LABORATOIRE D'INFORMATIQUE
DE NANTES ATLANTIQUE

UMR CNRS 6241
Université de Nantes
Ecole des Mines de Nantes

Learning probabilistic finite automata

Colin de la Higuera
University of Nantes

Pattern Analysis, Statistical Modelling and Computational Learning

dépasser les frontières

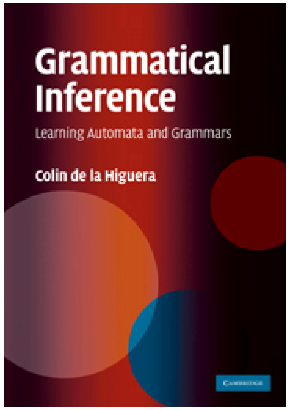
Région PAYS DE LA LOIRE

Acknowledgements

- Laurent Miclet, Jose Oncina, Tim Oates, Rafael Carrasco, Paco Casacuberta, Rémi Eyraud, Philippe Ezequel, Henning Fernau, Thierry Murgue, Franck Thollard, Enrique Vidal, Frédéric Tantini,...
- List is necessarily incomplete. Excuses to those that have been forgotten

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

Chapters 5 and 16



Outline

1. PFA
2. Distances between distributions
3. FFA
4. Basic elements for learning PFA
5. ALERGIA
6. MDI and DSAI
7. Open questions

1 PFA

Probabilistic finite (state) automata

Practical motivations

(Computational biology, speech recognition, web services, automatic translation, image processing …)

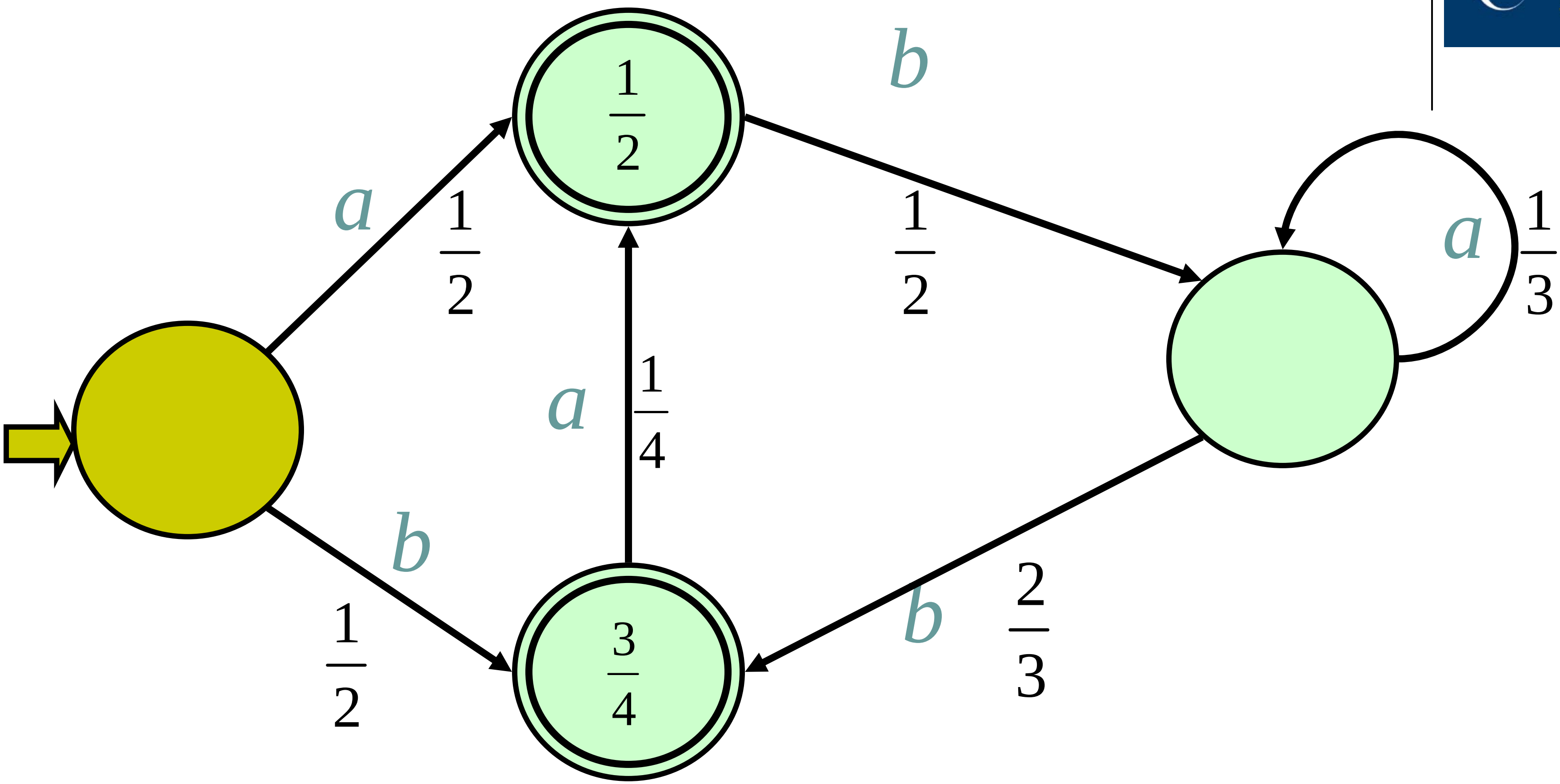
- A lot of positive data
- Not necessarily any negative data
- No ideal target
- Noise

The grammar induction problem, revisited

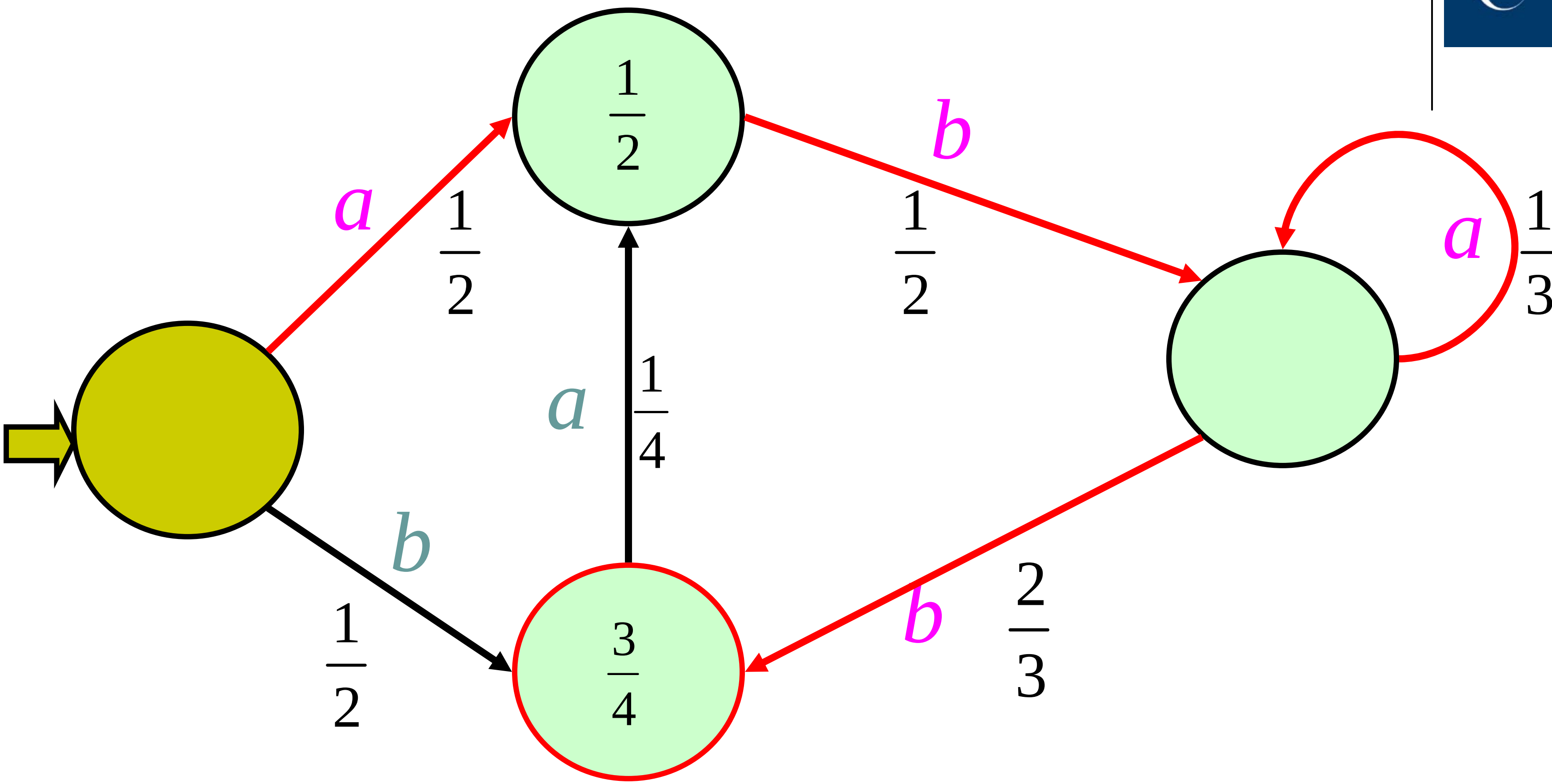
- The data consists of positive strings, «generated» following an unknown distribution
- The goal is now to find (learn) this distribution
- Or the grammar/automaton that is used to generate the strings

Success of the probabilistic models

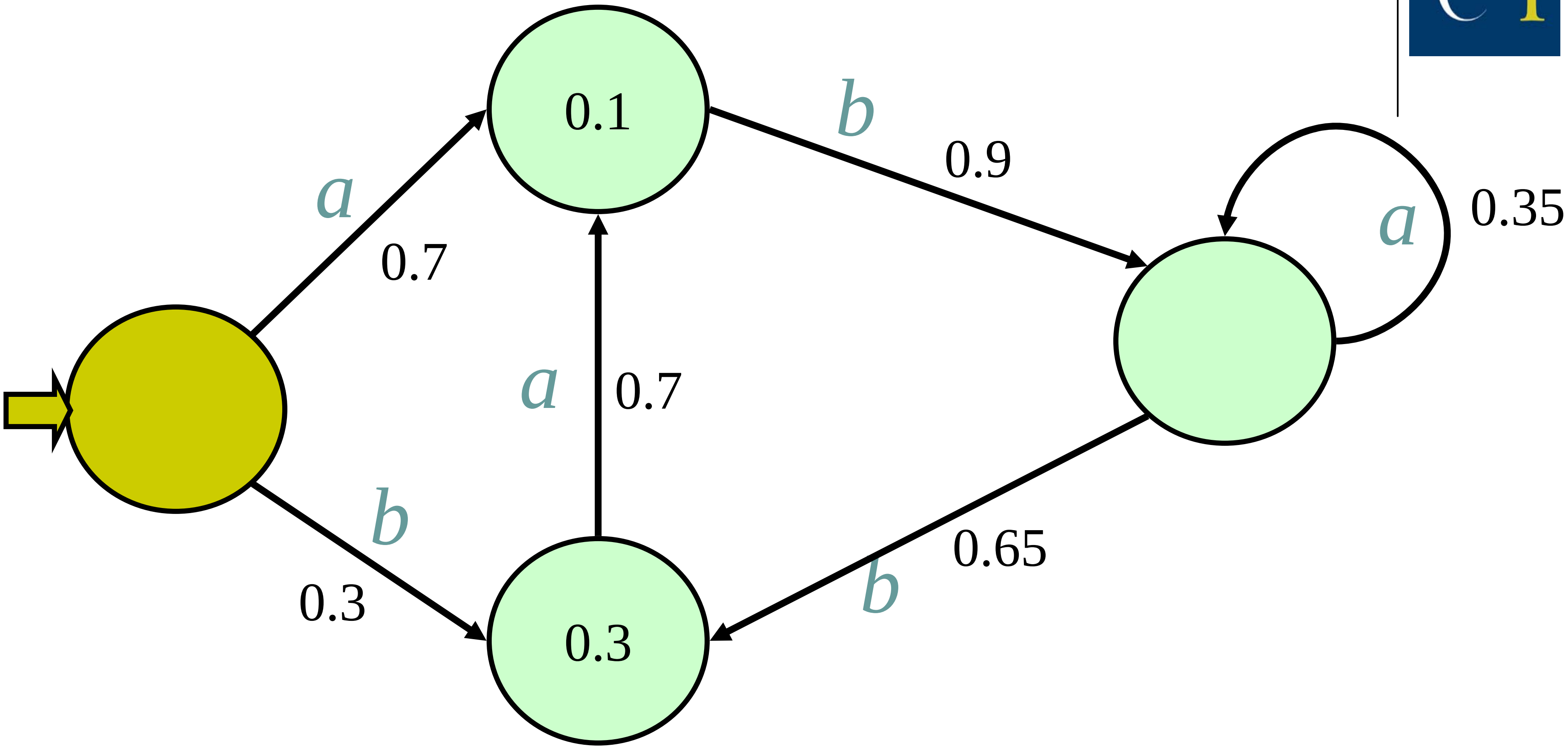
- *n*-grams
- Hidden Markov Models
- Probabilistic grammars

