Hidden Markov Models

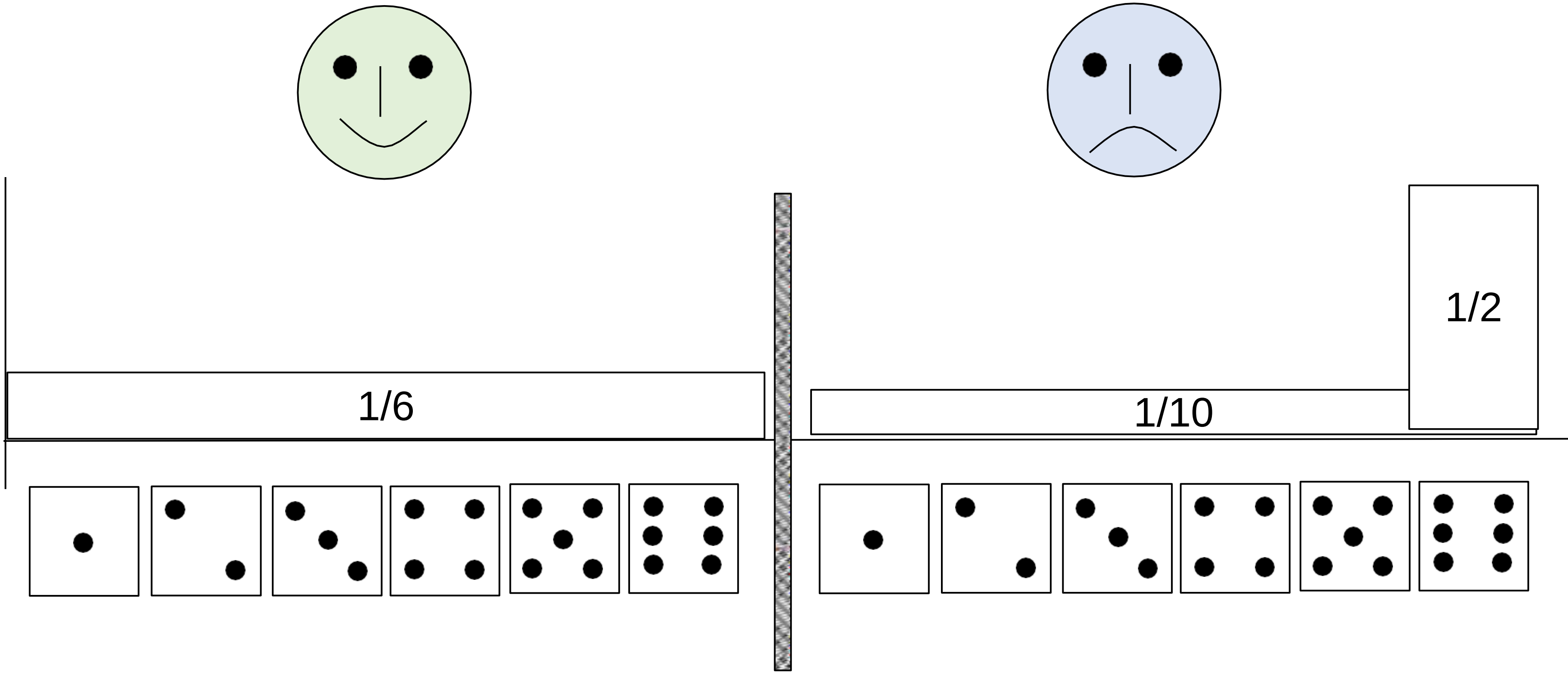
Lecture 7.3

by Marina Barsky

See sample code in [*casino.py*]

The honest and the dishonest casino

Choose L with P(L) = 0.01



We assume that: P(F) = 0.99 P(L) = 0.01

Prior probabilities – before we see any evidence (sequence)

Recap: the odds given evidence (sequence)

- P (W1|evidence) = P(evidence|W1)*P(W1)/P(evidence)
- P (W2|evidence) = P(evidence|W2)*P(W2)/P(evidence)

- To compare P (W1|evidence) vs P (W2|evidence) :

P (W1|evidence) / P (W2|evidence)

- Or to avoid underflow:

log [P (W1|evidence) / P (W2|evidence)]

- Log odds ratio = log [P(evidence|W1)*P(W1)/ P(evidence|W2)*P(W2)]
- If > 0 – first is more likely, if < 0 – second is more likely

Bayes theorem for Markov sequences

- Pick a die at random - and roll
- We get 3 consecutive sixes: '666'
- Is the die loaded? What is the probability?

- We want to know P(L|3 sixes)
- From Bayes theorem:

P(L|3 sixes) = P(3 sixes|L)*P(L)/P(3 sixes)

P(F|3 sixes) = P(3 sixes|F)*P(F)/P(3 sixes)

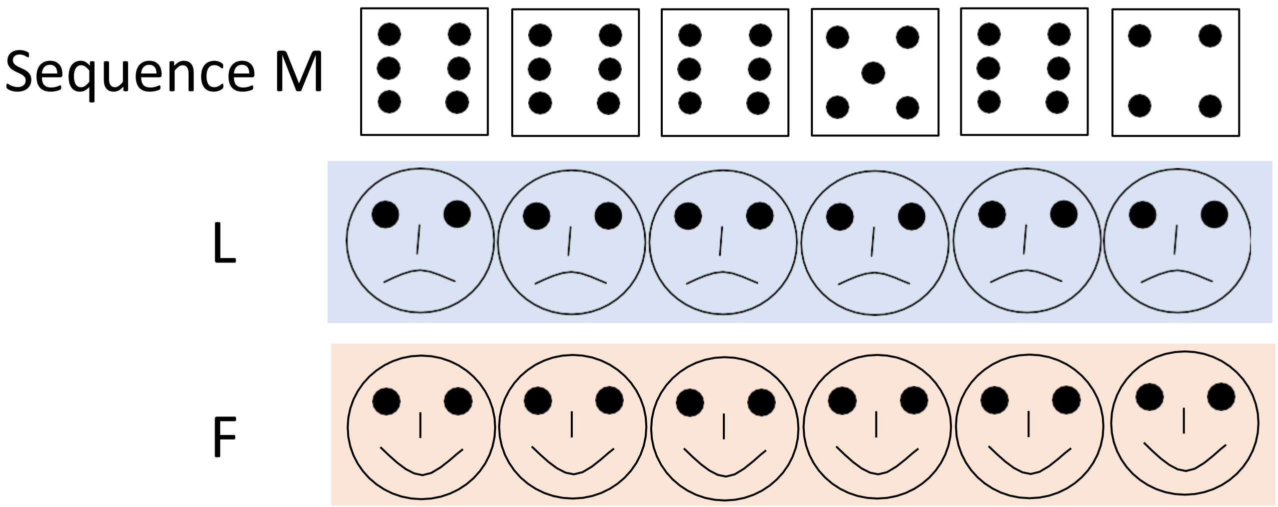
The sequence was generated either by fair or by loaded die

P(3 sixes) = P(3 sixes|F)*P(F) + P(3 sixes|L) *P(L) = 0.0058

- P (**L**|3 sixes) = (0.5*0.5*0.5 * 0.01) /0.0058 = **0.215**
- P(**F**|3 sixes) = (1/6)*(1/6)*(1/6)*0.99 / 0.0058 = **0.785**

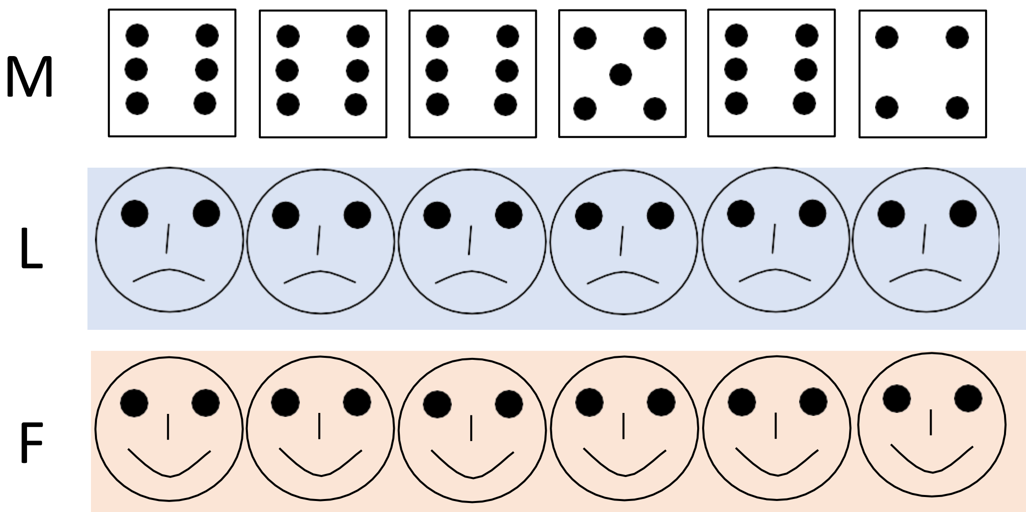
Not enough evidence to conclude that the die was Loaded

If two models **are <u>equally likely</u>**, we can use the conditional probabilities for discrimination



We can just compare P(M | L) and P(M | F)

