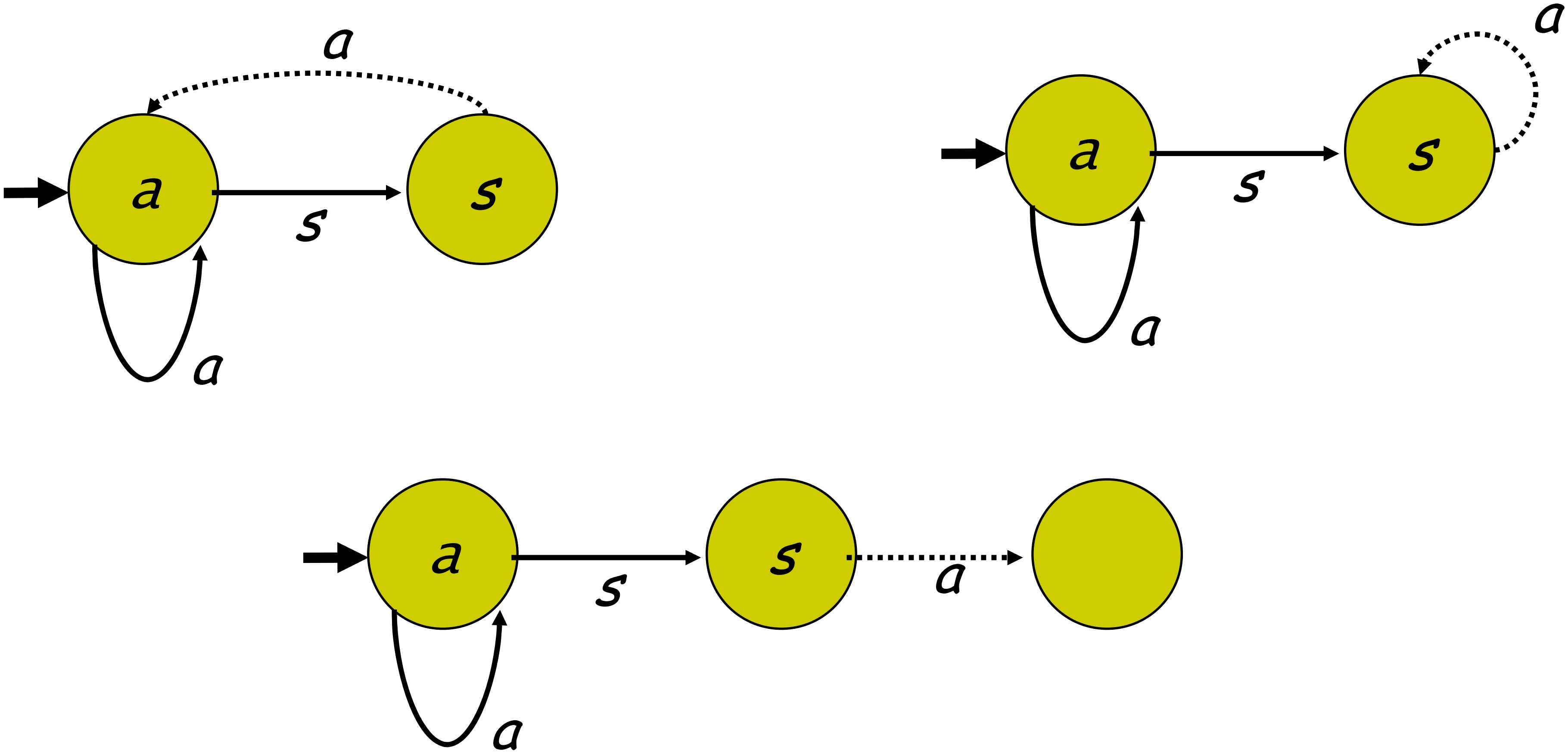
Consistent:



Have to deal with:



Three Candidates



Fourth decision:

$\lambda \rightarrow a$

$a \rightarrow a$

$as \rightarrow s$

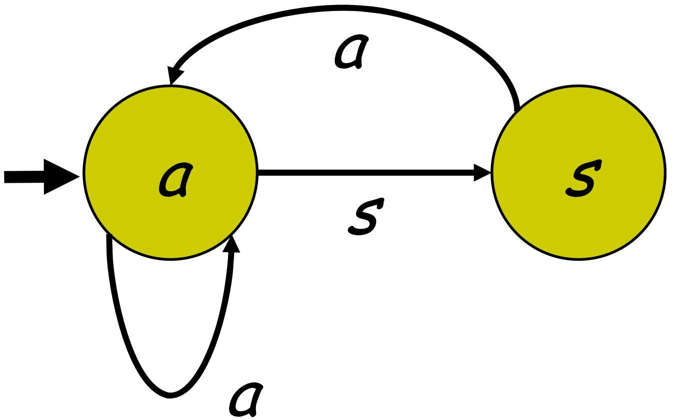
$asa \rightarrow a$

$asaa \rightarrow a$

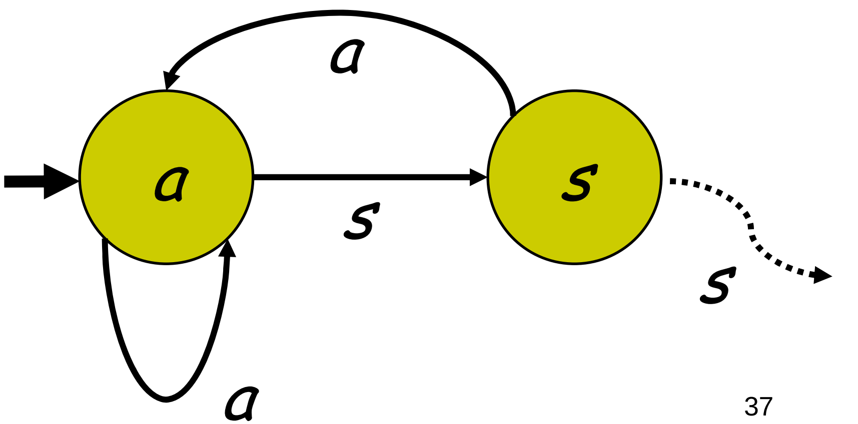
$asaas \rightarrow s$

$asaass \rightarrow ?$

Consistent:



But have to deal with:



Fifth decision:

$\lambda \rightarrow a$

$a \rightarrow a$

$as \rightarrow s$

$asa \rightarrow a$

$asaa \rightarrow a$

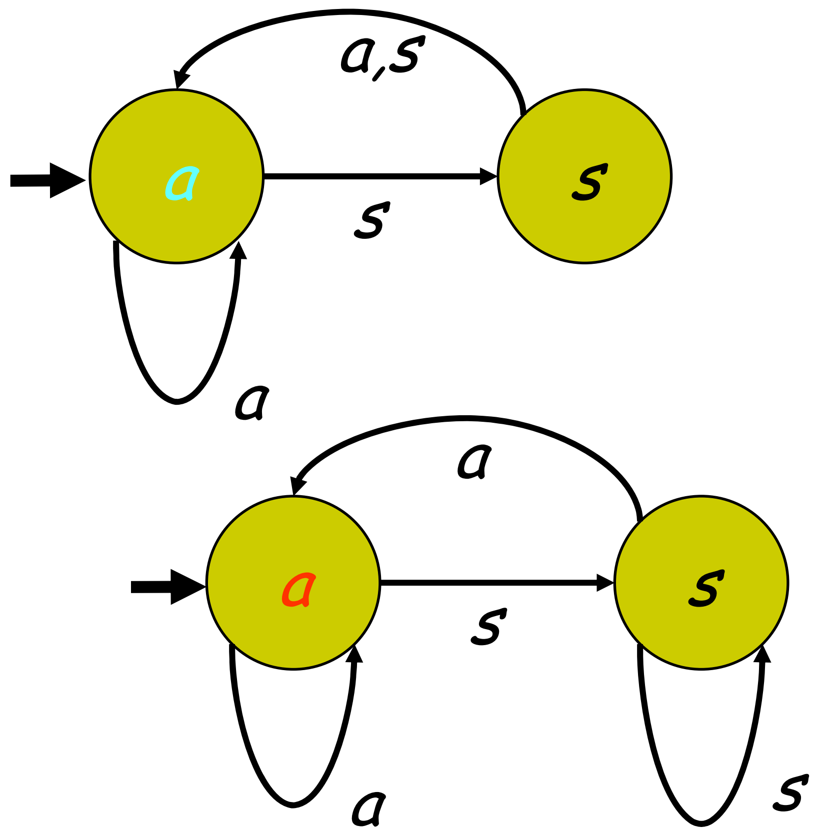
$asaas \rightarrow s$

$asaass \rightarrow s$

$asaasss \rightarrow s$

$asaasssa \rightarrow s$

Inconsistent:



$\lambda \rightarrow a$

$a \rightarrow a$

$as \rightarrow s$

$asa \rightarrow a$

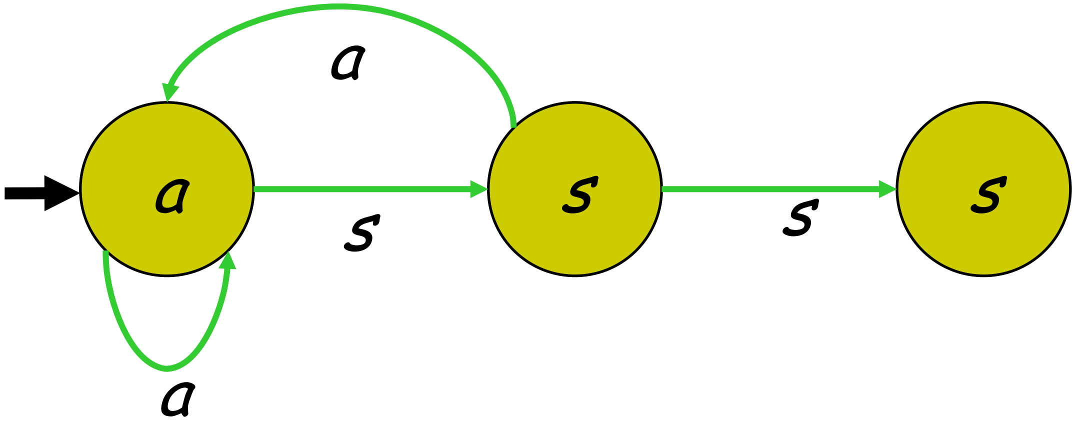
$asaa \rightarrow a$

$asaas \rightarrow s$

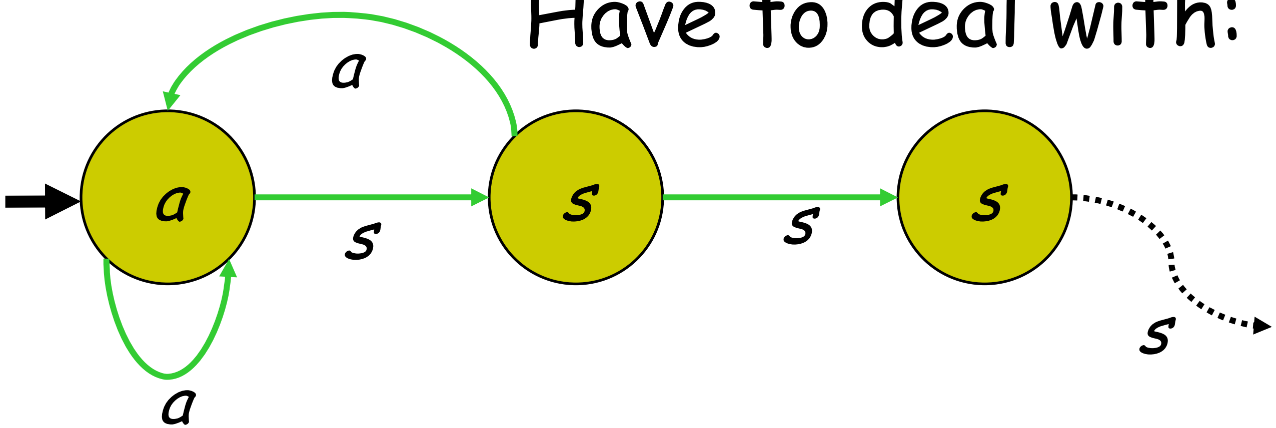
$asaass \rightarrow s$

$asaasss \rightarrow ?$

Consistent:



Have to deal with:



Sixth decision:

$\lambda \rightarrow a$

$a \rightarrow a$

$as \rightarrow s$

$asa \rightarrow a$

$asaa \rightarrow a$

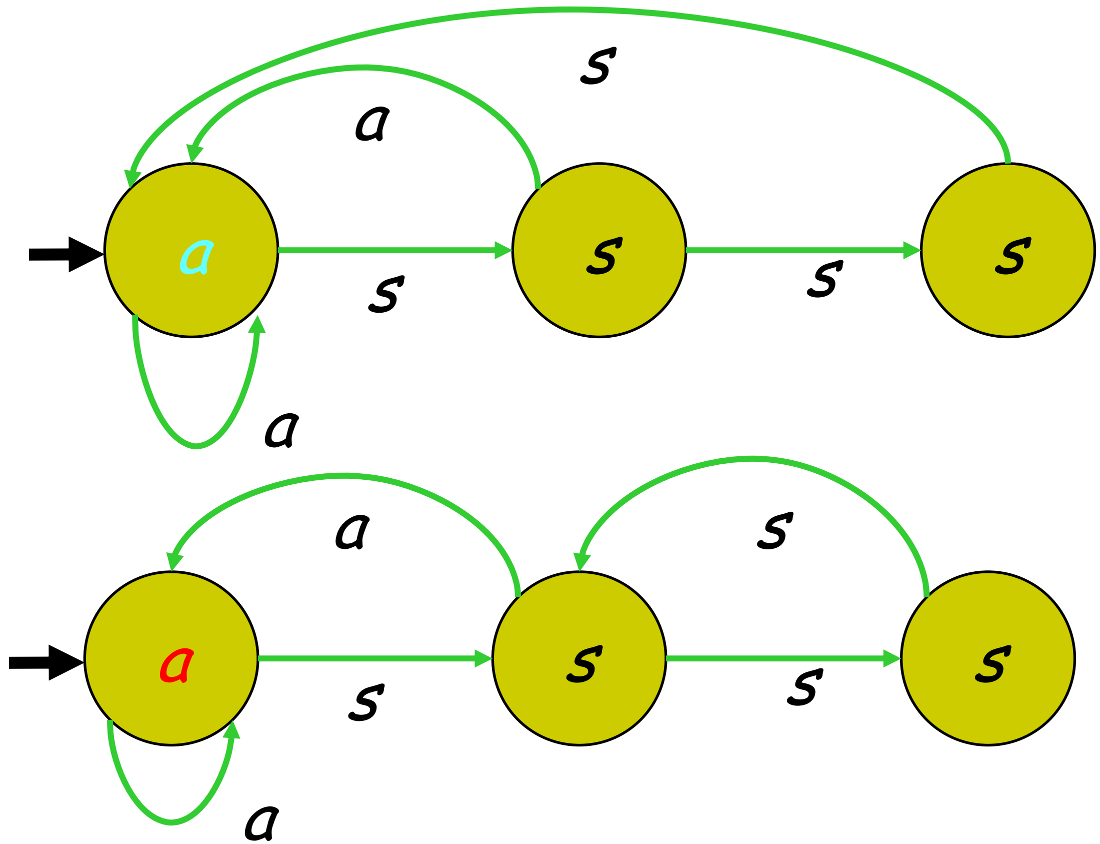
$asaas \rightarrow s$

$asaass \rightarrow s$

$asaasss \rightarrow s$

$asaasssa \rightarrow s$

Inconsistent:



$\lambda \to a$

$a \to a$

$as \to s$

$asa \to a$

$asaa \to a$

$asaas \to s$

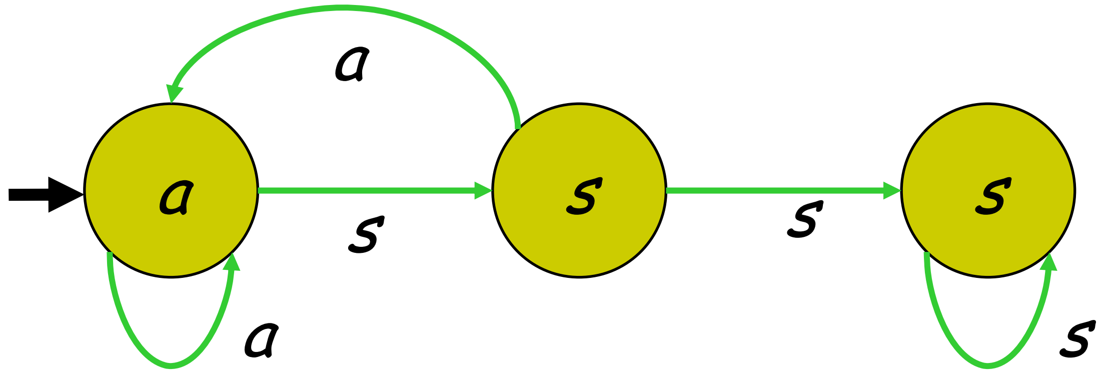
$asaass \to s$

$asaass \to s$

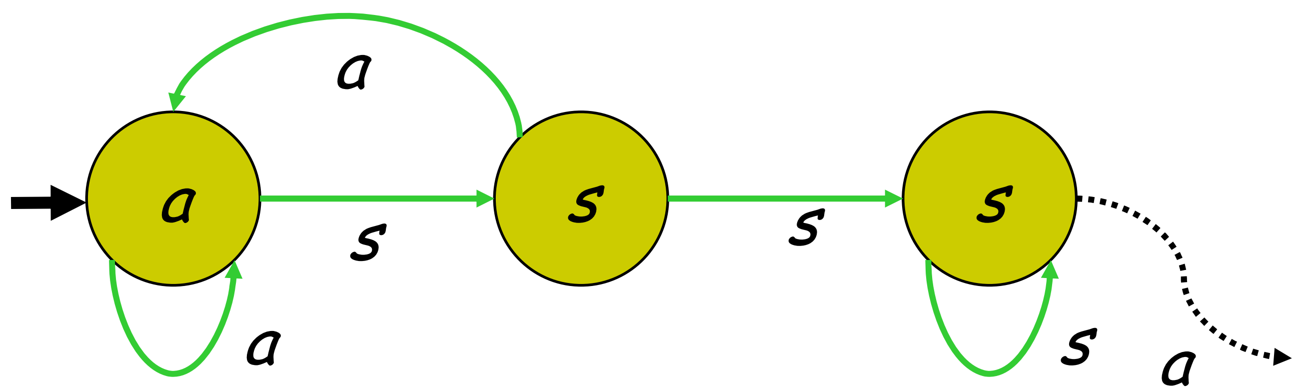
$asaasss \to s$

$asaasssa \to$?

Consistent:



Have to deal with:



Seventh decision:

$asaasss \to s$

$asaasssa \to s$

$\lambda \to a$

$a \to a$

$as \to s$

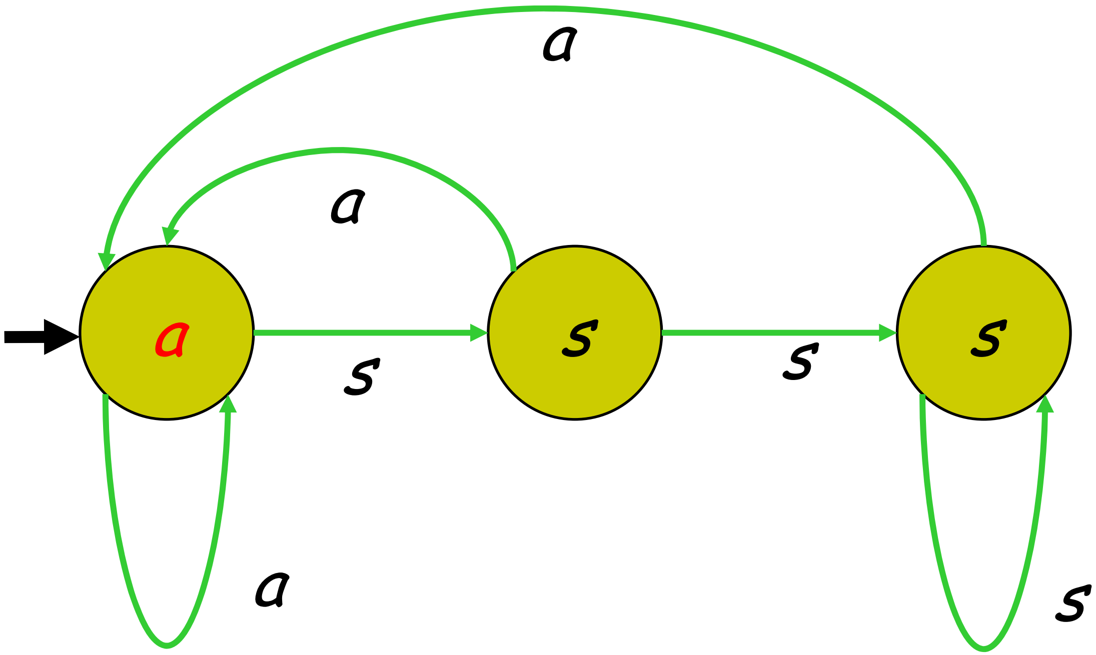
$asa \to a$

$asaa \to a$

$asaas \to s$

$asaass \to s$

Inconsistent:



$\lambda \rightarrow a$

$a \rightarrow a$

$as \rightarrow s$

$asa \rightarrow a$

$asaa \rightarrow a$

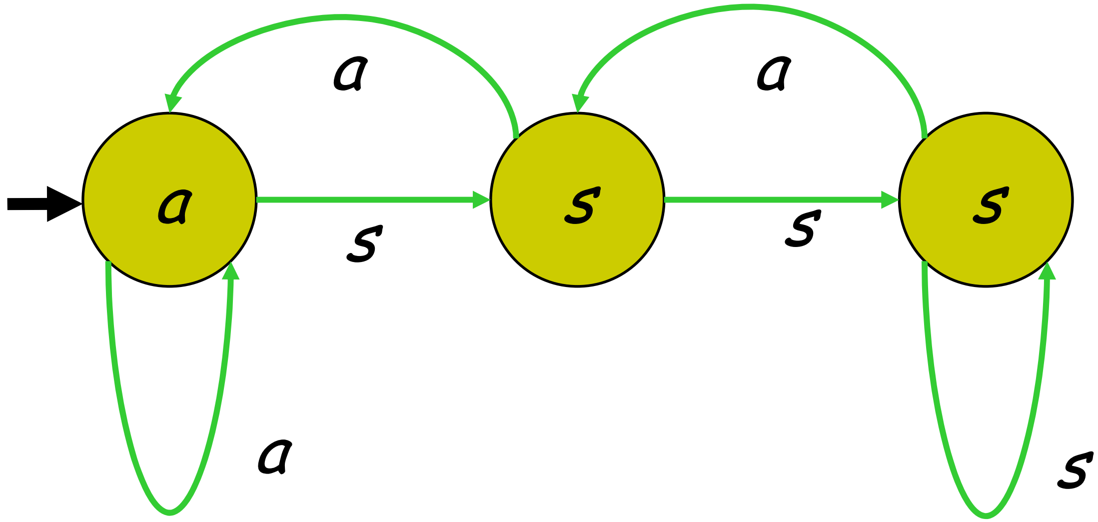
$asaas \rightarrow s$

$asaass \rightarrow s$

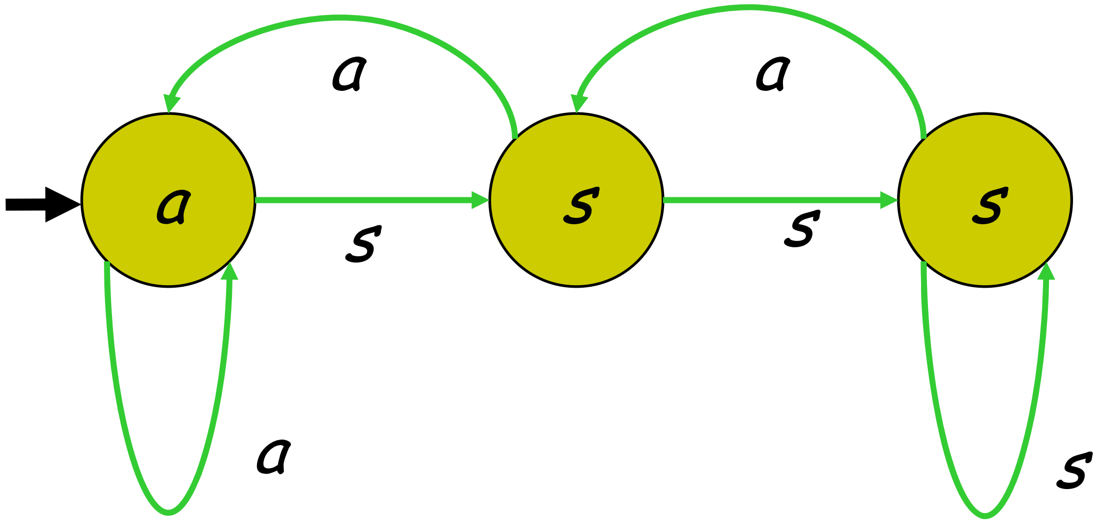
$asaasss \rightarrow s$

$asaasssa \rightarrow s$

Consistent:



Result



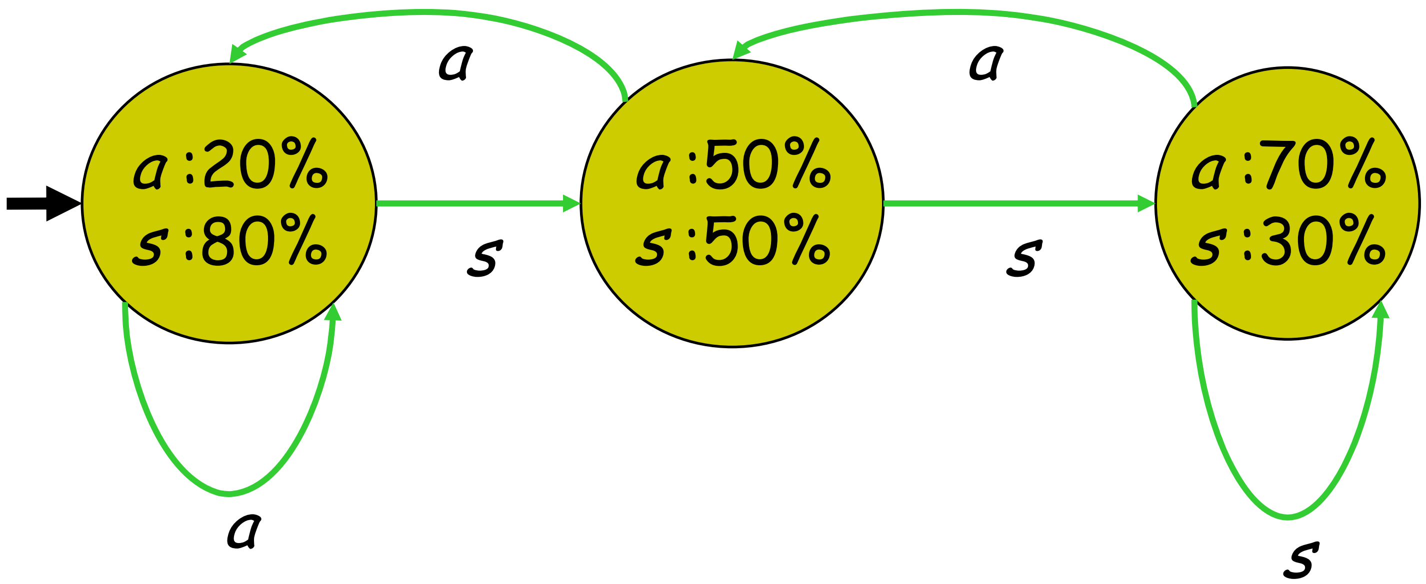
How do we get hold of the learning data?

a) through observation

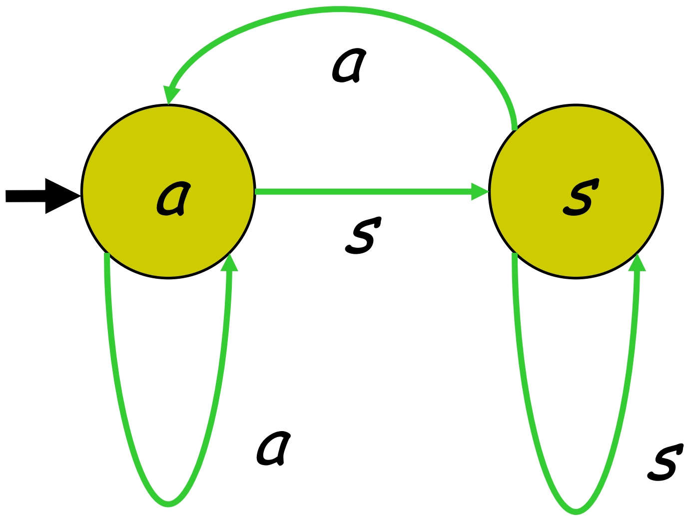
b) through exploration (like here)

An open problem

The strategy is probabilistic:



Tit for Tat



3 What does learning mean?

- Suppose we write a program that can learn FSM… are we done?
- The 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 question #1

- Is 17 a random number?
- Is 0110110110110101011000111101 a random sequence?