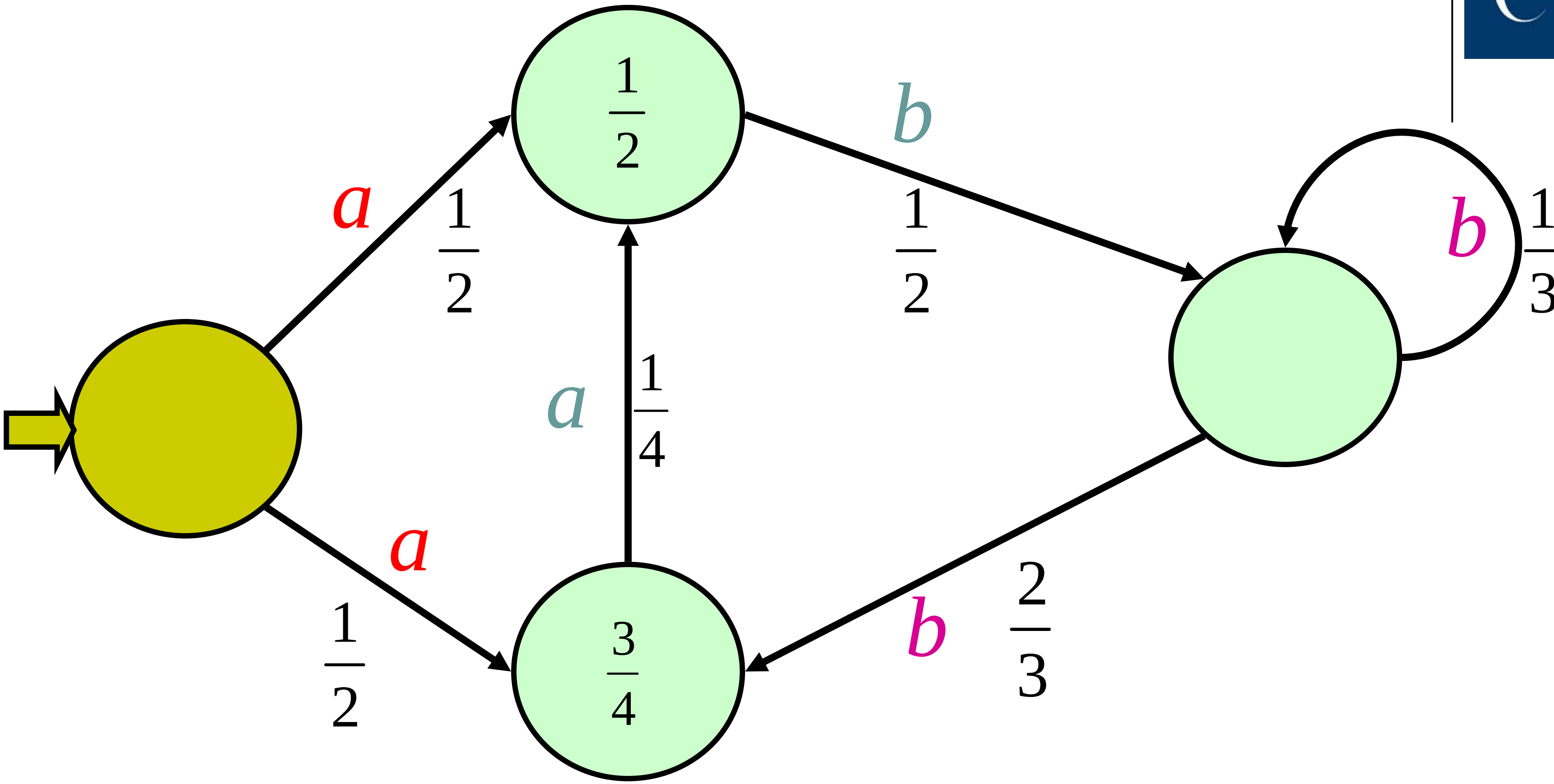


DPFA: Deterministic Probabilistic Finite Automaton

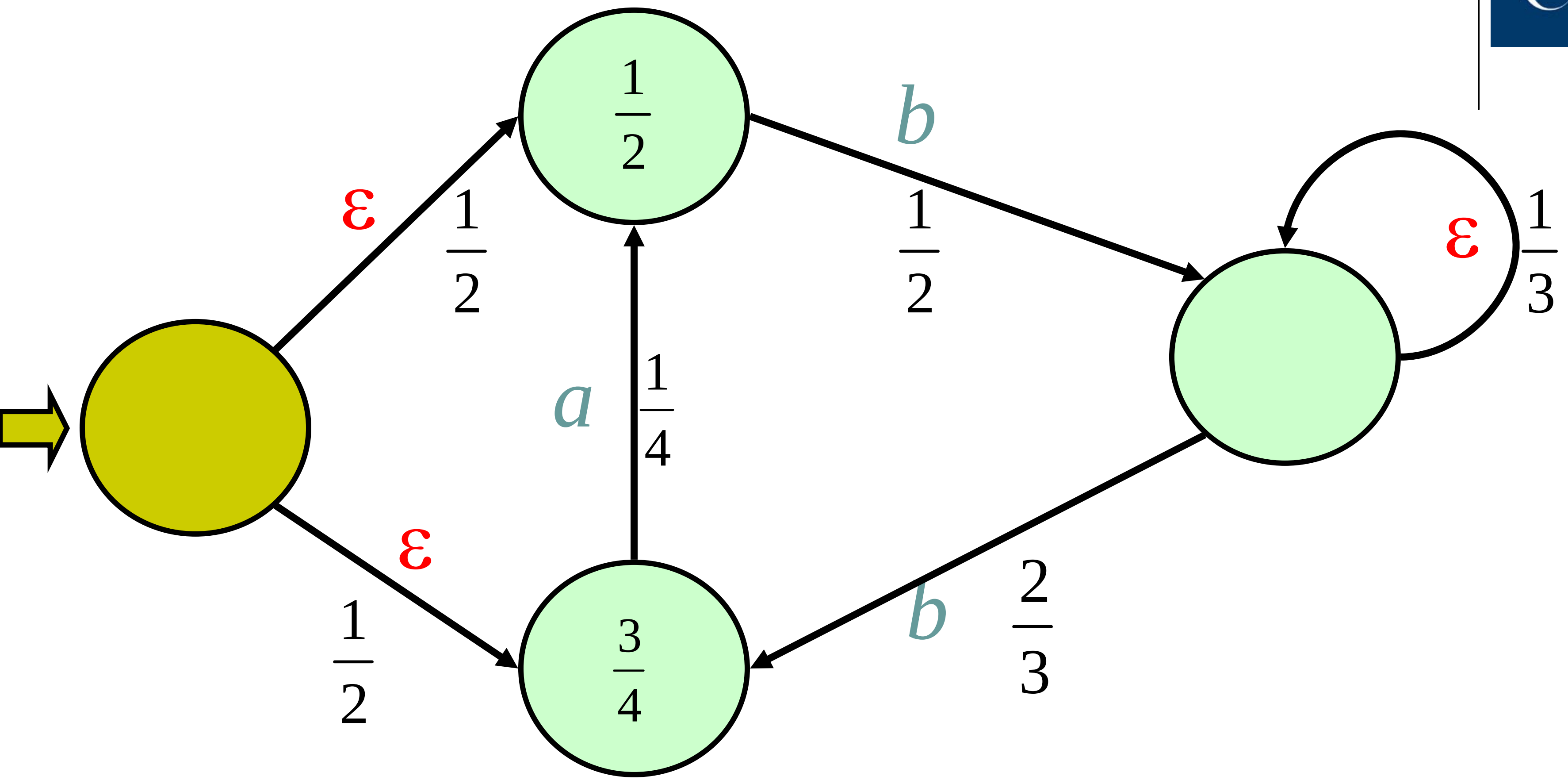


$$\Pr_A(abab) = \frac{1}{2} \times \frac{1}{2} \times \frac{1}{3} \times \frac{2}{3} \times \frac{3}{4} = \frac{1}{24}$$


0.1
b
0.9
a
0.35
a
0.7
a
0.7
b
0.65
b
0.3
0.3



PFA: Probabilistic Finite (state) Automaton



ε-*PFA*: Probabilistic Finite (state) Automaton with ε-transitions

How useful are these automata?

- They can define a distribution over Σ^*
- They do not tell us if a string belongs to a language
- They are good candidates for grammar induction
- There is (was?) not that much written theory

Basic references

- The *HMM* literature
- Azaria Paz 1973: **Introduction to probabilistic automata**
- Chapter 5 of my book
- **Probabilistic Finite-State Machines**, Vidal, Thollard, cdlh, Casacuberta & Carrasco
- *Grammatical Inference* papers

Automata, definitions

Let $\mathcal{D}$ be a distribution over Σ^*

$$0 \leq Pr_{\mathcal{D}}(w) \leq 1$$

$$\sum_{w \in \Sigma^*} Pr_{\mathcal{D}}(w) = 1$$

A Probabilistic Finite (state) Automaton is a $\langle Q, \Sigma, I_P, F_P, \delta_P \rangle$

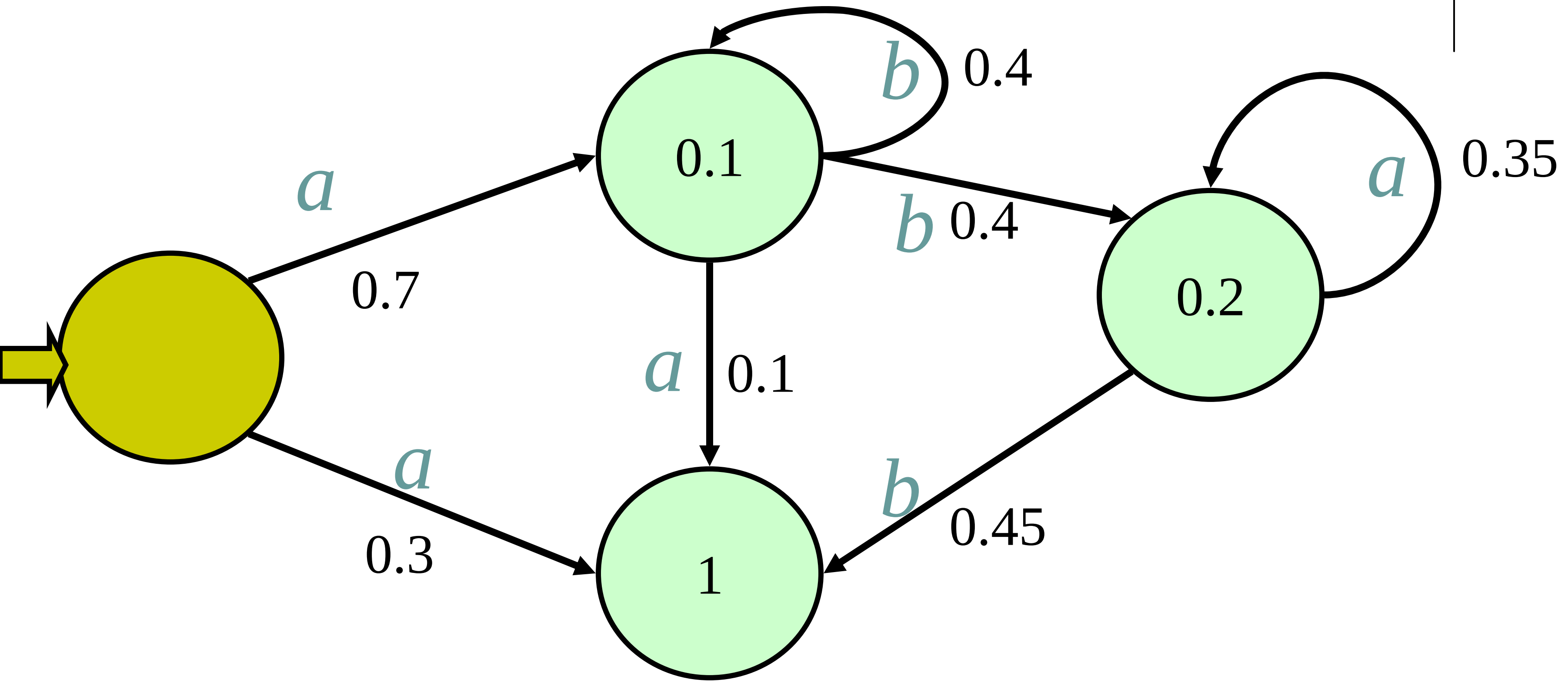
- Q set of states
- $I_P : Q \rightarrow [0;1]$
- $F_P : Q \rightarrow [0;1]$
- $\delta_P : Q \times \Sigma \times Q \rightarrow [0;1]$

What does a PFA do?

- It defines the probability of each string *w* as the sum (over all paths reading *w*) of the products of the probabilities

- $Pr_{\mathcal{A}}(w)=\sum_{\pi_i \in paths(w)} Pr(\pi_i)$
- $\pi_i=q_{i0}a_{i1}q_{i1}a_{i2}\ldots a_{in}q_{in}$
- $Pr(\pi_i)=I_P(q_{i0})\cdot F_P(q_{in}) \cdot \prod_{a_{ij}} \delta_P(q_{ij-1},a_{ij},q_{ij})$

- Note that if λ-transitions are allowed the sum may be infinite



Pr(*aba*) = 0.7*0.4*0.1*1 +0.7*0.4*0.45*0.2
= 0.028+0.0252=0.0532

- non deterministic *PFA*: many initial states/only one initial state
- an λ-*PFA*: a *PFA* with λ-transitions and perhaps many initial states
- *DPFA*: a deterministic *PFA*

Consistency

A PFA is consistent if

- $Pr_A(\Sigma^*)=1$
- $\forall x\in\Sigma^*\ 0\leq Pr_A(x)\leq 1$

Consistency theorem

A is consistent *if* every state is useful (accessible and co-accessible) and

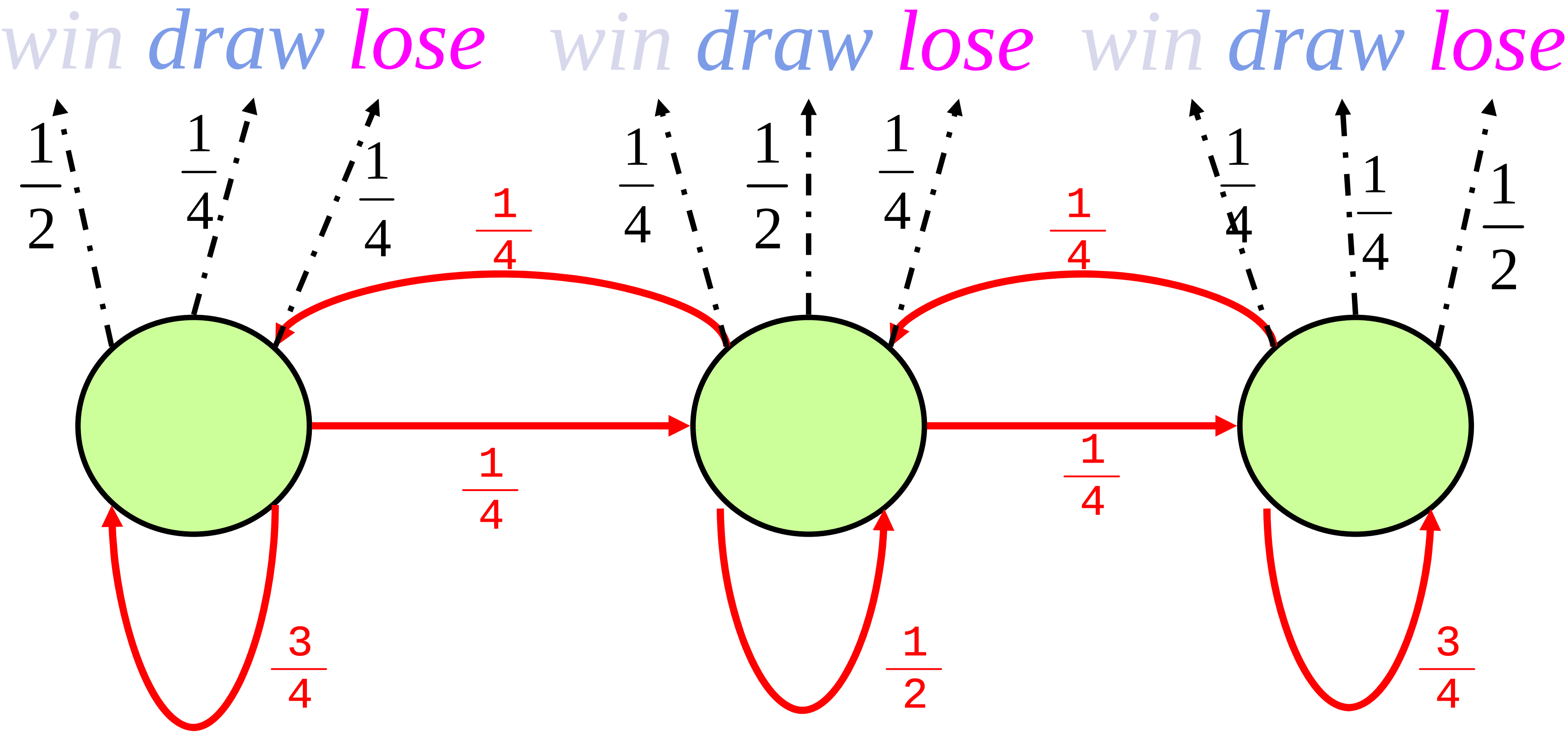
$$\forall q \in Q$$

$$F_P(q) + \sum_{q' \in Q, a \in \Sigma} \delta_P(q,a,q') = 1$$

Equivalence between models

- Equivalence between *PFA* and *HMM*…
- But the *HMM* usually define distributions over each Σ^n