We can use conditional probabilities for discrimination



OR

	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

P(M | L)=0.5*0.5*0.5*0.1*0.5*0.1=0.000625 = $6.25*10^{-4}$

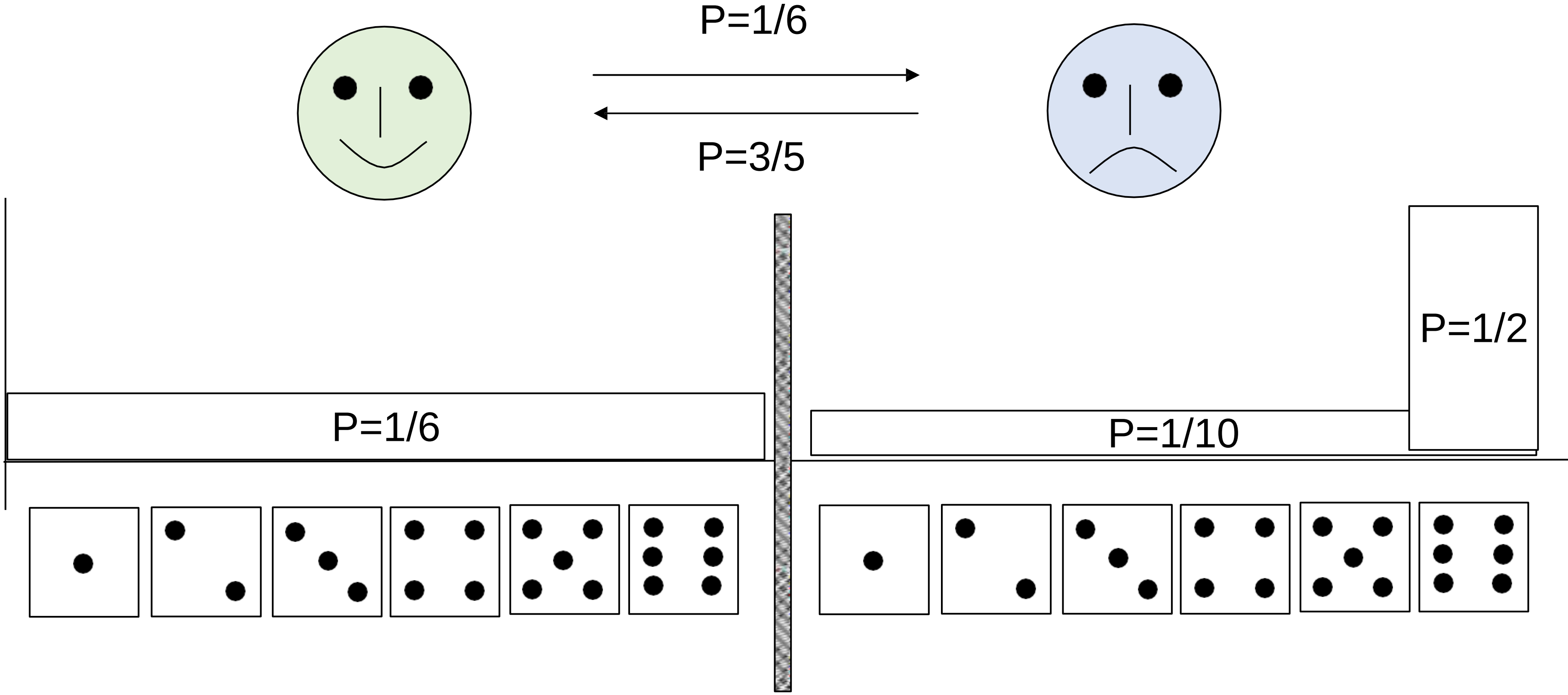
P(M | F)=0.17*0.17*0.17*0.17*0.17*0.17=0.000024 = $2.4 *10^{-5}$

How confident we are that this sequence was produced by a loaded die? P(M and model L)/ P(M and model F)=25.89

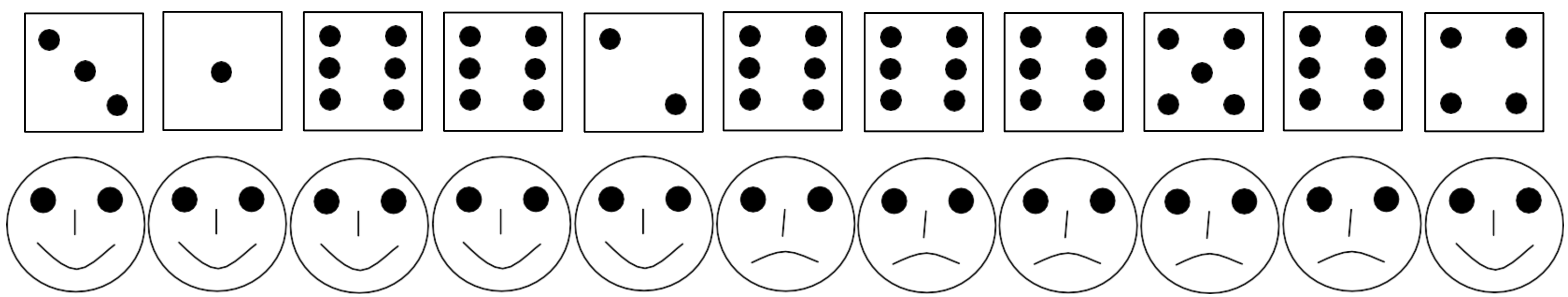
Or log [P(M I model L)/ P(M | F)]=1.4

Log-odds ratio

The occasionally dishonest casino



Sequence generated by a model of an occasionally dishonest casino



Markov chains: recap

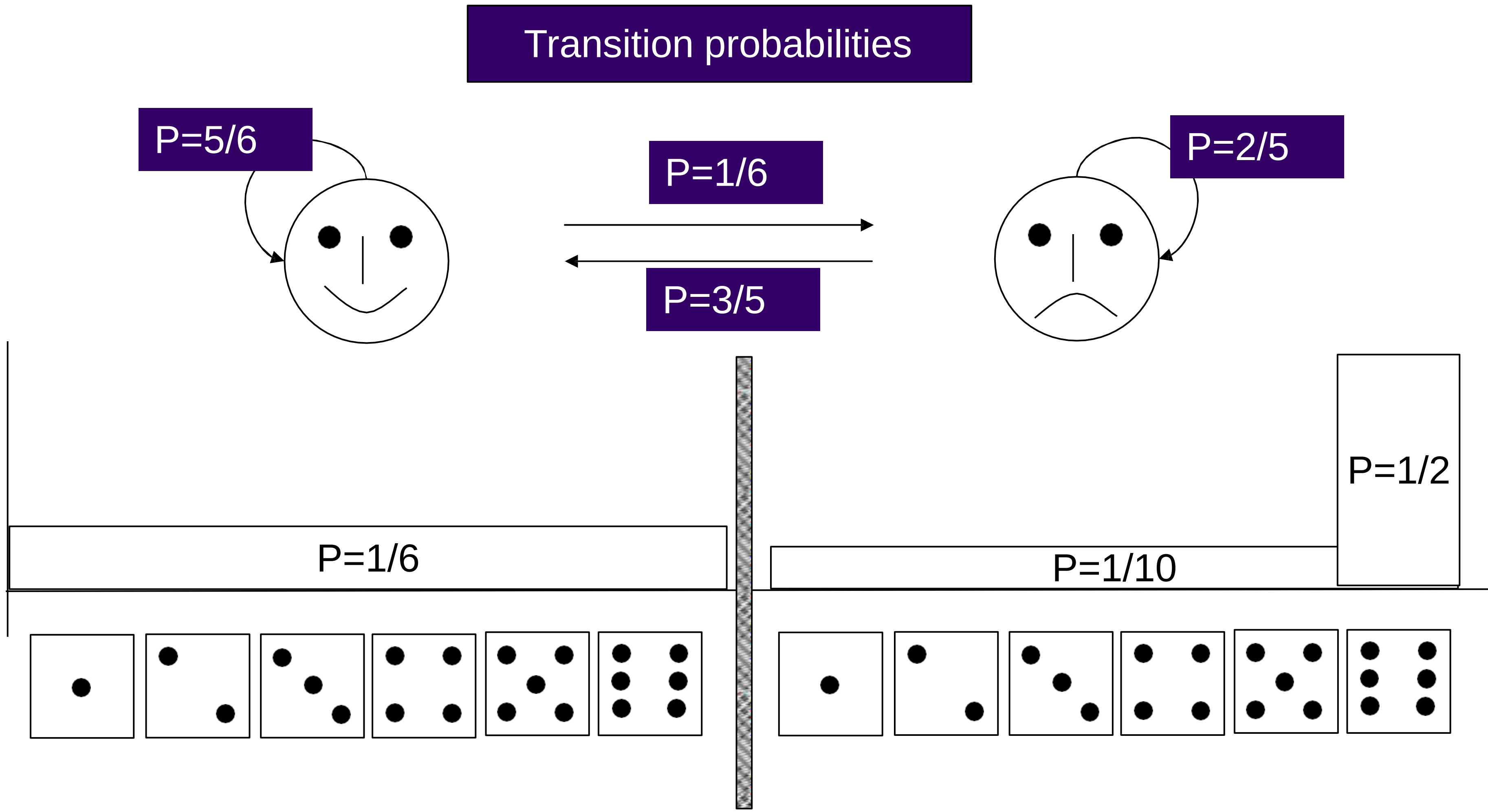
- The system can be in a finite number of states
- Transition from state to state is not predetermined, but rather is specified in terms of *probabilities*
- The transition probabilities depend only on the immediate history

- *The process of transitions from state to state* is called a ***Markov process*** or a ***Markov chain***