(Is FSM *A* the correct FSM for sample *S*?)

Motivating question #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 question #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 we would like to say

- 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]

4 Some convergence criteria

- What would we like to say?
- That in the near future, given some string, we can predict if this string belongs to the language or not
- It would be nice to be able to bet €1000 on this

(if not) What would we like to say?

- That if the solution we have returned is not good, then that is because the initial data was bad (insufficient, biased)
- Idea: blame the data, not the algorithm

Suppose we cannot say anything of the sort?

- Then that means that we may be terribly wrong even in a favourable setting
- Thus there is a hidden bias
- Hidden bias: the learning algorithm is supposed to be able to learn anything inside class $\mathcal{A}$, but can really only learn things inside class $\mathcal{B}$, with $\mathcal{B} \subset \mathcal{A}$

4.1 Non probabilistic setting

- Identification in the limit
- Resource bounded identification in the limit
- Active learning (query learning)

Identification in the limit

- E. M. Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967
- E. M. Gold. Complexity of automaton identification from given data. *Information and Control*, 37:302–320, 1978

The general idea

- Information is presented to the learner who updates its hypothesis after each piece of data
- At some point, always, the learner will have found the correct concept and not change from it

Example

2

{2}

3

{2, 3}

5

Fibonacci numbers

7

11

Prime numbers

103

23

31

A presentation is

a function $\varphi : \mathbb{N} \rightarrow X$

- where X is some set,
- and such that φ is associated to a language L through a function *yields*: $yields(\varphi) = L$
- If $\varphi(\mathbb{N}) = \psi(\mathbb{N})$ then $yields(\varphi) = yields(\psi)$

Some types of presentations (1)

- A *text* presentation of a language $L \subseteq \Sigma^*$ is a function $\varphi : \mathbb{N} \rightarrow \Sigma^*$ such that $\varphi(\mathbb{N})=L$
- φ is an infinite succession of all the elements of L
- (note : small technical difficulty with $\emptyset$)

Some types of presentations (2)

- An *informed* presentation (or an informant) of $L \subseteq \Sigma^*$ is a function $\varphi : \mathbb{N} \rightarrow \Sigma^* \times \{-,+\}$ such that $\varphi(\mathbb{N})=(L,+)\cup(\overline{L},-)$
- φ is an infinite succession of all the elements of Σ^* labelled to indicate if they belong or not to L

Presentation for $\{a^n b^n : n \in \mathbb{N}\}$

- Legal presentation from text: λ, a^2b^2, a^7b^7…
- Illegal presentation from text: ab, ab, ab,…
- Legal presentation from informant : $(\lambda,+)$, $(abab,-)$, $(a^2b^2,+)$, $(a^7b^7\ldots,+)$, $(aab,-)$,…

Naming function ($\mathbb{L}$)

- Given a presentation φ, φ_n is the set of the first n elements in φ
- A learning algorithm $\mathbf{a}$ is a function that takes as input a set φ_n and returns a representation of a language
- Given a grammar G, $\mathbb{L}(G)$ is the language generated/recognised/ represented by G

Convergence to a hypothesis

- Let L be a language from a class $\mathcal{L}$, let φ be a presentation of L and let φ_n be the first n elements in f,
- **a** *converges* to G with φ if
  - $\forall n \in \mathbb{N}$: $\mathbf{a}(\varphi_n)$ halts and gives an answer
  - $\exists n_0 \in \mathbb{N}$: $n \geq n_0 \Rightarrow \mathbf{a}(\varphi_n) = G$

Identification in the limit

A class of languages

$$\begin{tikzcd}
\mathcal{L} & & \mathcal{P}res \subseteq \mathbb{N} \to X \arrow[ll, dashed, "yields"'] \arrow[dl, "\mathbf{a}"] \\
& \mathcal{G} \arrow[ul, "\mathbf{L}"] &
\end{tikzcd}$$

A learner

The naming function

A class of grammars

$$\mathbf{L}(\mathbf{a}(\varphi)) = yields(\varphi)$$

$$\varphi(\mathbb{N}) = \psi(\mathbb{N}) \Rightarrow yields(\varphi) = yields(\psi)$$

Consistency and conservatism

- We say that the learning function **a** is *consistent* if φ_n is consistent with **a**(φ_n) $\forall n$
- A consistent learner is always consistent with the past
- We say that the learning function **a** is *conservative* if whenever $\varphi(n+1)$ is consistent with **a**(φ_n), we have **a**(φ_n)= **a**(φ_{n+1})
- A conservative learner doesn't change his mind needlessly

What about efficiency?

- We can try to bound
  - global time
  - update time
  - errors before converging (IPE)
  - mind changes (MC)
  - queries
  - good examples needed

Resource bounded identification in the limit

- Definitions of IPE, CS, MC, update time, etc...
- What should we try to measure?
  - The size of G?
  - The size of L?
  - The size of f?
  - The size of φ_n?

About the learner

We are addressing here the question of polynomial identification in the limit. So we will not recall every time that the learning algorithm **a** ('*the learner*') does **identify in the limit**!