A *football* HMM



Equivalence between *PFA* with λ-transitions and *PFA* without λ-transitions

cdlh 2003, Hanneforth & cdlh 2009

- Many initial states can be transformed into one initial state with λ-transitions;
- λ-transitions can be removed in polynomial time;
- Strategy:
  - number the states
  - eliminate first λ-loops, then the transitions with highest ranking arrival state

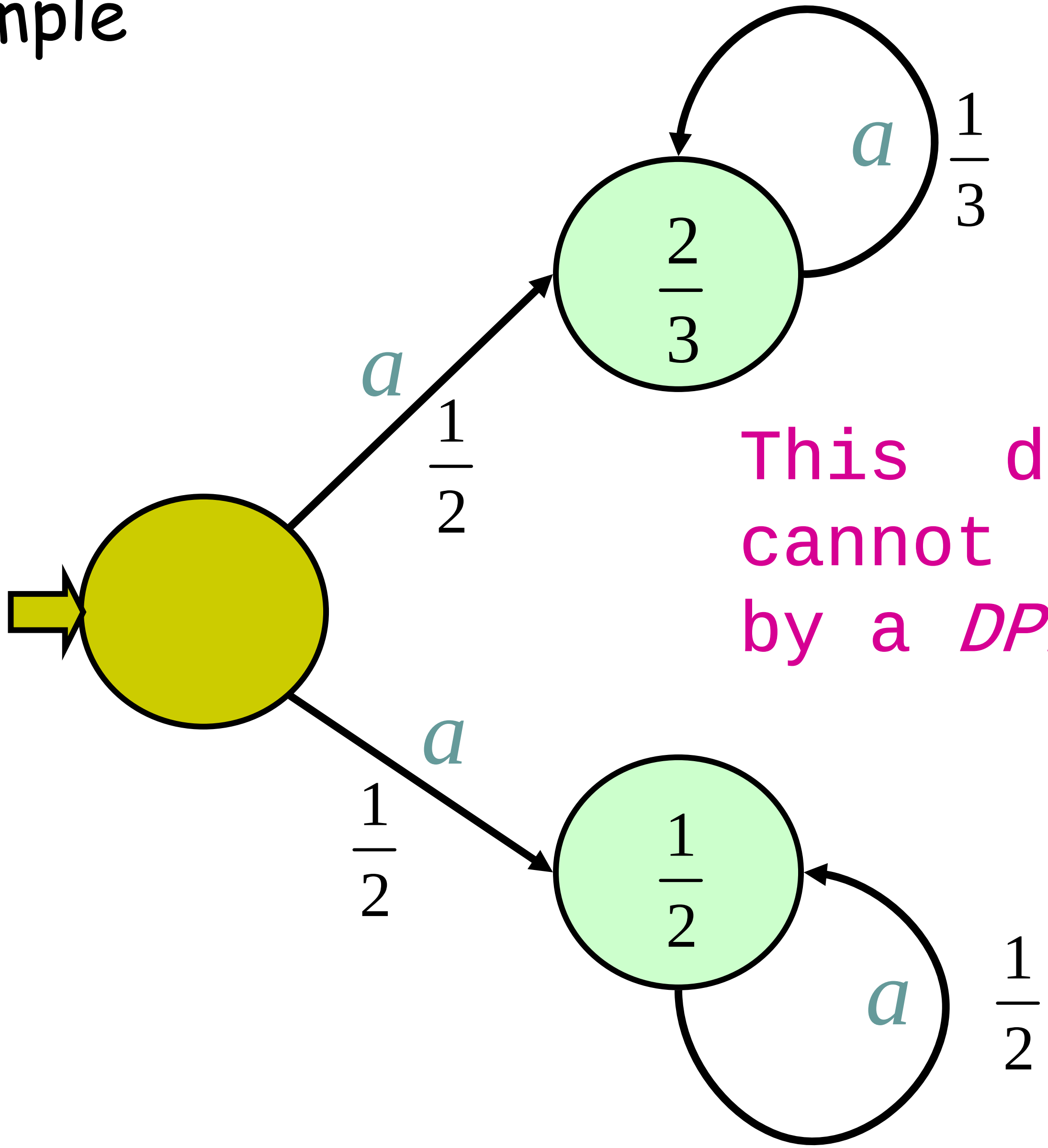
PFA are strictly more powerful than *DPFA*

Folk theorem

(and) You can't even tell in advance if you are in a good case or not

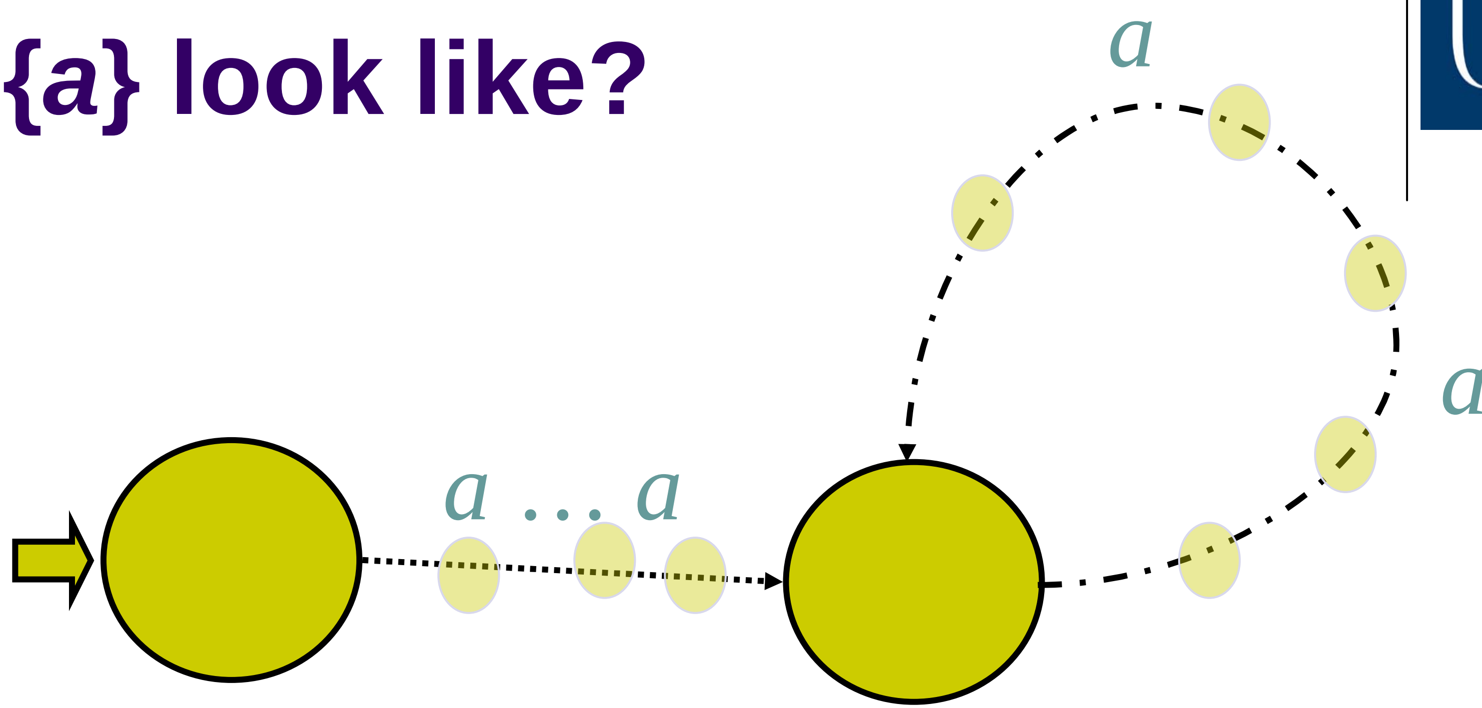
(see: Denis & Esposito 2004)

Example



This distribution cannot be modelled by a *DPFA*

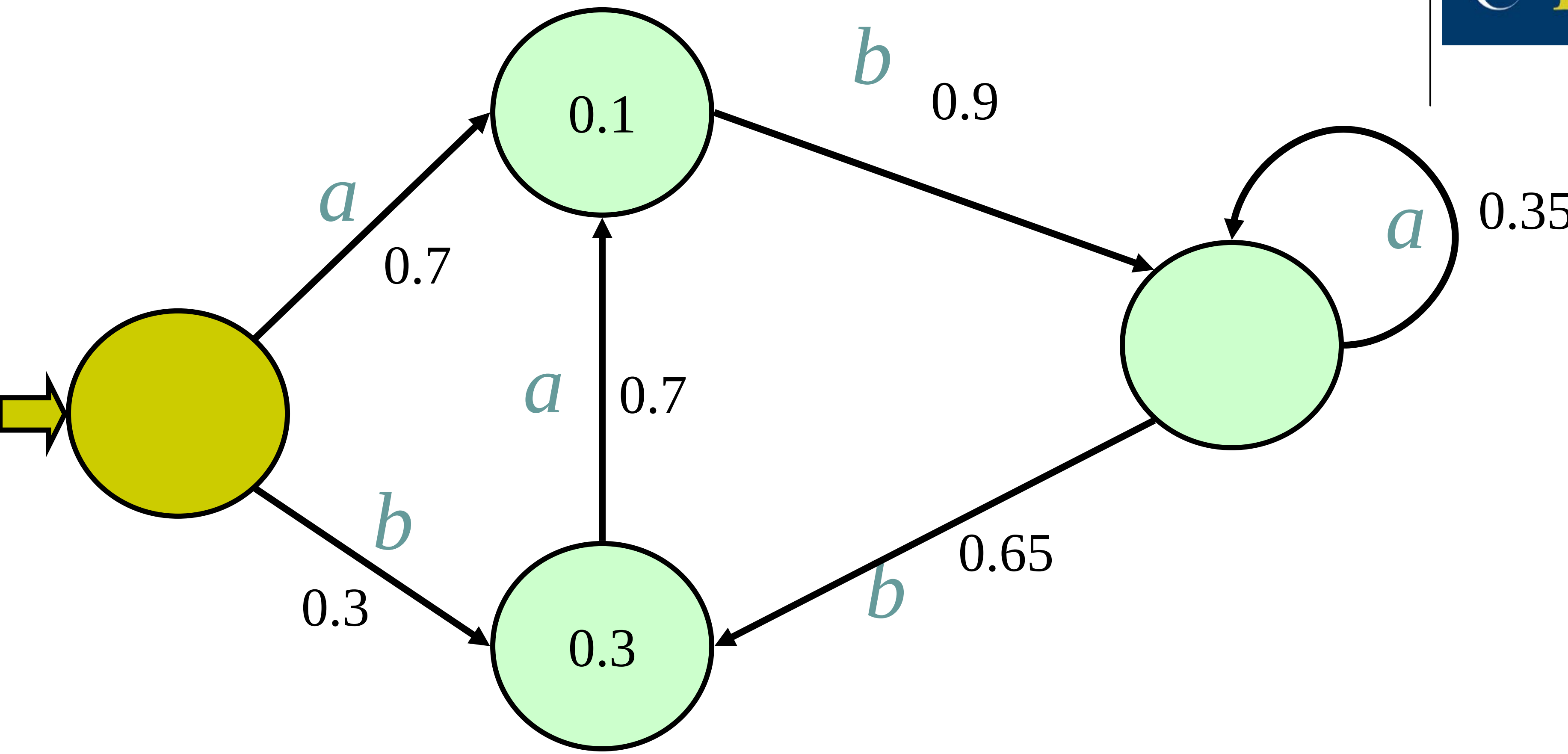
What does a *DPFA* over Σ ={*a*} look like?



And with this architecture you cannot generate the previous one

Parsing issues

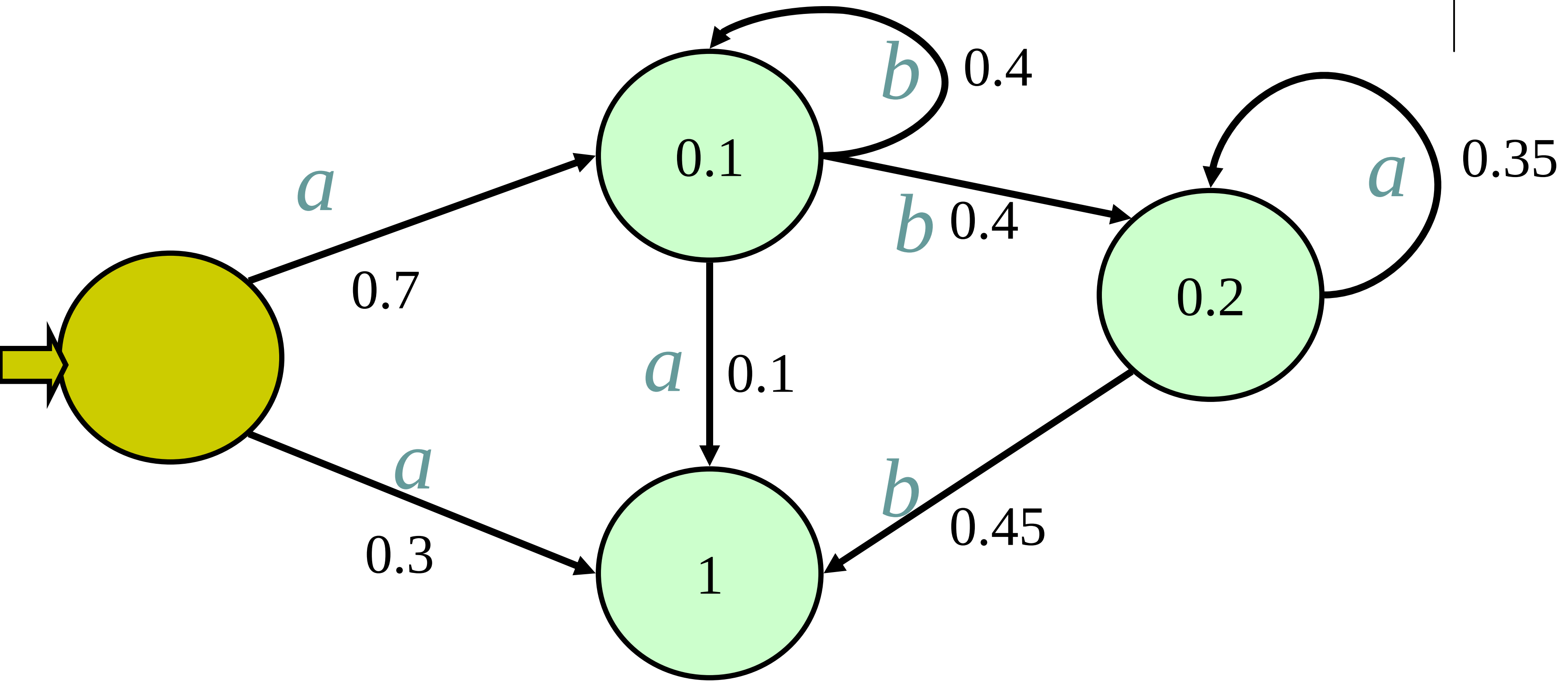
- Computation of the probability of a string or of a set of strings
- Deterministic case
  - Simple: apply definitions
  - Technically, rather sum up logs: this is easier, safer and cheaper



Pr(*aba*) = 0.7*0.9*0.35*0 = 0

Pr(*abb*) = 0.7*0.9*0.65*0.3 = 0.12285

Non-deterministic case



Pr(*aba*) = 0.7*0.4*0.1*1 +0.7*0.4*0.45*0.2
= 0.028+0.0252=0.0532

In the literature

- The computation of the probability of a string is by dynamic programming : $O(n^2 m)$
- 2 algorithms: *Backward* and *Forward*
- If we want the most probable derivation to define the probability of a string, then we can use the *Viterbi* algorithm

Forward algorithm

- $A[i,j]=\Pr(q_i | a_1..a_j)$
  (The probability of being in state q_i after having read $a_1..a_j$)
- $A[i,0]=I_P(q_i)$
- $A[i,j+1]= \sum_{k\leq|Q|} A[k,j] \cdot \delta_P(q_k,a_{j+1},q_i)$
- $\Pr(a_1..a_n)= \sum_{k\leq|Q|} A[k,n] \cdot F_P(q_k)$

2 Distances

What for?