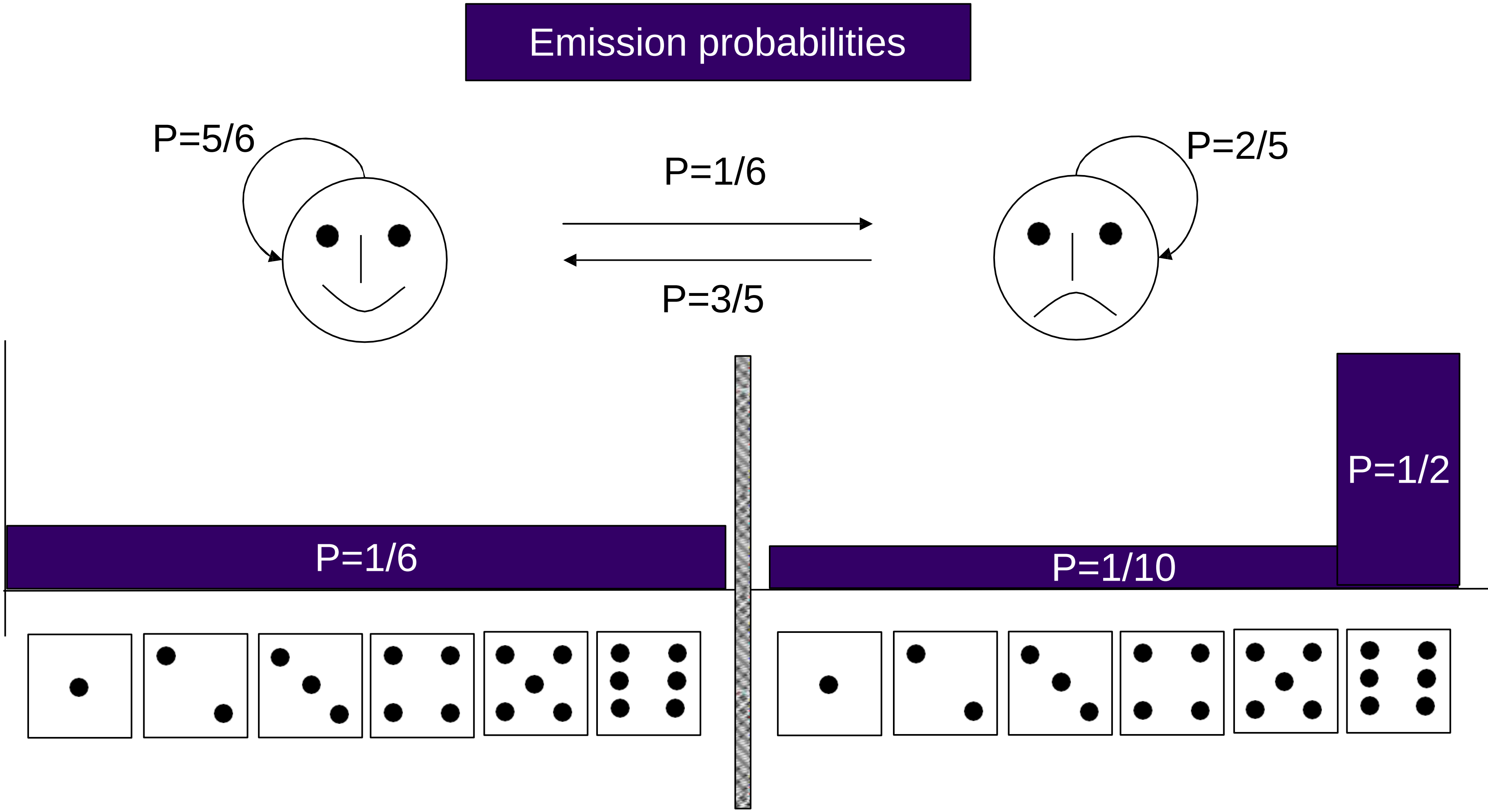
States can also behave probabilistically

- While in a particular state, system emits a symbol m_k from a finite alphabet with the probability $e_i(m_k)$, called *an emission probability* of symbol m_k in state W_i
- If we construct the schedule of observation times, and at each point in time record the symbols emitted by a system along with the state, we obtain 2 sequences:
  - the sequence of emitted symbols which is called *an observed sequence M*
  - the sequence of states π which is called a *path* through system states

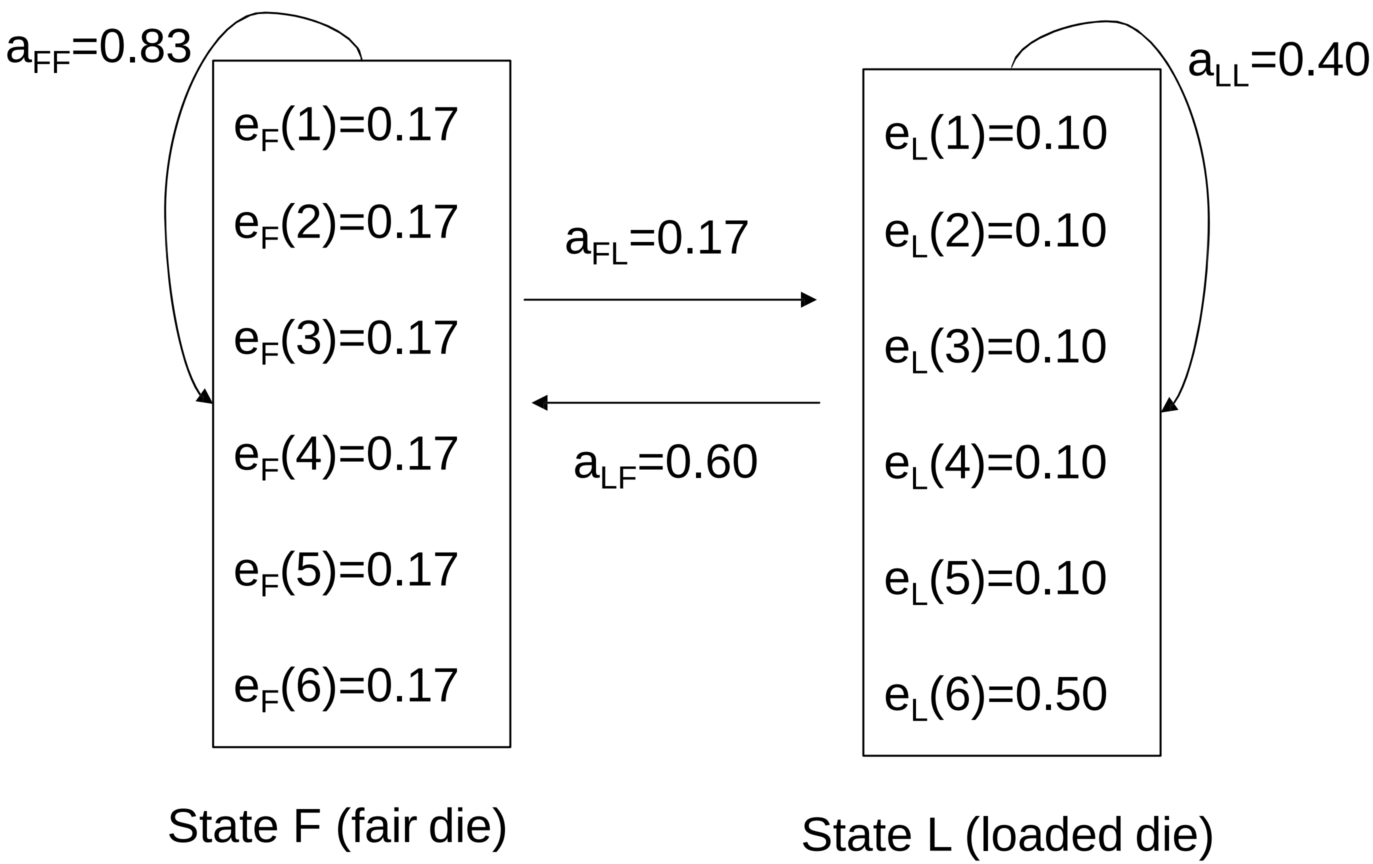
Terminology



Terminology



Transition and emission diagram



Tabular parameters

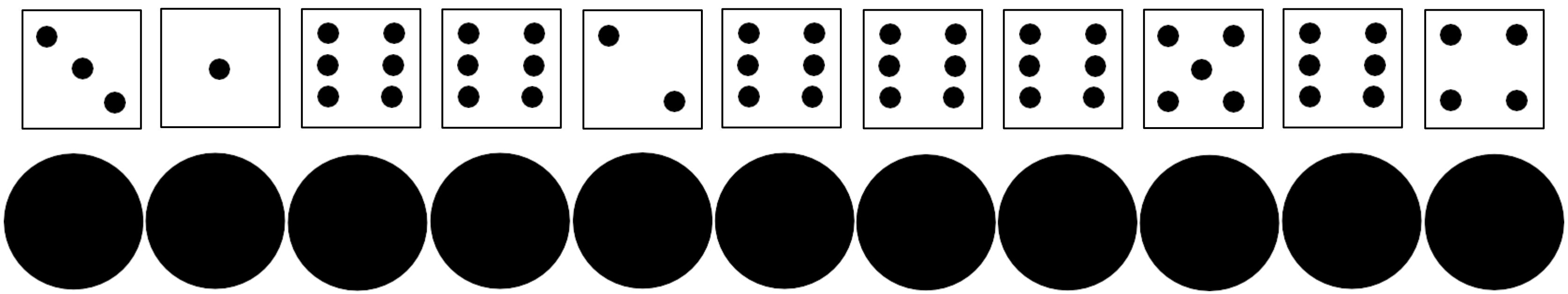
The state transition matrix

	F	L
F	0.83	0.17
L	0.60	0.40

Emission probabilities

	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

Hidden Markov Model (HMM)



States are unknown (hidden)

3 types of questions to HMM

1. Given a sequence of *N* observations, what is the probability of obtaining this sequence given a particular state path (**Sequence probability**)
2. Given a sequence of *N* observations, what is the most probable sequence of the underlying states (**Most probable *path***)
3. Given a sequence of N observations, what is the probability that the i-th observation was produced when the system was in state Wj

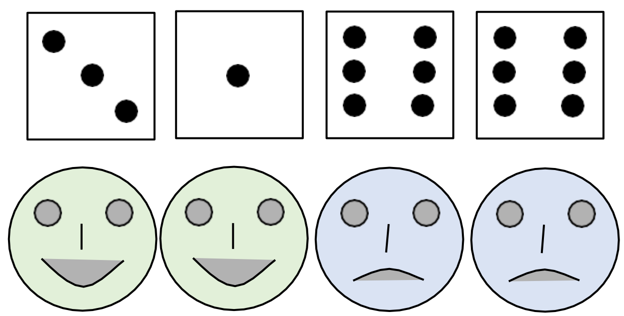
Question 1

Given a sequence and a path, what is the sequence probability?