The size of *G* : ‖*G*‖

- The size of a grammar is the number of bits needed to encode the grammar
- Better some value polynomial in the desired quantity
- Example:
  - DFA : # of states
  - CFG : # of rules * length of rules
  - …

The size of *L*

- If no grammar system is given, meaningless
- If $\mathcal{G}$ is the class of grammars then $\|L\| = \min\{\|G\| : G \in \mathcal{G} \wedge \mathbb{L}(G)=L\}$
- Example: the size of a regular language when considering DFA is the number of states of the minimal DFA that recognizes it

Is a grammar representation reasonable?

- Difficult question: typical arguments are that NFA are better than DFA because you can encode more languages with less bits
- Yet redundancy is necessary!

Proposal

- A grammar class is reasonable if it encodes sufficient different languages
- *Ie* with *n* bits you have 2^{n+1} encodings so optimally you should have 2^{n+1} different languages

But

- We should allow for redundancy and for some strings that do not encode grammars
- Therefore a grammar representation is reasonable if there exists a polynomial $p()$ and for any n the number of different languages encoded by grammars of size n is at least $p(2^n)$

4.2 Probabilistic settings

- PAC learning
- Identification with probability 1
- PAC learning distributions

Learning a language from sampling

- We have a distribution over Σ^*
- We sample twice:
  - once to learn
  - once to see how well we have learned
- The PAC setting

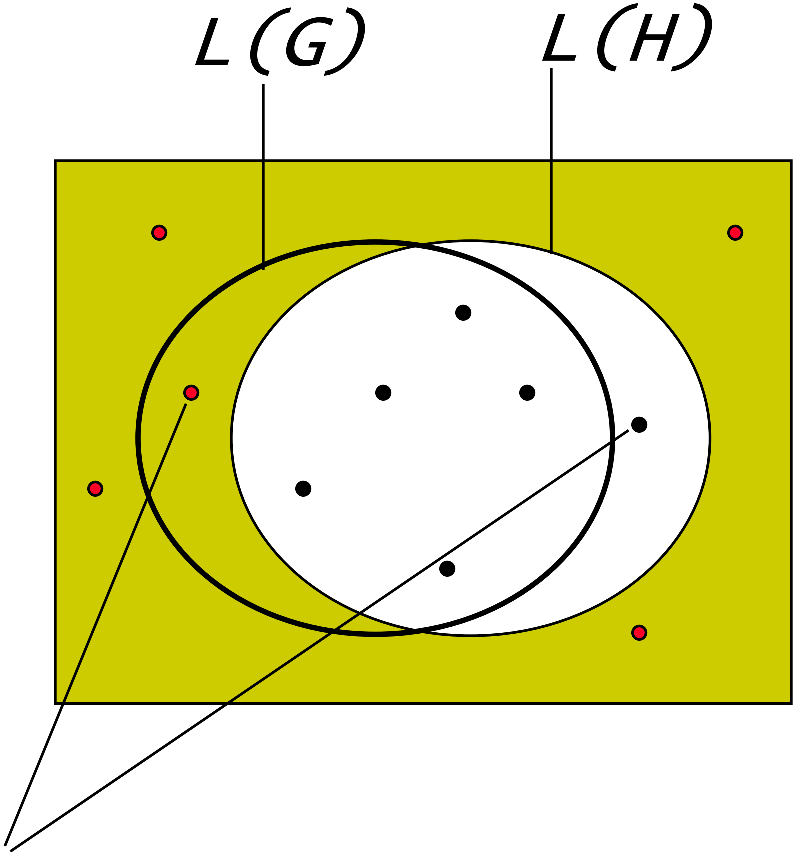
PAC-learning
(Valiant 84, Pitt 89)

- $\mathcal{L}$ a class of languages
- $\mathcal{G}$ a class of grammars
- $\varepsilon > 0$ and $\delta > 0$
- *m* a maximal length over the strings
- *n* a maximal size of machines

$\mathcal{H}$ is ε -*AC* (approximately correct)*

if

$$\Pr_D[\mathcal{H}(x) \neq \mathcal{G}(x)] < \varepsilon$$



Errors: we want this < ε

(French radio)

- Unless there is a surprise there should be no surprise
- (after the last primary elections, on 3rd of June 2008)

Results

- Using cryptographic assumptions, we cannot PAC-learn DFA
- Cannot PAC-learn NFA, CFGs with membership queries either

Alternatively

- Instead of learning classifiers in a probabilistic world, learn directly the distributions!
- Learn probabilistic finite automata (deterministic or not)

No error

- This calls for identification in the limit with probability 1
- Means that the probability of not converging is 0

Results

- If probabilities are computable, we can learn with probability 1 finite state automata
- But not with bounded (polynomial) resources
- Or it becomes very tricky (with added information)

With error

- PAC definition
- But error should be measured by a distance between the target distribution and the hypothesis
- L_1, L_2, L_∞ ?

Results

- Too easy with L_∞
- Too hard with L_1
- Nice algorithms for biased classes of distributions

Conclusion

- A number of paradigms to study identification of learning algorithms
- Some to learn classifiers
- Some to learn distributions