- Estimate the quality of a language model
- Have an indicator of the convergence of learning algorithms
- Construct kernels

2.1 Entropy

- How many bits do we need to correct our model?
- Two distributions over Σ^*: $\mathcal{D}$ et $\mathcal{D}'$
- *Kullback Leibler divergence* (or relative entropy) between $\mathcal{D}$ and $\mathcal{D}'$:

$$\sum_{w \in \Sigma^*} Pr_{\mathcal{D}}(w) \times \left| \log Pr_{\mathcal{D}}(w) - \log Pr_{\mathcal{D}'}(w) \right|$$

2.2 Perplexity

- The idea is to allow the computation of the divergence, but relatively to a test set (*S*)
- An approximation (*sic*) is perplexity: inverse of the geometric mean of the probabilities of the elements of the test set

$$\prod_{w \in S} \Pr_{\mathcal{D}}(w)^{-1/|S|}$$

$$=$$

$$\frac{1}{\sqrt[|S|]{\prod_{w \in S} \Pr_{\mathcal{D}}(w)}}$$

Problem if some probability is null...

Why multiply (1)

- We are trying to compute the probability of independently drawing the different strings in set S

Why multiply? (2)

- Suppose we have two predictors for a coin toss
  - Predictor 1: heads 60%, tails 40%
  - Predictor 2: heads 100%
- The tests are H: 6, T: 4
- Arithmetic mean
  - P1: 36%+16%=0,52
  - P2: 0,6
- Predictor 2 is the better predictor ;-)

2.3 Distance d_2

$$d_2(\mathcal{D}, \mathcal{D}') = \sqrt{\sum_{w \in \Sigma^*} (\Pr_{\mathcal{D}}(w) - \Pr_{\mathcal{D}'}(w))^2}$$

Can be computed in polynomial time if $\mathcal{D}$ and $\mathcal{D}'$ are given by *PFA* (Carrasco & cdlh 2002)

This also means that equivalence of PFA is in **P**

3 FFA

Frequency Finite (state) Automata

A learning sample

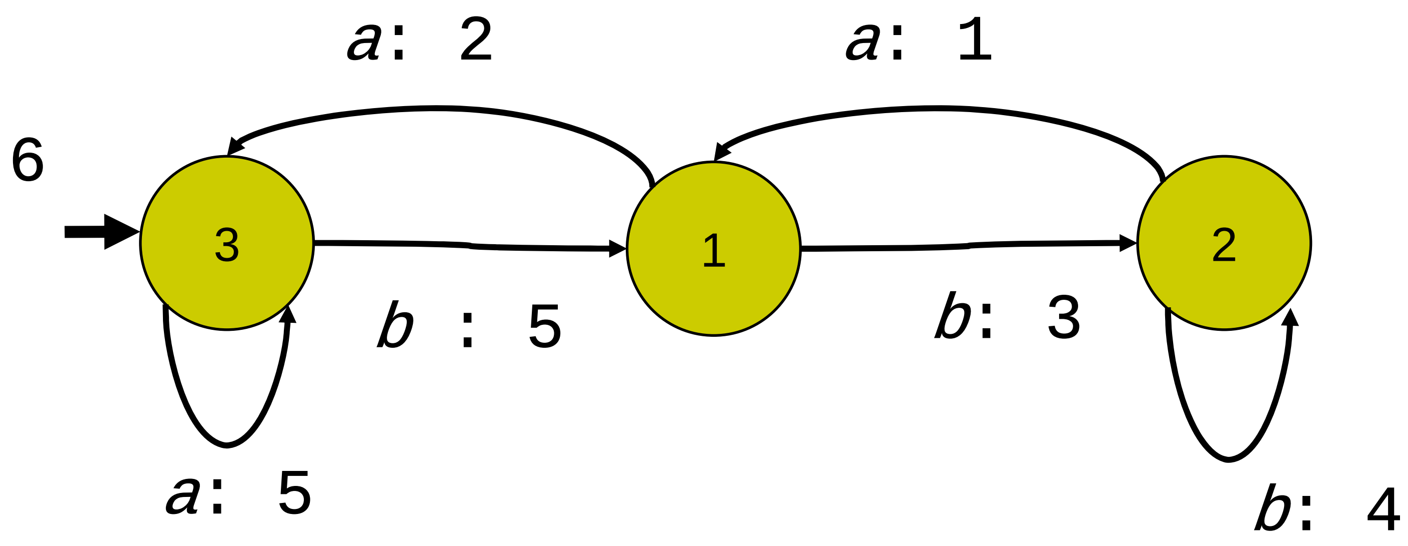
- is a multiset
- Strings appear with a frequency (or multiplicity)
- S={λ (3), *aaa* (4), *aaba* (2), *ababa* (1), *bb* (3), *bbaaa* (1)}

DFFA

A deterministic frequency finite automaton is a DFA with a frequency function returning a positive integer for every state and every transition, and for entering the initial state such that

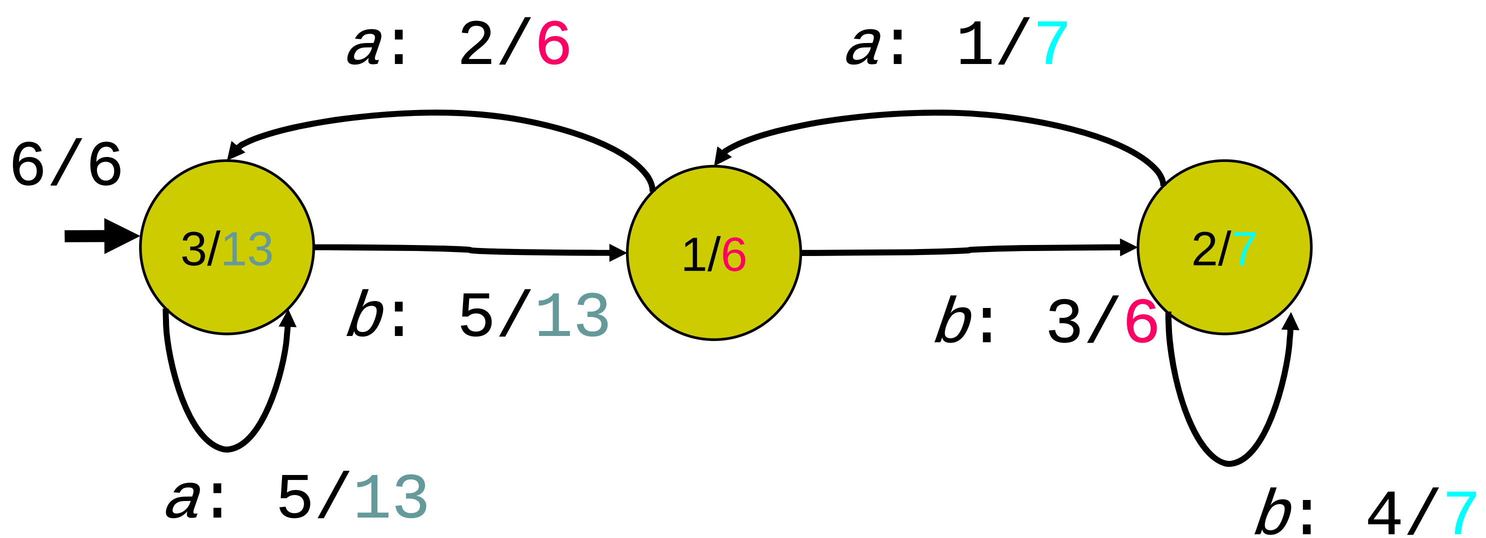
- the sum of what enters is equal to what exits and
- the sum of what halts is equal to what starts

Example



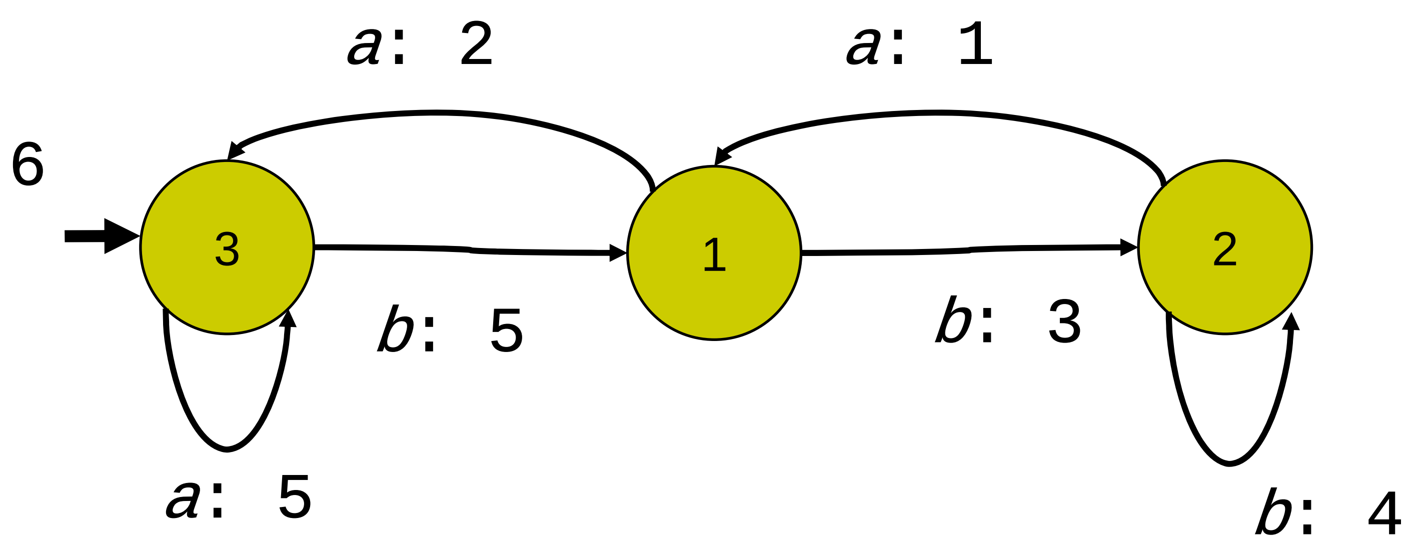
From a DFFA to a DPFA

Frequencies become relative frequencies by dividing by sum of exiting frequencies



From a DFA and a sample to a DFFA

S = {λ, *aaaa*, *ab*, *babb*, *bbbb*, *bbbbaa*}



Note

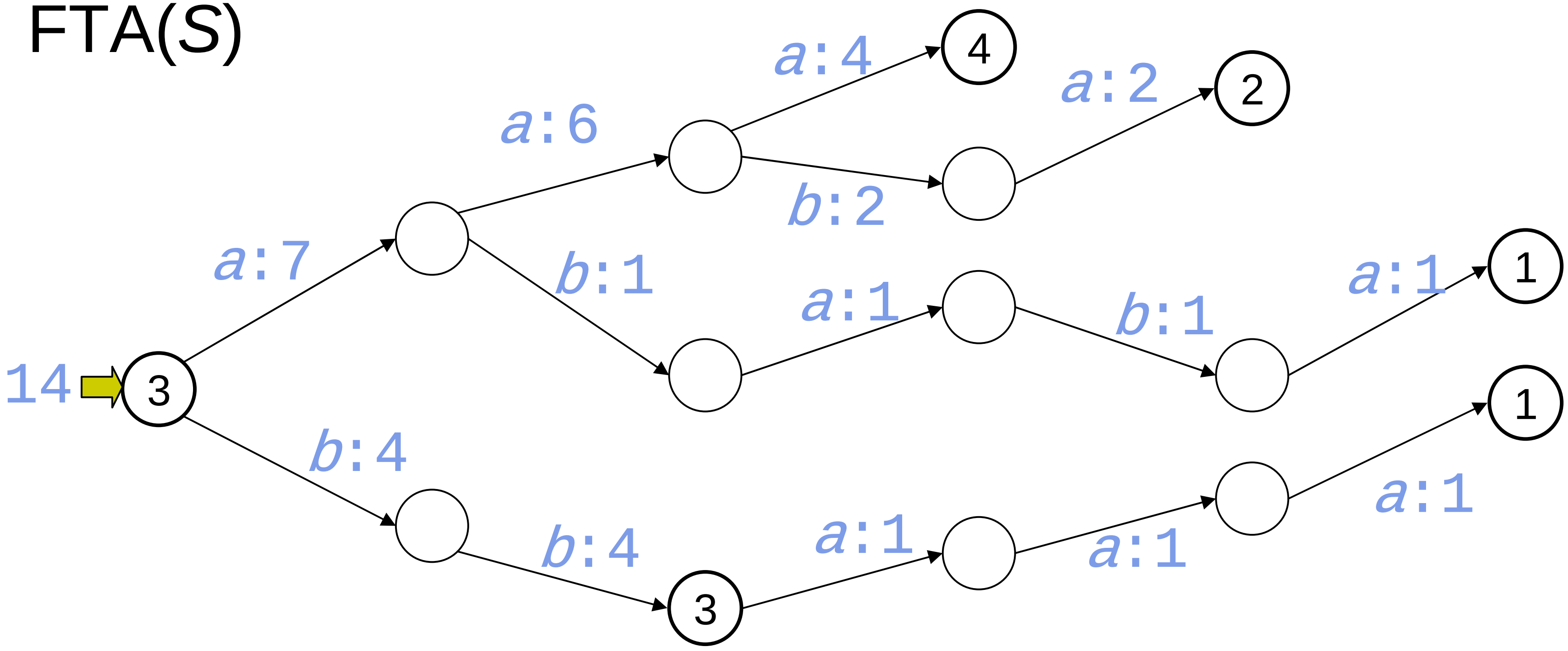
- Another sample may lead to the same DFFA
- Doing the same with a NFA is a much harder problem
- Typically what algorithm Baum-Welch (EM) has been invented for…

The frequency prefix tree acceptor

- The data is a multi-set
- The FTA is the smallest tree-like FFA consistent with the data
- Can be transformed into a PFA if needed