- The probability $P(M|\pi)$ is the *conditional probability* that sequence M was generated while system was moving from state to state according to π

The probability that the sequence was generated following a path π

- Pick a path π
- Calculate a joint probability of π and M



A suggested path

	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	F	L
F	0.83	0.17
L	0.60	0.40

P(M and π)=0.17 * 0.83 * 0.17 * 0.17 * 0.50 * 0.60 * 0.50=0.0006

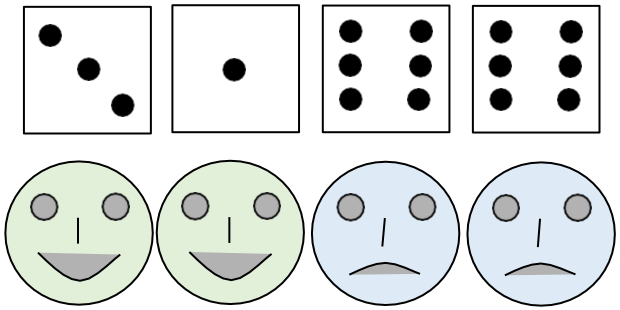
- Note that this is not P(π | M)

The probability that the sequence was generated following a path π **when π is unknown (hidden)**

- Pick a path π
- Calculate a joint probability of π and M

	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

A suggested path



	F	L
F	0.83	0.17
L	0.60	0.40

P(M and π)=0.17 * 0.83 * 0.17 * 0.17 * 0.50 * 0.60 * 0.50=0.0006

- Repeat **for each possible path** and choose a path which maximizes P(π and M).
- Total 2^N calculations (for 2 states and sequence of length N)

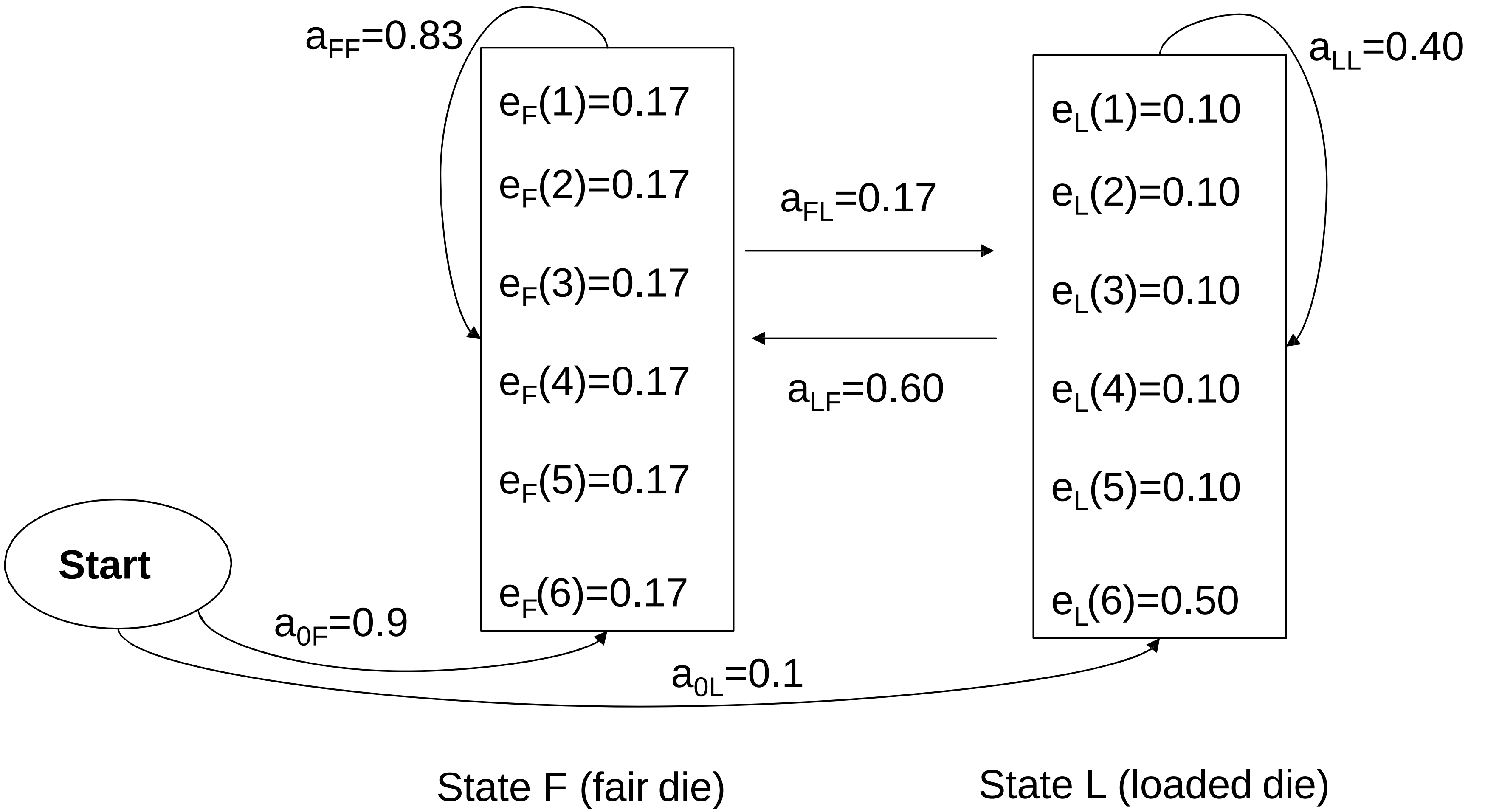
Question 2

Given only a sequence of observations, what is the most probable path of states?

Viterbi algorithm: dynamic programming

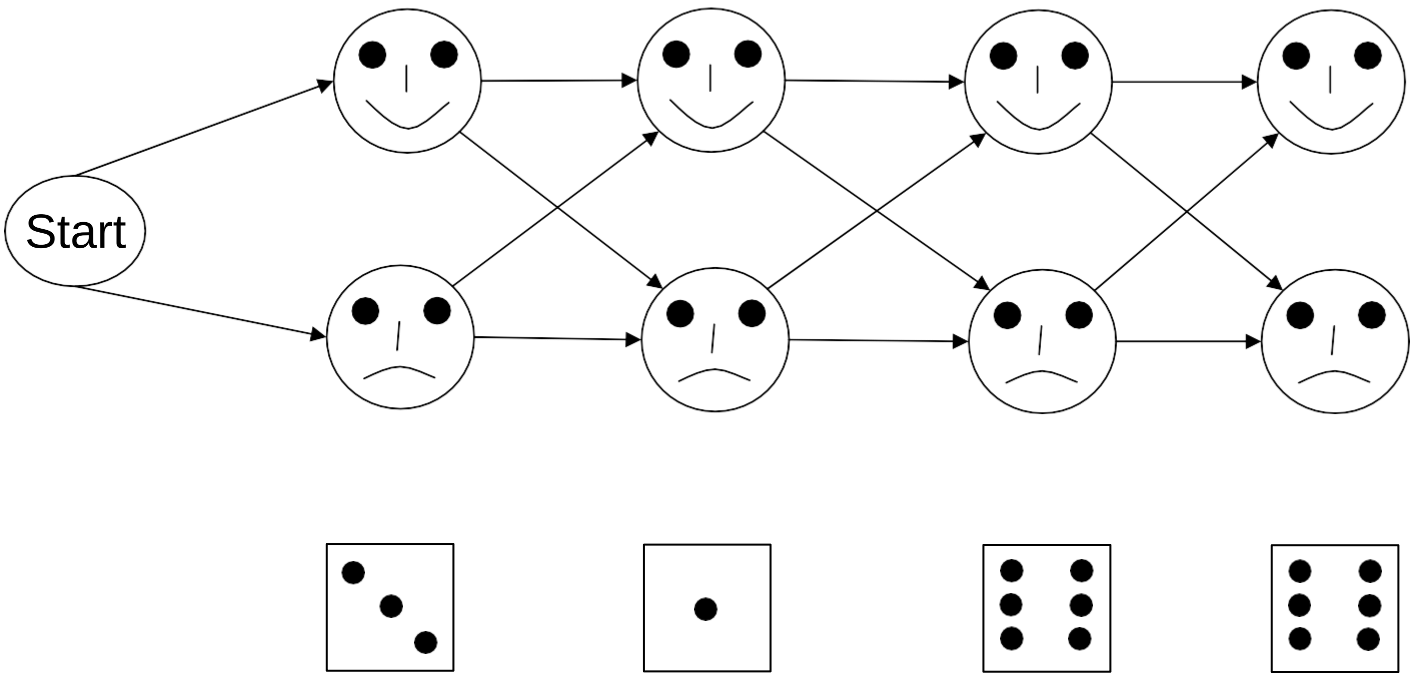
Dynamic programming. Initialization – the probability of choosing a die for the first time

- Add to the system a **start state** and parameters – the probabilities of choosing a fair or a loaded die in the beginning of a game



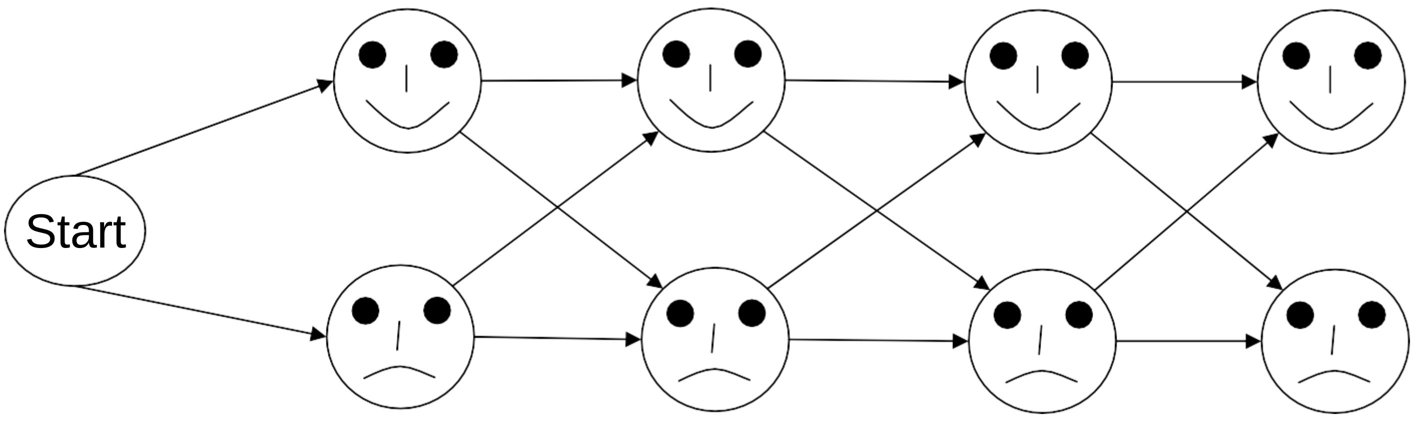
Dynamic programming. Initialization

The graph of a process.