From the sample to the FTA

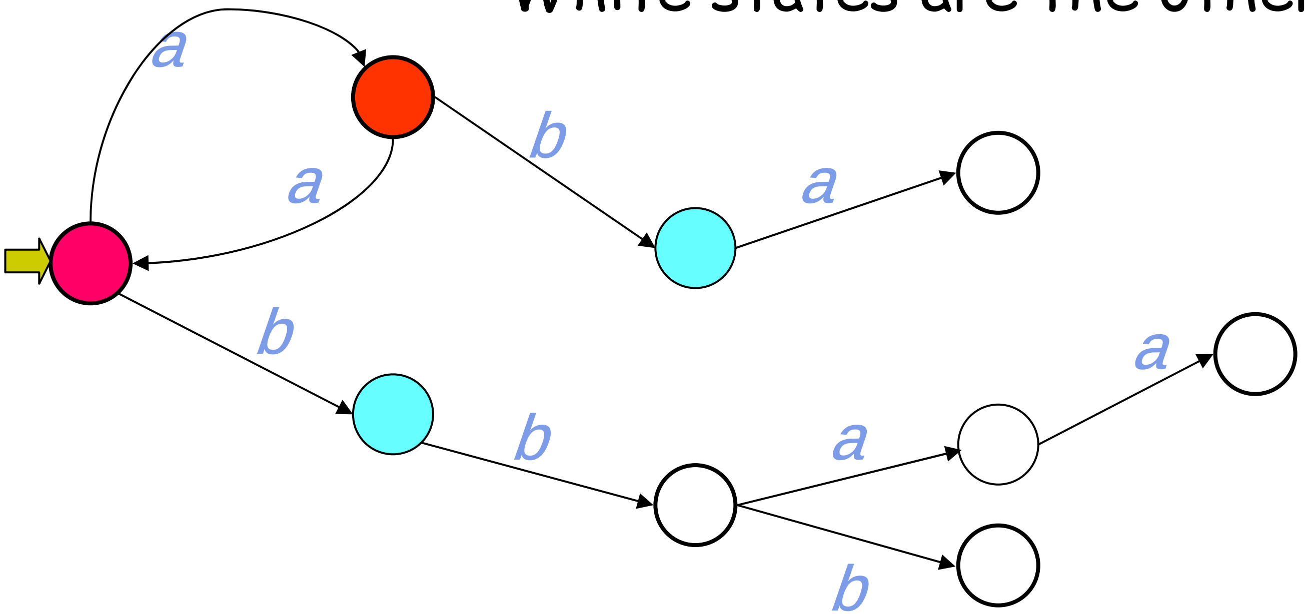
FTA(*S*)



S={λ (3), *aaa* (4), *aaba* (2), *ababa* (1), *bb* (3), *bbaaa* (1)}

Red, Blue and White states

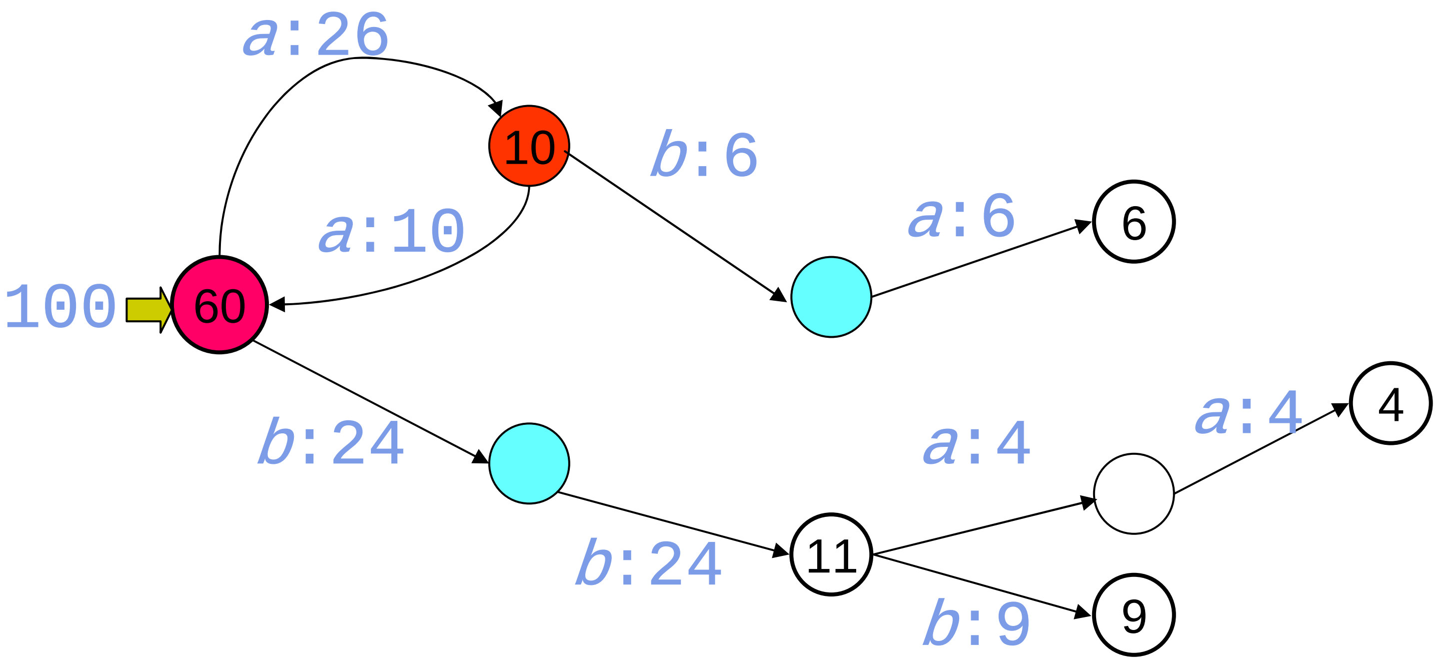
-Red states are confirmed states
-Blue states are the (non Red) successors of the Red states
-White states are the others



Same as with DFA and what RPNI does

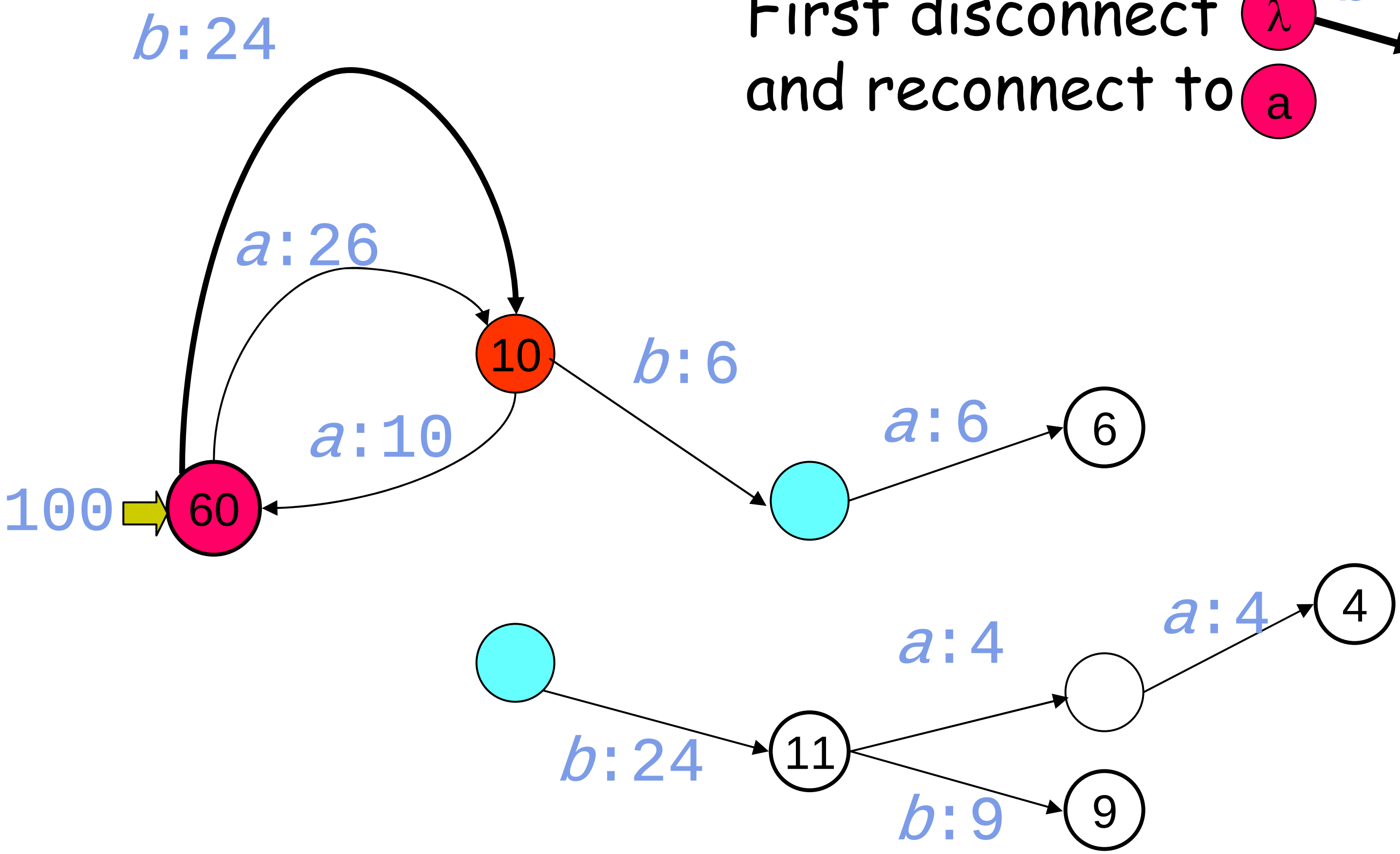
Merge and fold

Suppose we decide to merge λ —*b*→ with state a



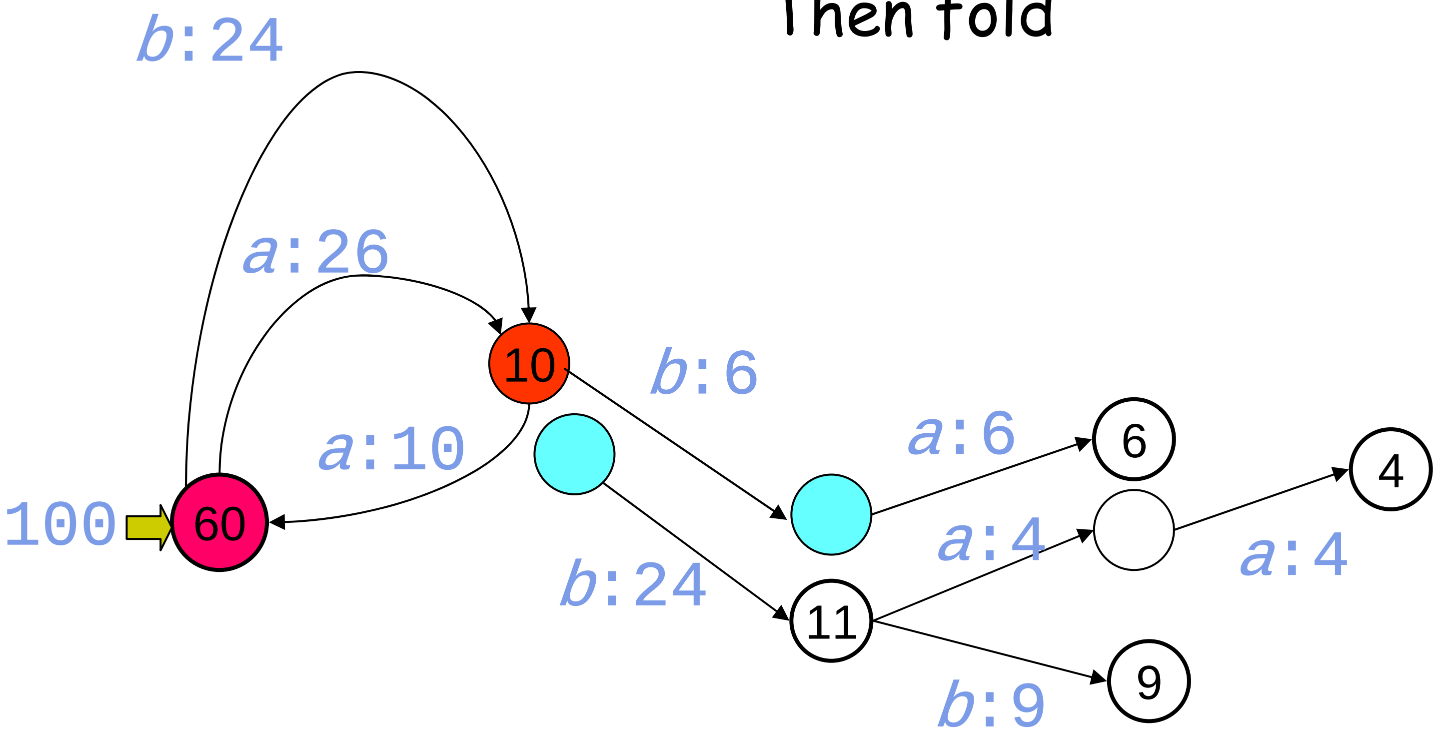
Merge and fold

First disconnect λ $\xrightarrow{b}$ and reconnect to a



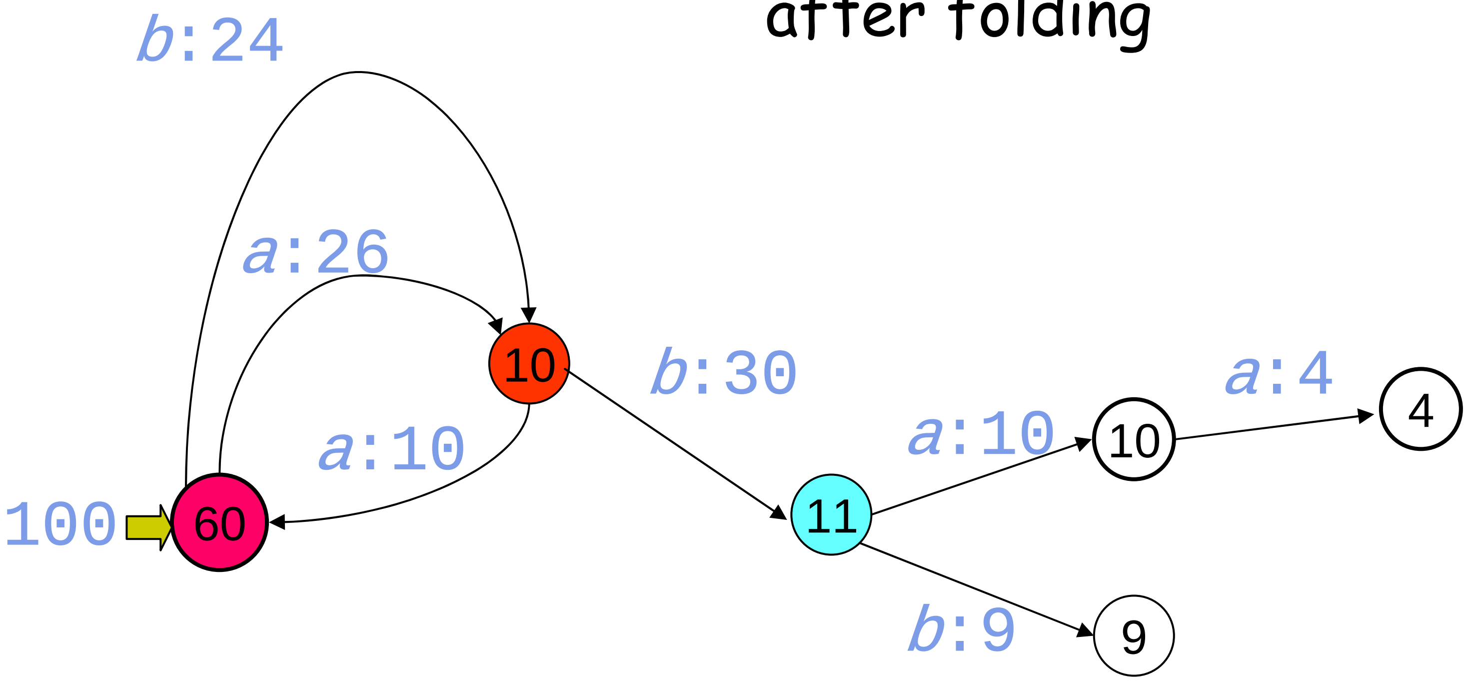
Merge and fold

Then fold



Merge and fold

after folding



State merging algorithm

A=FTA(S); *Blue* ={$\delta(q_I,a)$: $a\in\Sigma$ };

Red ={q_I}

While *Blue*$\neq\varnothing$ do

choose q from *Blue* such that Freq(q)$\geq t_0$

if $\exists p\in$ *Red*: d(A_p,A_q) is small

then A = merge_and_fold(A,p,q)

else *Red* = *Red* $\cup$ {q}

Blue = {$\delta(q,a)$: $q\in$ *Red*} – {*Red*}

The real question

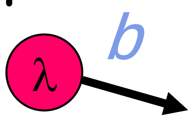
- How do we decide if d(A_p,A_q) is small?
- Use a distance…
- Be able to compute this distance
- If possible update the computation easily
- Have properties related to this distance

Deciding if two distributions are similar

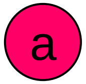
- If the two distributions are known, equality can be tested
- Distance (L_2 norm) between distributions can be exactly computed
- But what if the two distributions are unknown?