$P(\pi_{F,1})=a_{0F}*e_F(M[1])$
$P(\pi_{L,1})=a_{0L}*e_L(M[1])$

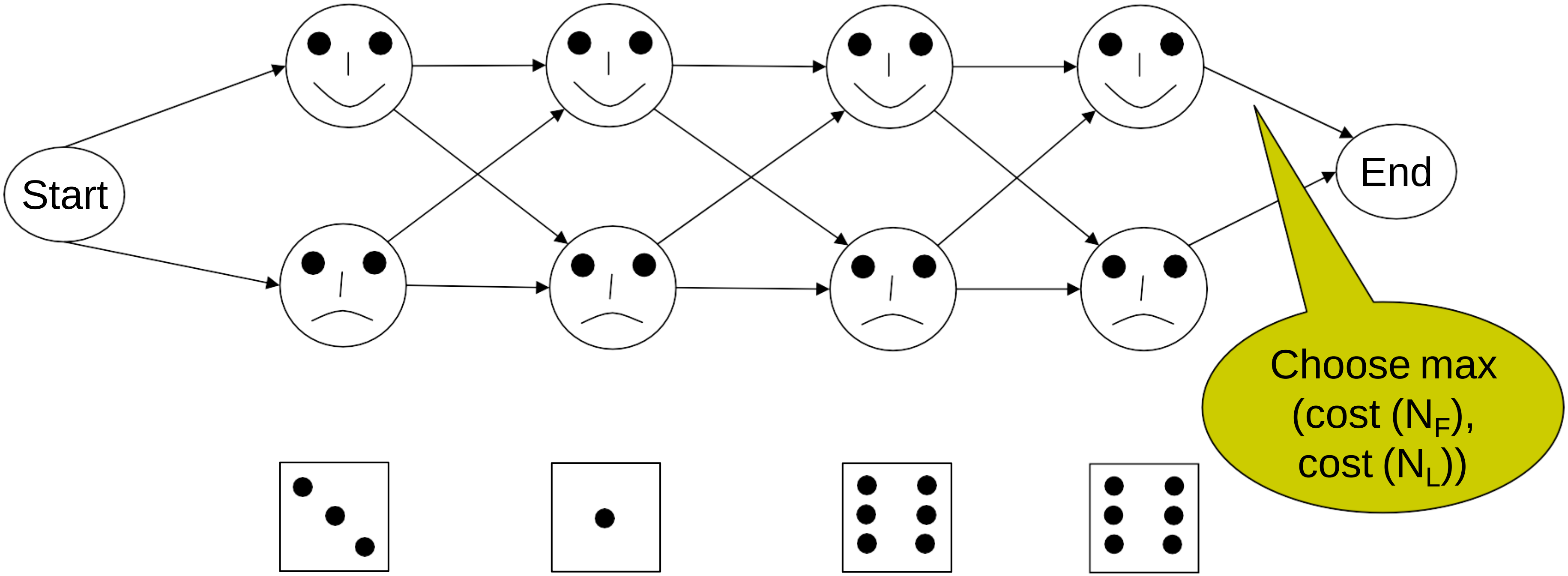
Dynamic programming.
Recurrence relation



We are looking for a path which maximizes the probability of sequence M

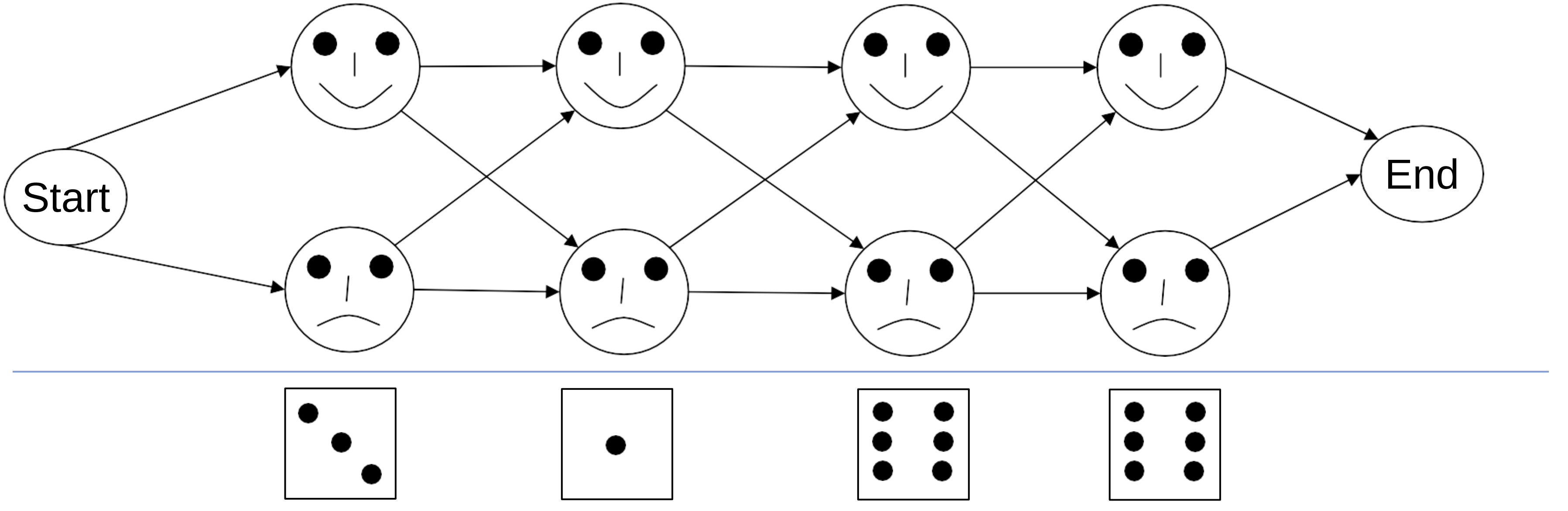
Dynamic programming.
Recurrence relation

If we know the best paths ending at states L and F in position 4, we can choose max between them and terminate the program



Dynamic programming. Recurrence relation

This can be repeated for each combination of a position in a sequence of observations and one of 2 states



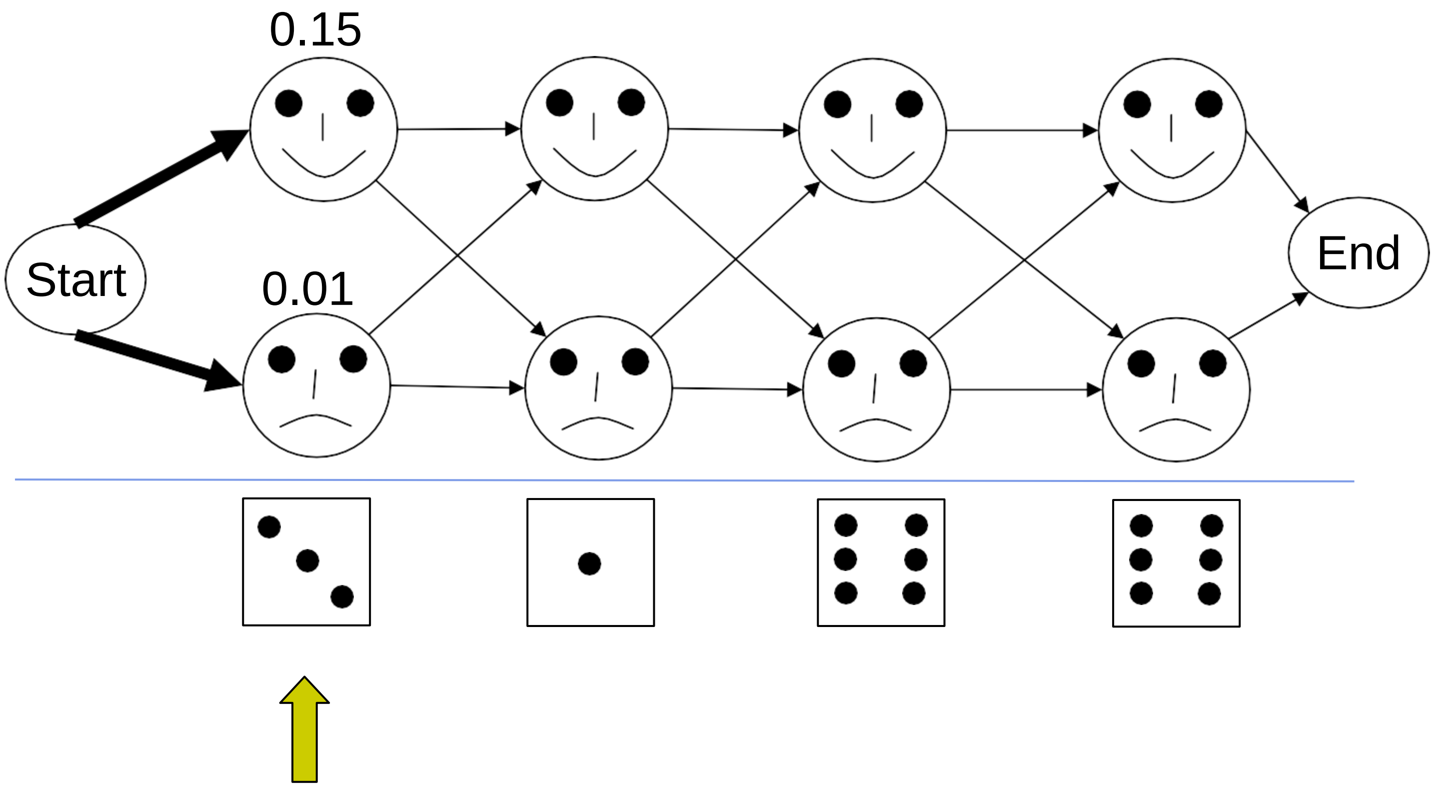
$P(\pi_{F,i+1}) = \max \{P(\pi_{F,i})*a_{FF},\ P(\pi_{L,i})*a_{LF}\} * e_F(M[i+1])$