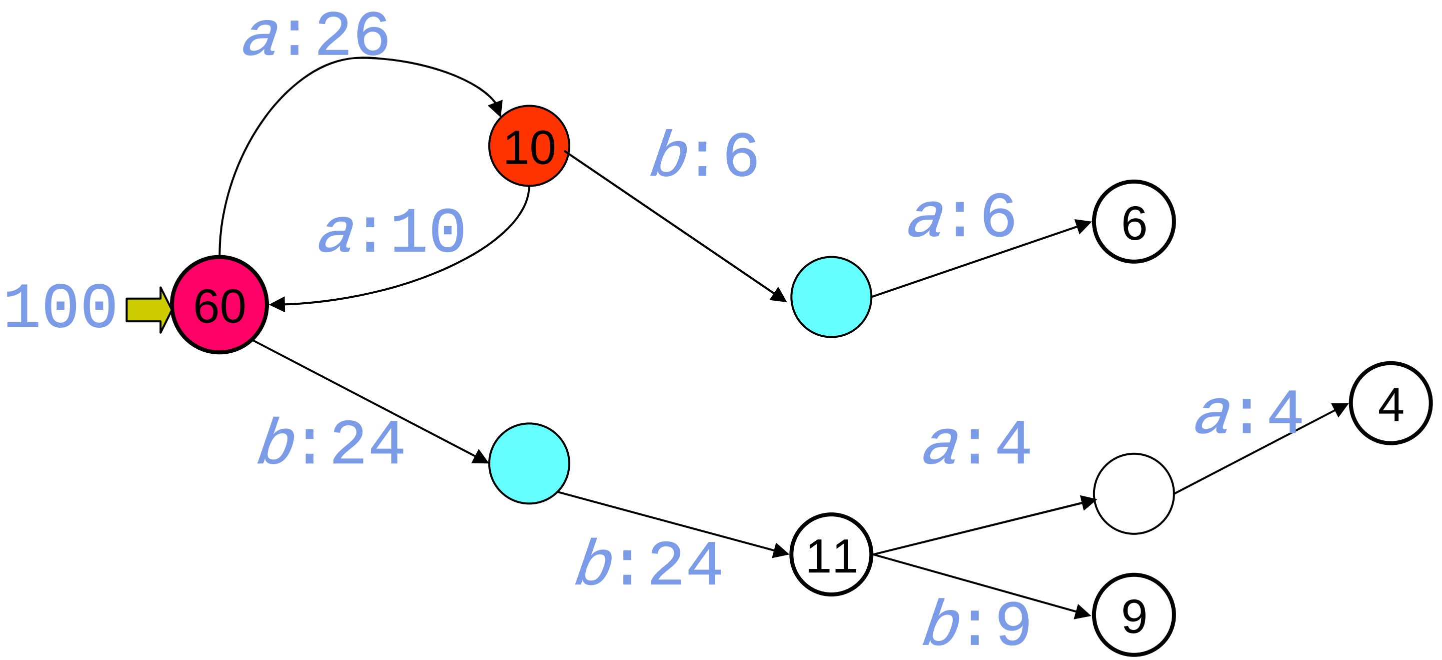
Taking decisions

Suppose we want to merge



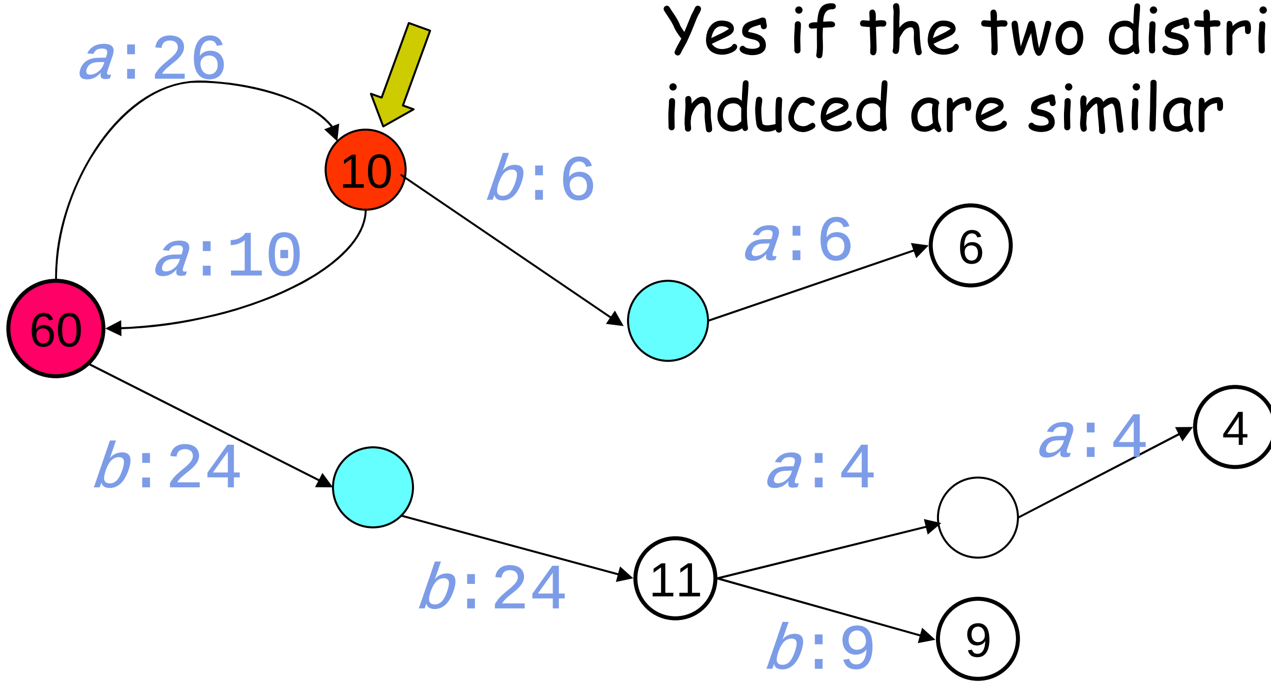
with state

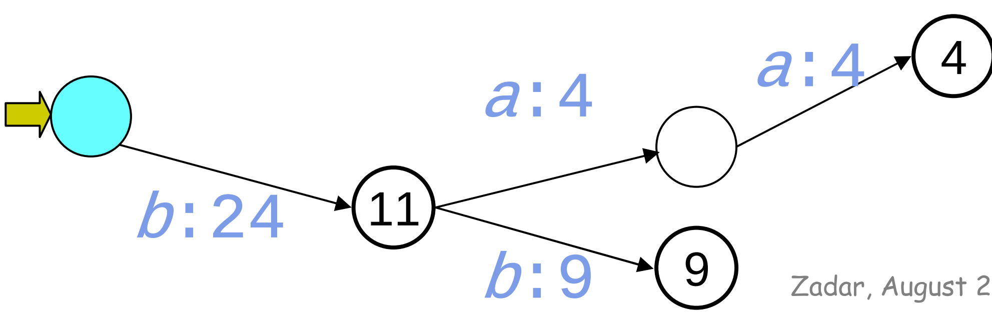




Taking decisions

Yes if the two distributions induced are similar





5 Alergia

Alergia's test

- $\mathcal{D}_1 \approx \mathcal{D}_2$ if $\forall x\, Pr_{\mathcal{D}1}(x) \approx Pr_{\mathcal{D}2}(x)$
- Easier to test:
  - $Pr_{\mathcal{D}1}(\lambda)=Pr_{\mathcal{D}2}(\lambda)$
  - $\forall a\in\Sigma\ Pr_{\mathcal{D}1}(a\Sigma^*)=Pr_{\mathcal{D}2}(a\Sigma^*)$
- And do this recursively!
- Of course, do it on frequencies

Hoeffding bounds

$$\gamma \leftarrow \left| \frac{f_1}{n_1} - \frac{f_2}{n_2} \right|$$

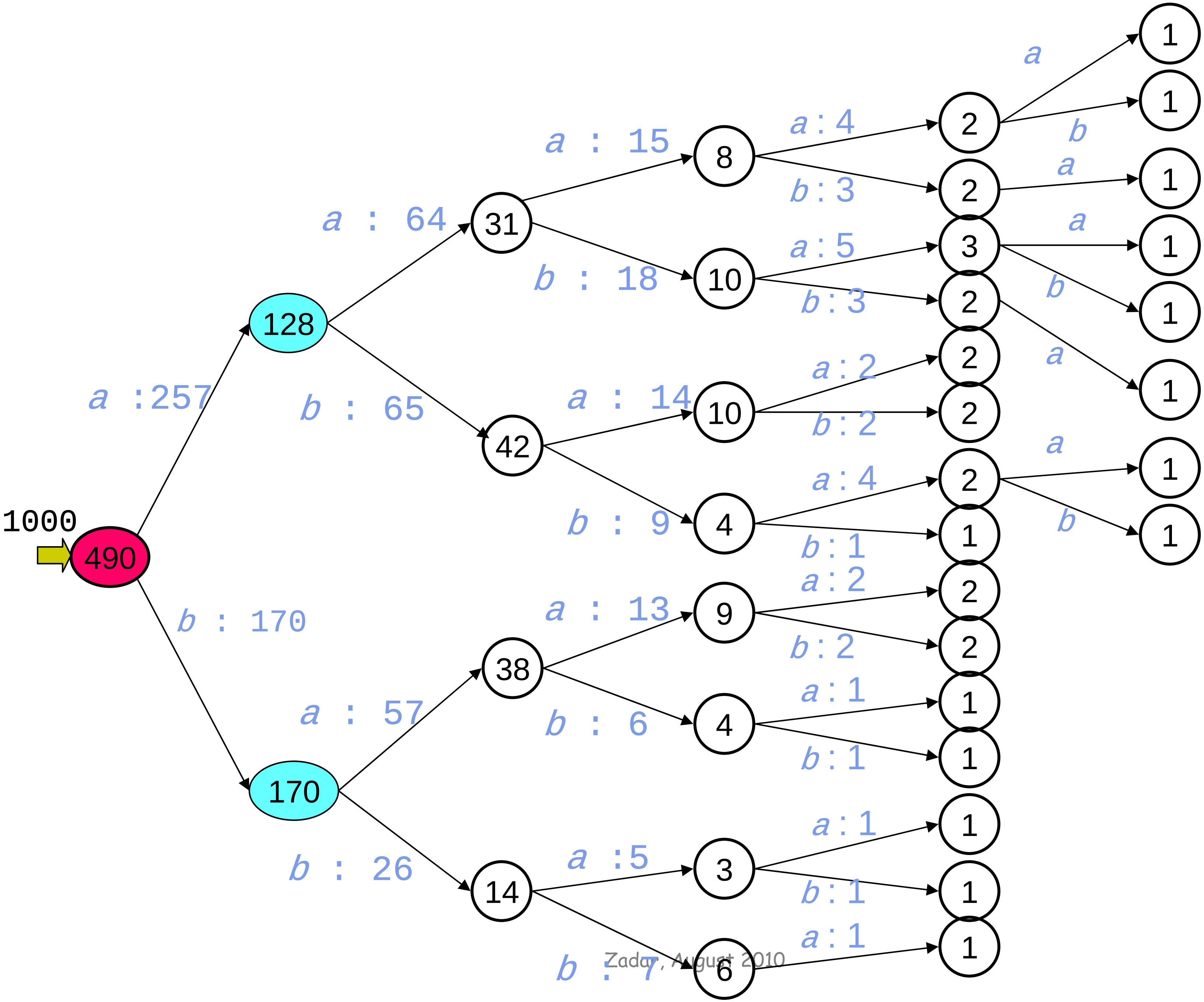
$$\gamma < \left(\sqrt{\frac{1}{n_1}} + \sqrt{\frac{1}{n_2}} \right) . \sqrt{\frac{1}{2} \ln \frac{2}{\alpha}}$$

γ indicates if the relative frequencies $\frac{f_1}{n_1}$ and $\frac{f_2}{n_2}$ are sufficiently close

A run of Alergia
Our learning multisample

S={λ(490), *a*(128), *b*(170), *aa*(31), *ab*(42), *ba*(38), *bb*(14), *aaa*(8), *aab*(10), *aba*(10), *abb*(4), *baa*(9), *bab*(4), *bba*(3), *bbb*(6), *aaaa*(2), *aaab*(2), *aaba*(3), *aabb*(2), *abaa*(2), *abab*(2), *abba*(2), *abbb*(1), *baaa*(2), *baab*(2), *baba*(1), *babb*(1), *bbaa*(1), *bbab*(1), *bbba*(1), *aaaaa*(1), *aaaab*(1), *aaaba*(1), *aabaa*(1), *aabab*(1), *aabba*(1), *abbaa*(1), *abbab*(1)}

- Parameter α is arbitrarily set to 0.05. We choose 30 as a value for threshold t_0.
- Note that for the blue state who have a frequency less than the threshold, a special merging operation takes place