$P(\pi_{L,i+1}) = \max \{P(\pi_{L,i})*a_{LL},\ P(\pi_{F,i})*a_{FL}\} * e_L(M[i+1])$

$P(\pi^*) = \max \{P(\pi_{F,N}),\ P(\pi_{L,N})\}$

Note: the probabilities are ***multiplied***, not added up

Viterbi algorithm. Demo 1

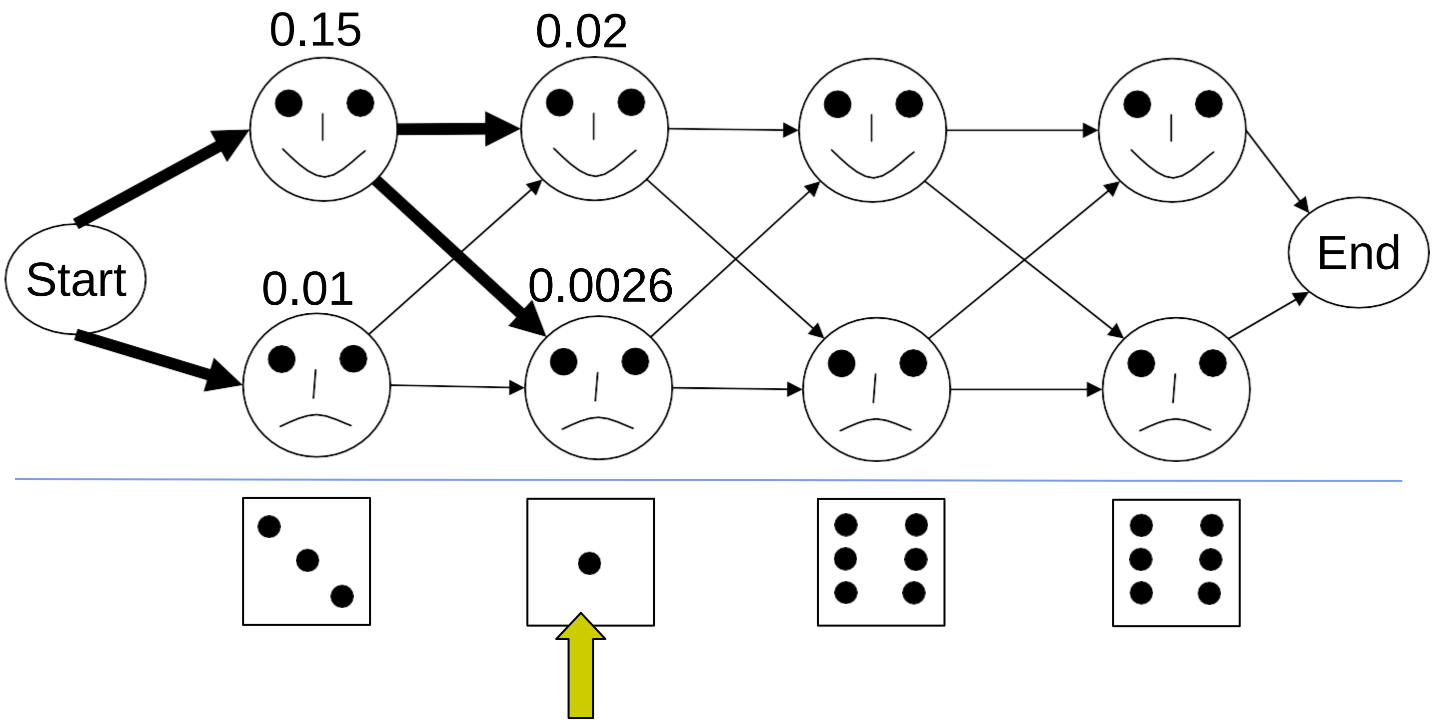


	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	F	L
F	0.83	0.17
L	0.60	0.40
0	0.90	0.10

We have reached position i=1 with the probability 0.9*0.17 of going to the F state and emitting 3, and with probability 0.1*0.10 of going to the L-state and emitting 3. There are no other possibilities

Viterbi algorithm. Demo 2



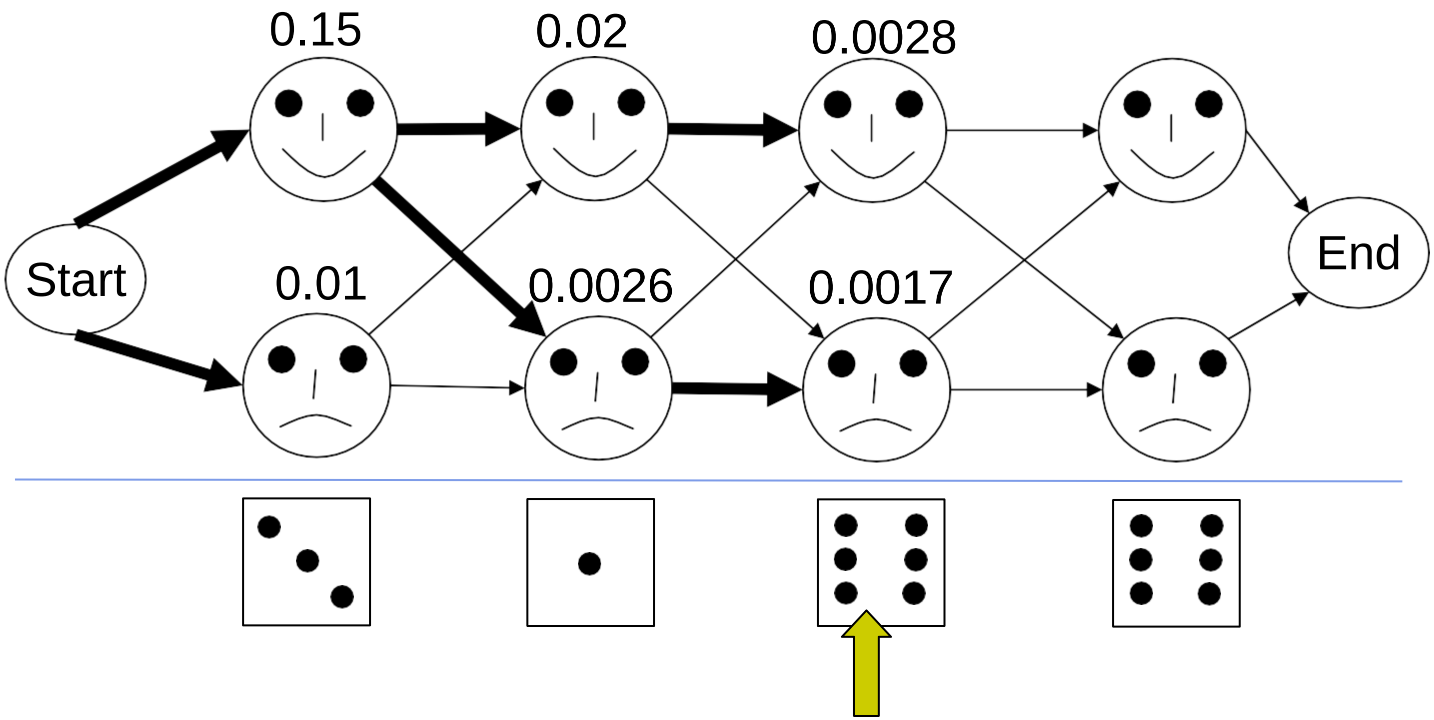
	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	F	L
F	0.83	0.17
L	0.60	0.40
0	0.90	0.10

We can reach position i=2 (F-state) with the probability 0.15*0.83*0.17 or with probability 0.01*0.6*0.10. We chose the max between these two: 0.15*0.83*0.17=0.002

The L-state in position i=2 can be reached with probability 0.01*0.40*0.10 or 0.15*0.17*0.10=0.0026. The second is larger so we choose it.

Viterbi algorithm. Demo 3



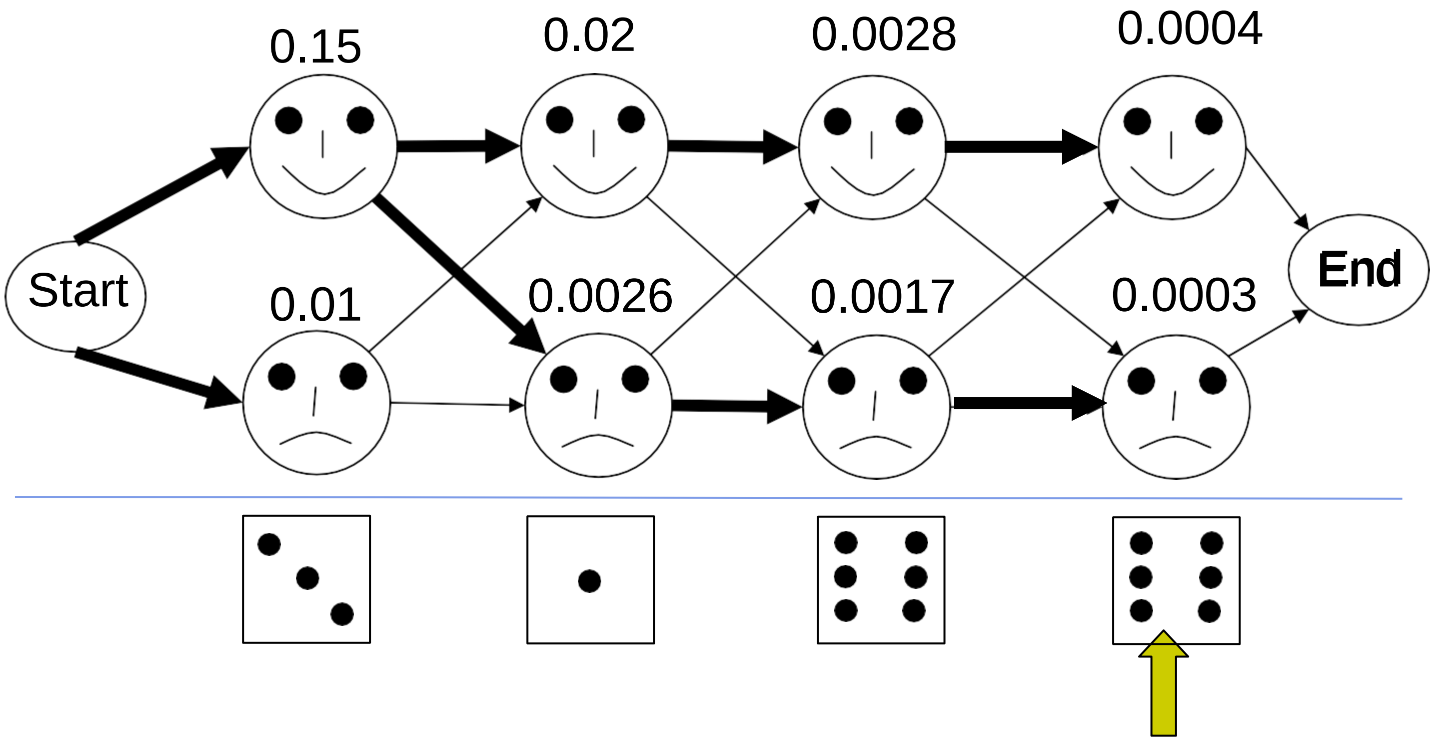
	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	F	L
F	0.83	0.17
L	0.60	0.40
0	0.90	0.10

We can reach position i=3 (F-state) with the probability 0.02*0.83*0.17=0.0028 or with probability 0.0026*0.4*0.17=0.00018. We chose the max between these two: 0.02*0.83*0.17=0.0028

The L-state in position i=3 can be reached with probability 0.02*0.17*0.50=0.0017 or 0. 0026*0.4*0.5=0.0017. We chose the second - arbitrarily

Viterbi algorithm. Demo 4



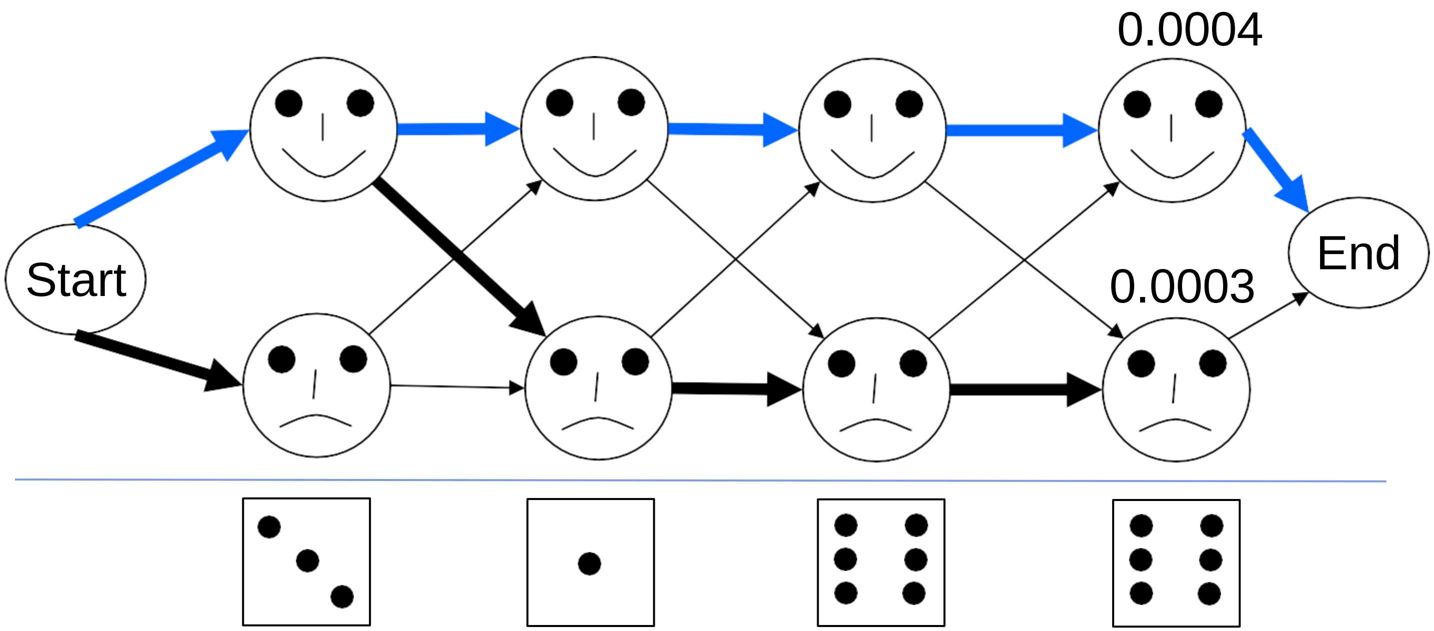
	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	F	L
F	0.83	0.17
L	0.60	0.40
0	0.90	0.10

We can reach position i=4 (F-state) with the probability 0.0028*0.83*0.17=0.0004 or with probability 0.0017*0.6*0.17=0.00017. We chose the max between these two: 0.0028*0.83*0.17=0.0004

The L-state in position i=4 can be reached with probability 0.0017*0.40*0.50=0.00034 or 0.0028*0.17*0.5 =0.00024. We chose the max: 0.0017*0.40*0.50=0.00034

Viterbi algorithm. Demo - end



	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	F	L
F	0.83	0.17
L	0.60	0.40
0	0.90	0.10

Choose max: 0.0004. So, the most probable sequence of states:

FFFF

Evidently, it is not enough to have 2 sixes in a row in order to be able to spot the loaded die.

Viterbi algorithm. Log-values

$P(\pi_{F,1})=a_{0F}*e_F(M[1])$ $P(\pi_{L,1})= a_{0L}*e_L(M[1])$

$P(\pi_{F,i+1})=\max\{ P(\pi_{F,i})*a_{FF}, P(\pi_{L,I})*a_{LF} \}* e_F(M[i+1])$

$P(\pi_{L,i+1})=\max\{P(\pi_{L,i})*a_{LL}, P(\pi_{F,i})*a_{FL}\} *e_L (M[i+1])$

$P(\pi^*)=\max\{P(\pi_{F,N}), P(\pi_{L,N})\}$

In order to avoid the underflow errors, in practice *log* is used instead of the actual probabilities

$P(\pi_{F,1})=\log a_{0F}+ \log e_F(M[1])$ $P(\pi_{L,1})= \log a_{0L}+ \log e_L(M[1])$

$P(\pi_{F,i+1})=\max\{P(\pi_{F,i})+ \log a_{FF}, P(\pi_{L,I})+ \log a_{LF} \}+ \log e_F(M[i+1])$

$P(\pi_{L,i+1})=\max\{P(\pi_{L,i})+ \log a_{LL}, P(\pi_{F,i})+ \log a_{FL}\} + \log e_L (M[i+1])$

$P(\pi^*)=\max\{P(\pi_{F,N}), P(\pi_{L,N})\}$

How good is the prediction

Overall, an underlying hidden pathway explains the given sequence well – the path explanation obtained with Viterbi is good

We can now answer these questions:

- What is the probability that a given sequence of observations came from a particular HMM
- Where in the sequence the model has probably changed

Activity. Discrimination by probability

- Markov models for the honest and for the dishonest casino are presented below:

e(Heads)=1/2	e(Heads)=3/4
e(Tails)=1/2	e(Tails)=1/4
Fair coin	Biased coin

Given that it is equally probable to choose F or L, find out which coin has most probably produced the following sequence of observations:

HHHTTHT

When the heads point to the biased coin?

- For sequence M of length N with *k* heads:

$P(M \mid \text{fair coin}) = \Pi_n(1/2) * P(F)/P(M) \sim 1/2^N$

$P(M \mid \text{biased coin}) = \Pi_k(3/4) * \Pi_{N-k}(1/4) * P(B)/P(M) \sim 3^k/4^k * 1/4^{N-k}$

- For this simple example, we can compute how many heads out of N are needed to conclude that the coin is biased:
- when P(M and fair coin) < P (M and biased coin) ?

$1/2^N < 3^k/4^N$

$1 < 3^k/2^N$

$2^N < 3^k$

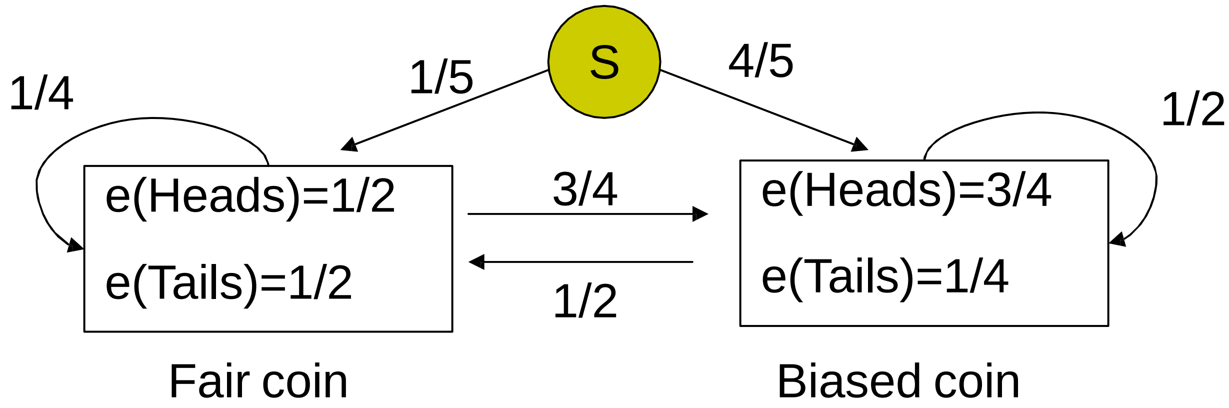
$N\log 2 < k\log 3$

$k > (\log 2/\log 3) * N$

$k > 0.63\ N$

Activity

- Using the Viterbi algorithm, find the most probable path of states for the following sequence given the following HMM.