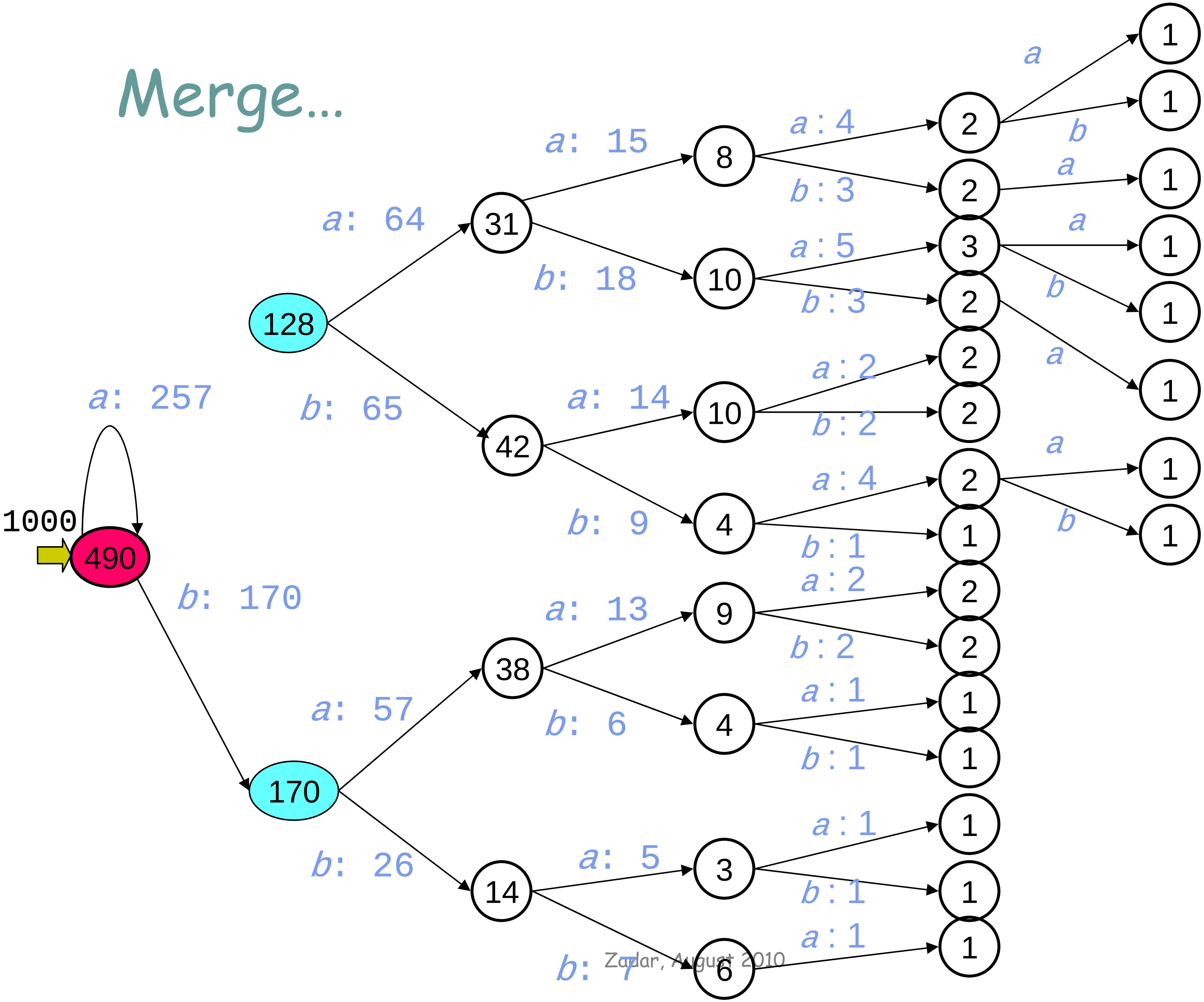

1000
490
a :257
b : 170
128
170
a : 64
b : 65
a : 57
b : 26
31
42
38
14
a : 15
b : 18
a : 14
b : 9
a : 13
b : 6
a :5
b : 7
8
10
10
4
9
4
3
6
a : 4
b : 3
a : 5
b : 3
a : 2
b : 2
a : 4
b : 1
a : 2
b : 2
a : 1
b : 1
a : 1
b : 1
a : 1
2
2
3
2
2
2
2
1
2
2
1
1
1
1
1
a
b
a
a
b
a
a
b
1
1
1
1
1
1
1
1

Can we merge λ and *a*?

- Compare λ and *a*, *a*Σ* and *aa*Σ*, *b*Σ* and *ab*Σ*
- 490/1000 with 128/257 ,
- 257/1000 with 64/257 ,
- 253/1000 with 65/257 , ...

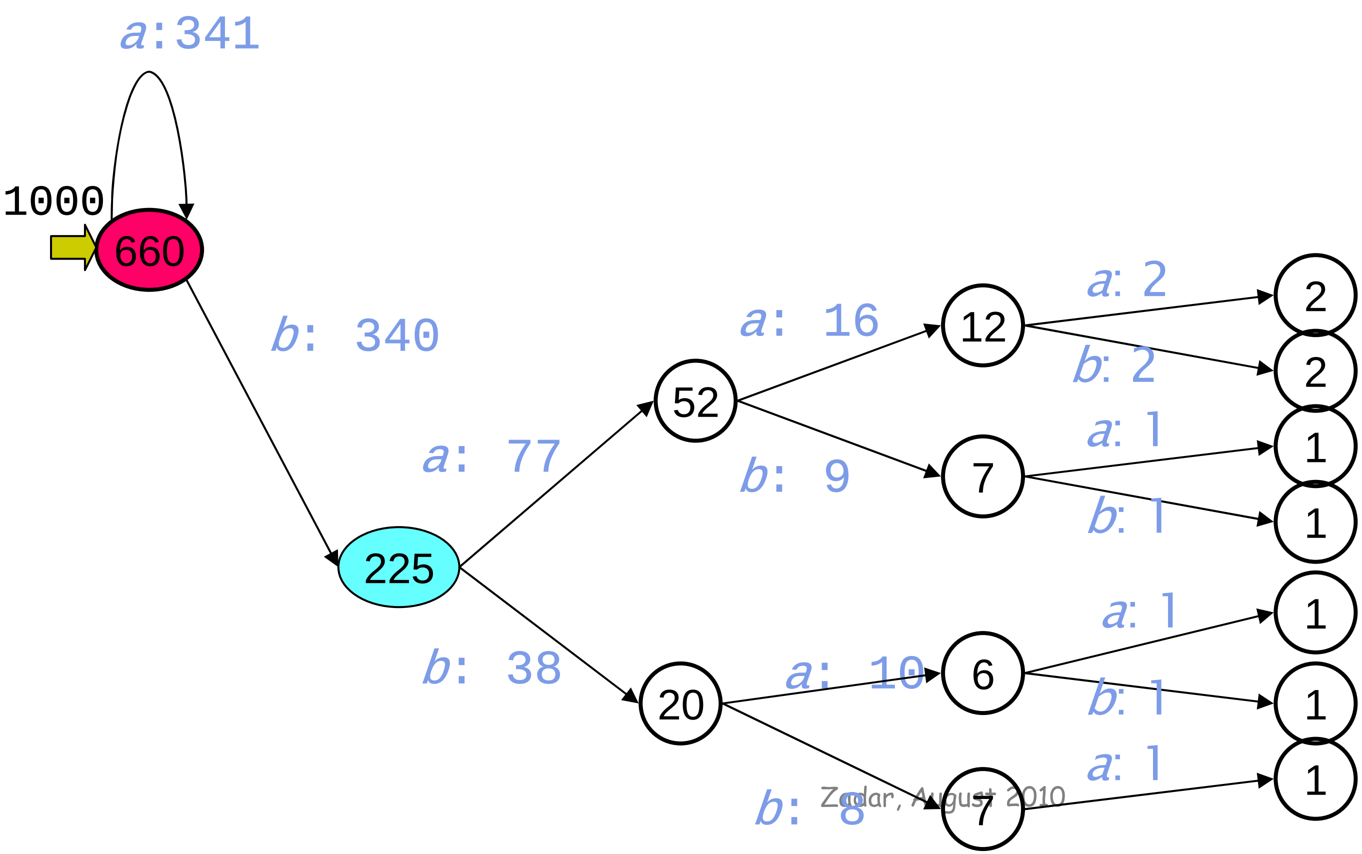
- All tests return true

Merge…

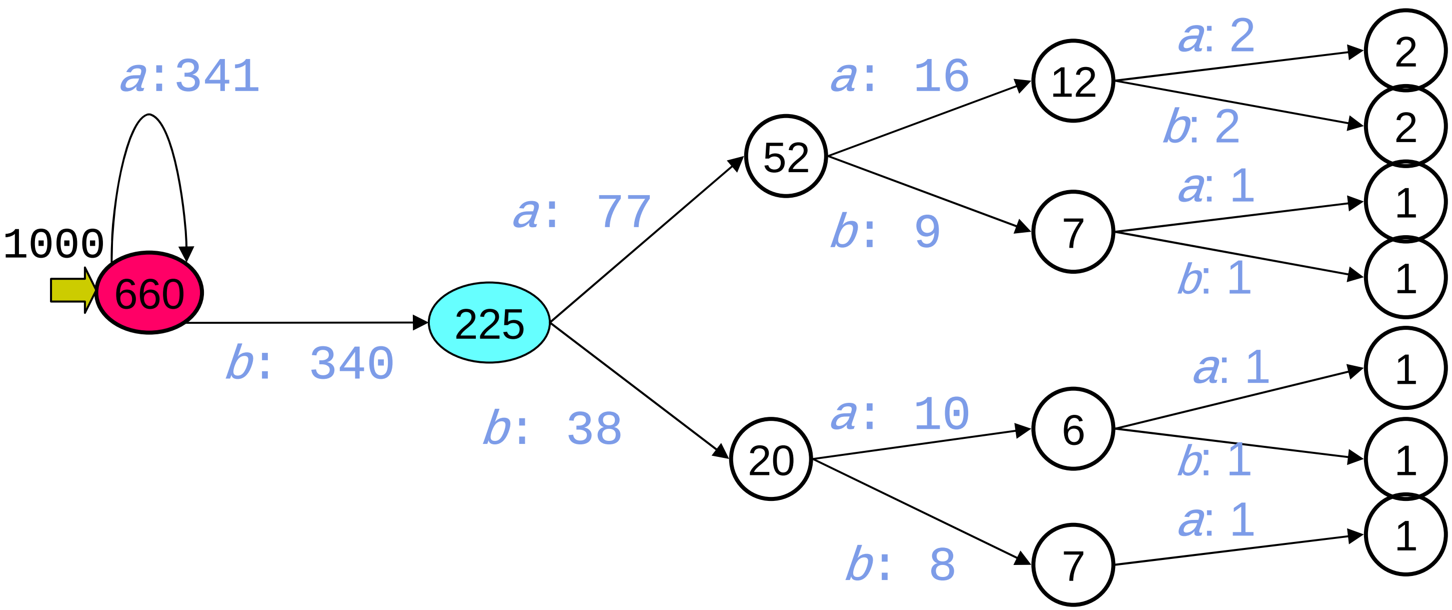


1000
490
a: 257
b: 170
128
170
a: 64
b: 65
a: 57
b: 26
31
42
38
14
a: 15
b: 18
a: 14
b: 9
a: 13
b: 6
a: 5
b: 7
8
10
10
4
9
4
3
6
a : 4
b : 3
a : 5
b : 3
a : 2
b : 2
a : 4
b : 1
a : 2
b : 2
a : 1
b : 1
a : 1
b : 1
a : 1
2
2
3
2
2
2
2
1
2
2
1
1
1
1
1
a
b
a
a
b
a
a
b
1
1
1
1
1
1
1
1

And fold


a:341
1000
660
b: 340
225
a: 77
b: 38
52
20
a: 16
b: 9
a: 10
b: 8
12
7
6
7
a: 2
b: 2
a: 1
b: 1
a: 1
b: 1
a: 1
2
2
1
1
1
1
1

Next merge ?
λ with b ?

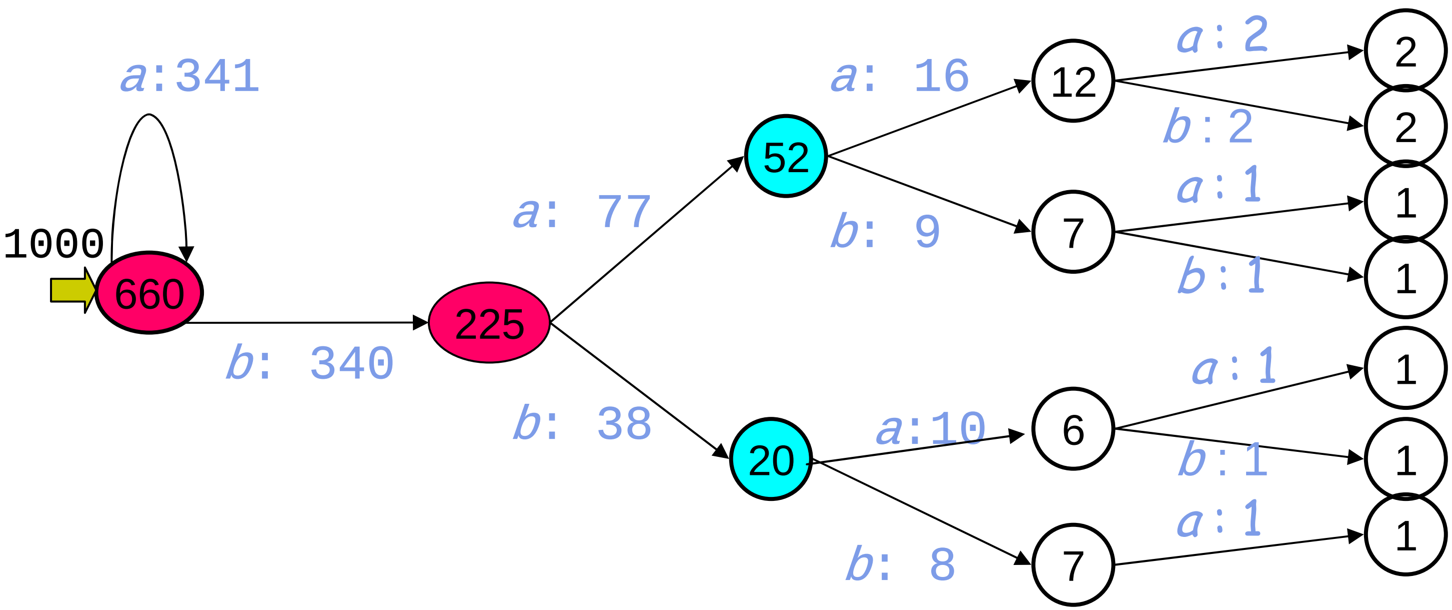


Can we merge λ and *b*?