Observed sequence: HTTHHH

Building a Hidden Markov Model

- 2 parts:
  - Model topology: what states there are and how are they connected
  - The assignment of parameter values: the transition and emission probabilities

Parameter estimation

- We are given a set of training sequences
- 2 cases:
  - When the states in the training sequences are known

    $a_{from,to}$=$count_{from,to}$/$\Sigma_x count_{from,x}$

    $e_{state\ i}$(symbol j)=$count_{state\ i}$(symbol j)/Σ_y(symbol y|$state_i$)

  - When the states are unknown
    - Viterbi training

Parameter estimation when the states are known - example

X	1	2	6	6	1	1	2
π	F	L	F	F	L	L	L

$e_F(3)=0$?

To avoid this, use *pseudocounts*

$e_F(1)=(1+1)/(3+6)$, 1 is a pseudocount, 6 is the number of different symbols

eF(1)=2/9

$e_F(2)=1/(3+6)=1/9$

$e_F(3)=1/(3+6)=1/9$

$e_F(4)=1/(3+6)=1/9$

$e_F(5)=1/(3+6)=1/9$

$e_F(6)=(2+1)/(3+6)=3/9$

$a_{F,L}=2/3$

$a_{F,F}=1/3$

$a_{L,F}=1/3$

$a_{L,L}=2/3$

Or with pseudocounts

$a_{F,L}=(2+1)/(3+2)=3/5$

$a_{F,F}=(1+1)/(3+2)=2/5$

$a_{L,F}=(1+1)/(3+2)=2/5$

$a_{L,L}=(2+1)/(3+2)=3/5$

Viterbi training for parameter estimation

- Pick a set of random parameters
- Repeat
  - Find the most probable path of states according to this set of parameters
  - This path partitions the sequences into partitions according to the states
  - Calculate new set of parameters, now from the known states
- Until the path does not change anymore

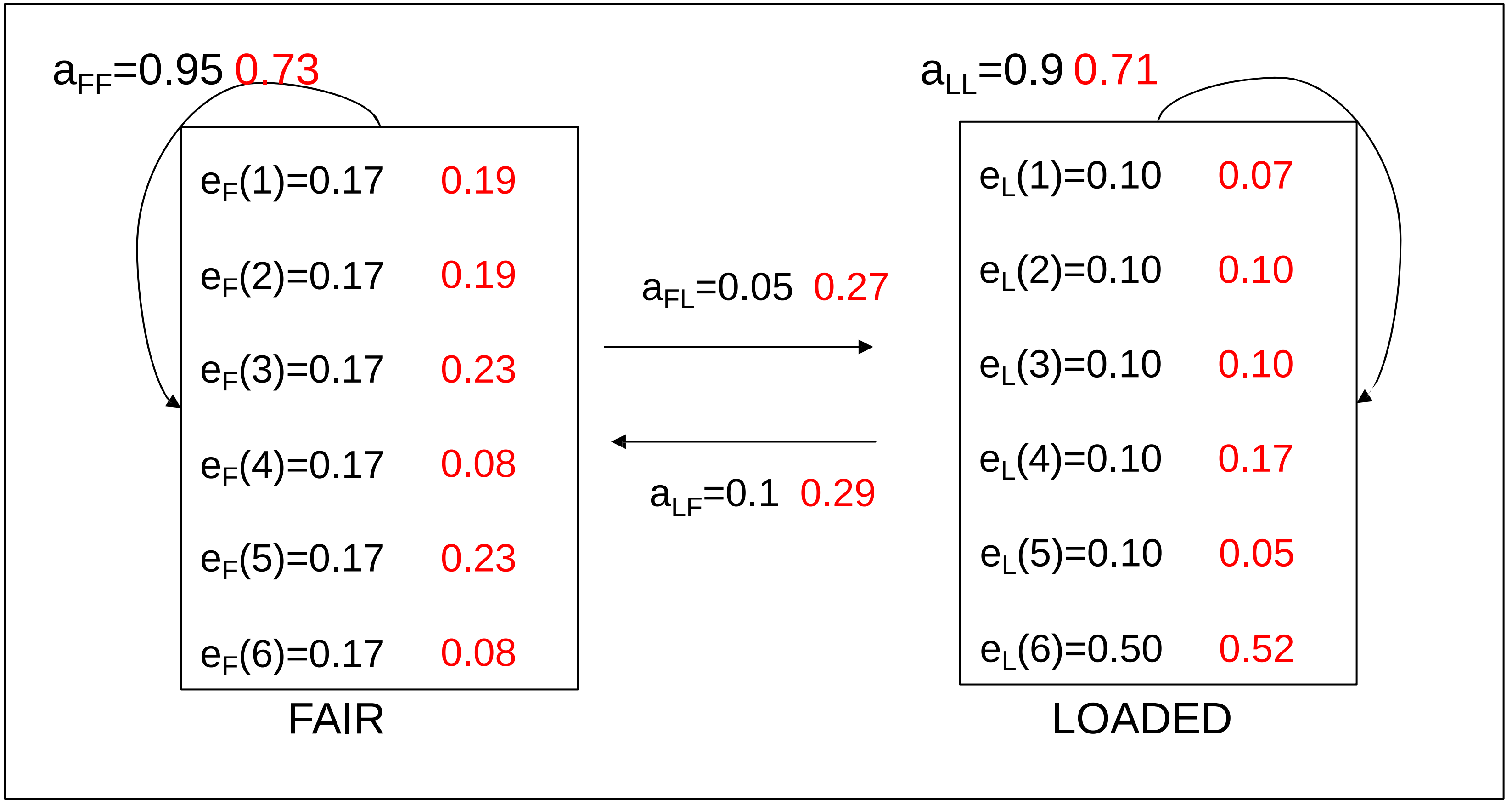
Viterbi training

- The assignment of paths is a discrete process, thus the algorithm converges precisely
- When there is no path change, the parameters will not change either, because they are determined completely by the paths
- The algorithm maximizes the probability

  $P(\text{observed data} \mid \Theta, \pi^*)$

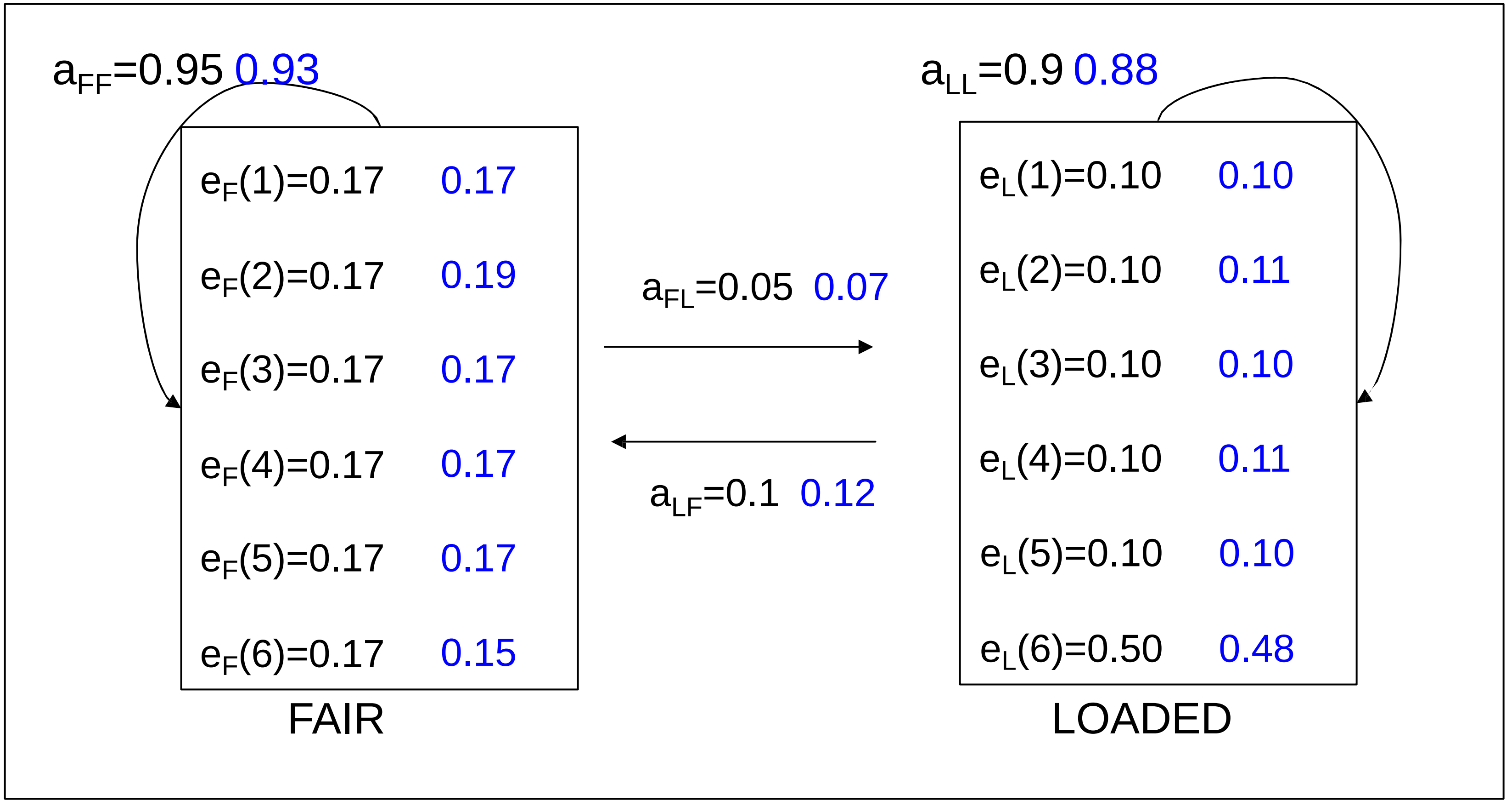
  and not $P(\text{observed data} \mid \Theta)$ which we ideally want

Parameter estimation – illustration 1



The parameters estimated for 300 random rolls and an iterative process started from randomly selected parameters

Parameter estimation – illustration 2



The parameters estimated for 30 000 random rolls and an iterative process started from randomly selected parameters