- Compare λ and *b*, *a*Σ* and *ba*Σ*, *b*Σ* and *bb*Σ*
- 660/1341 and 225/340 are different (giving γ= 0.162)
- On the other hand

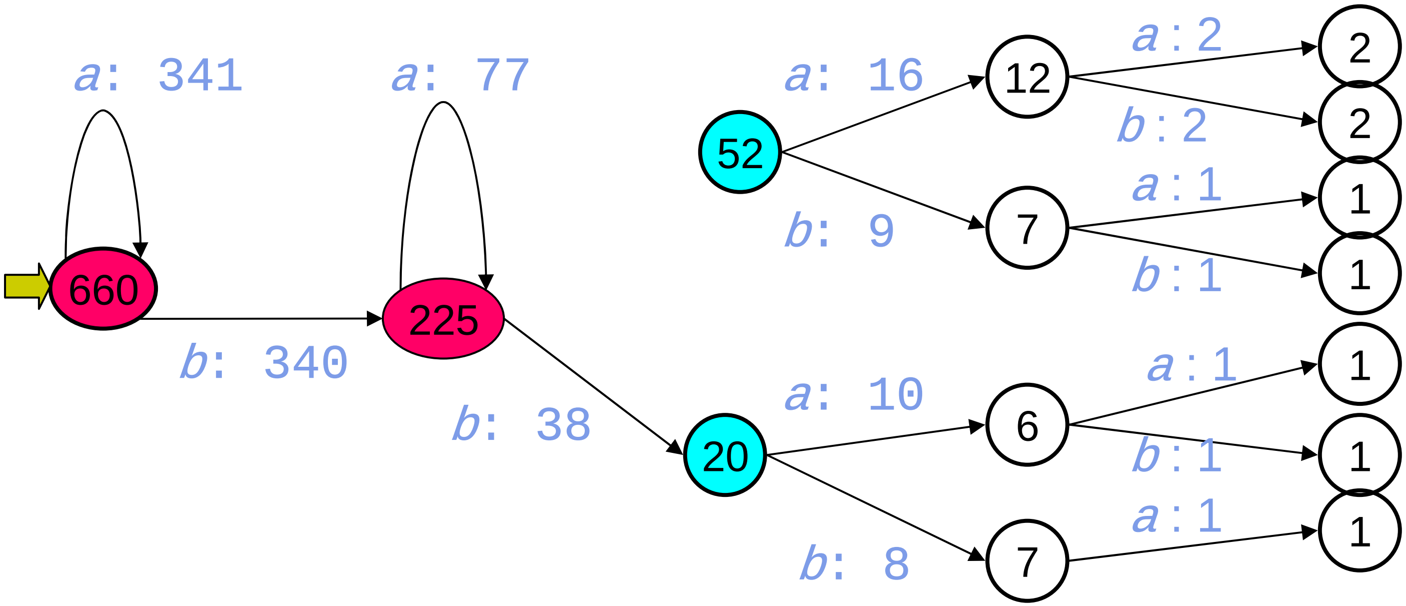
$$\left(\sqrt{\frac{1}{n_1}}+\sqrt{\frac{1}{n_2}}\right)\cdot\sqrt{\frac{1}{2}\ln\frac{2}{\alpha}}=0.111$$

Promotion



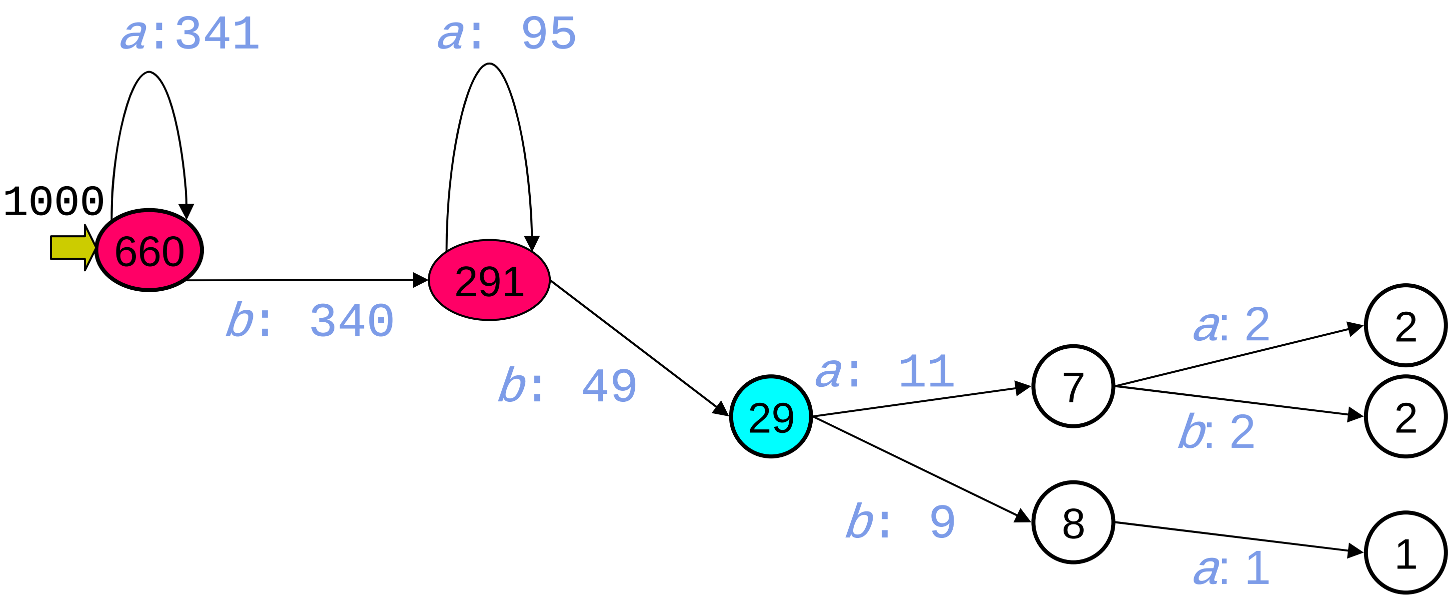
a:341
1000
660
b: 340
225
a: 77
b: 38
52
20
a: 16
b: 9
a:10
b: 8
12
7
6
7
a:2
b:2
a:1
b:1
a:1
b:1
a:1
2
2
1
1
1
1
1

Merge

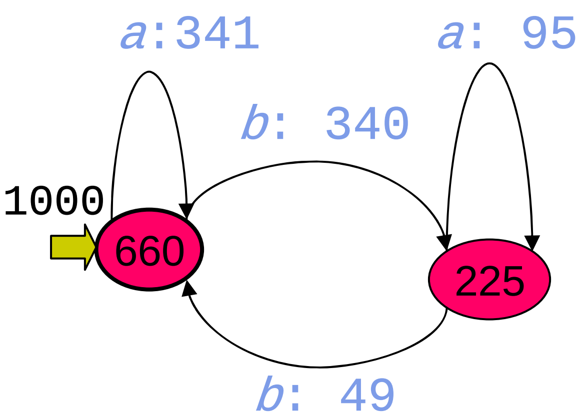


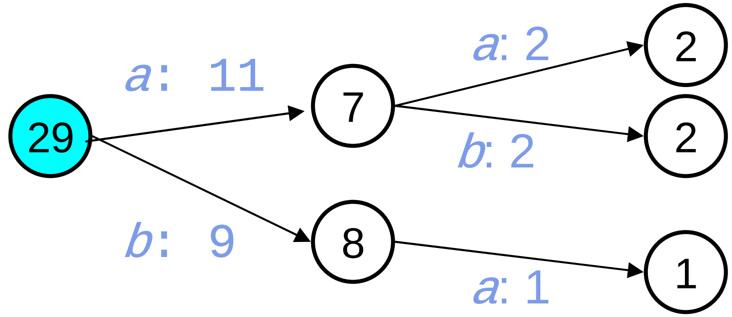
a: 341
a: 77
660
225
b: 340
b: 38
a: 16
b: 9
52
20
a: 10
b: 8
12
7
6
7
a:2
b:2
a:1
b:1
a:1
b:1
a:1
2
2
1
1
1
1
1

And fold

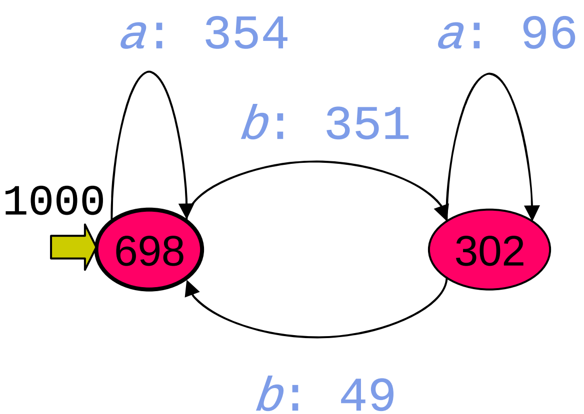


Merge

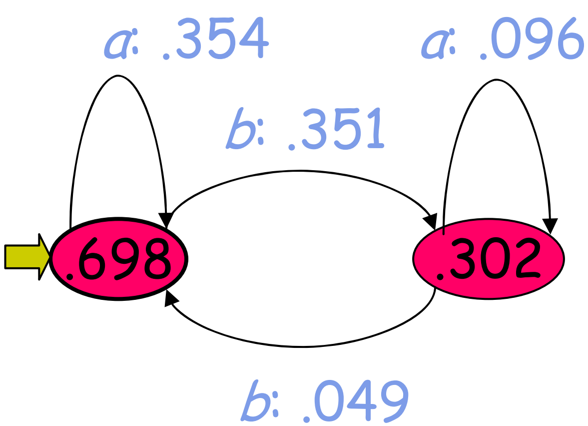




And fold



As a PFA



Conclusion and logic

- Alergia builds a DFFA in polynomial time
- Alergia can identify DPFA in the limit with probability 1
- No good definition of Alergia's properties

6 DSAI and MDI

Why not change the criterion?

Criterion for DSAI

- Using a distinguishable string
- Use norm L_∞
- Two distributions are different if there is a string with a very different probability
- Such a string is called μ-distinguishable
- Question becomes:

Is there a string x such that

$$|Pr_{A,q}(x)-Pr_{A,q'}(x)|>\mu$$

(much more to DSAI)

- D. Ron, Y. Singer, and N. Tishby. On the learnability and usage of acyclic probabilistic finite automata. In *Proceedings of Colt 1995*, pages 31–40, 1995.
- PAC learnability results, in the case where targets are acyclic graphs

Criterion for MDI

- MDL inspired heuristic
- Criterion is: does the reduction of the size of the automaton compensate for the increase in preplexity?
- F. Thollard, P. Dupont, and C. de la Higuera. Probabilistic Dfa inference using Kullback-Leibler divergence and minimality. In *Proceedings of the 17th International Conference on Machine Learning*, pages 975-982. Morgan Kaufmann, San Francisco, CA, 2000

7 Conclusion and open questions

- A good candidate to learn NFA is DEES
- Never has been a challenge, so state of the art is still unclear
- Lots of room for improvement towards probabilistic transducers and probabilistic context-free grammars

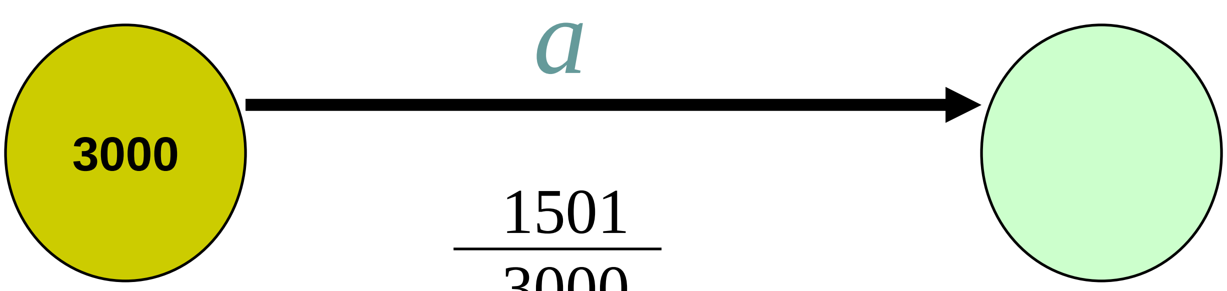
Appendix

Stern Brocot trees

Identification of probabilities

If we were able to discover the structure, how do we identify the probabilities?

- By estimation: the edge is used 1501 times out of 3000 passages through the state :

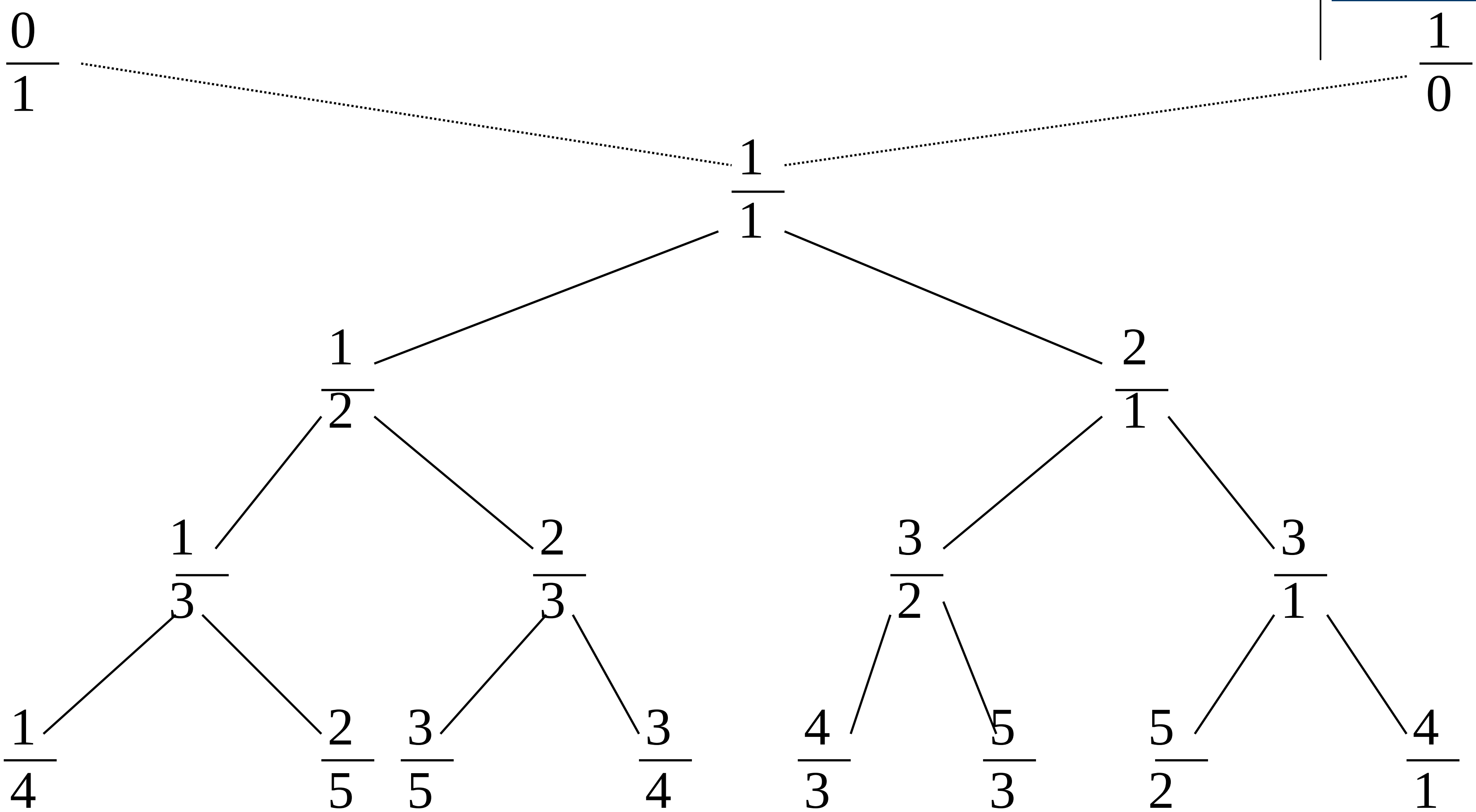


$$\frac{1501}{3000}$$

Stern-Brocot trees: (Stern 1858, Brocot 1860)

Can be constructed from two simple adjacent fractions by the «mean» operation

$$\frac{a}{b} \; m \; \frac{c}{d} = \frac{a+c}{b+d}$$


0/1
1/0
1/1
1/2
2/1
1/3
2/3
3/2
3/1
1/4
2/5
3/5
3/4
4/3
5/3
5/2
4/1

Idea:

- Instead of returning $c(x)/n$, search the Stern-Brocot tree to find a good simple approximation of this value.

Iterated Logarithm:

With probability 1, for a co-finite number of values of *n* we have:

$$\left| \frac{c(x)}{n} - \frac{a}{b} \right| < \sqrt{\frac{\lambda \log \log n}{n}}$$

$$\forall \lambda > 1$$