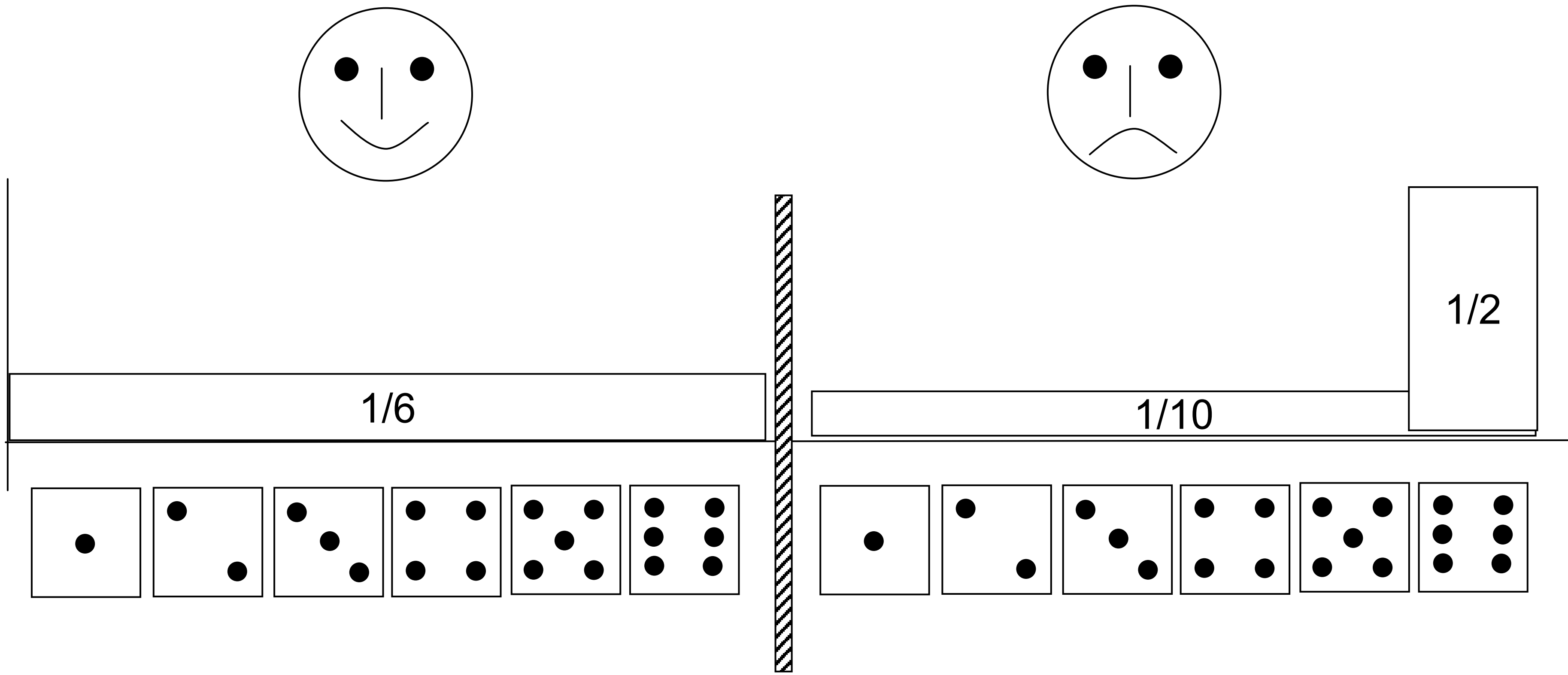
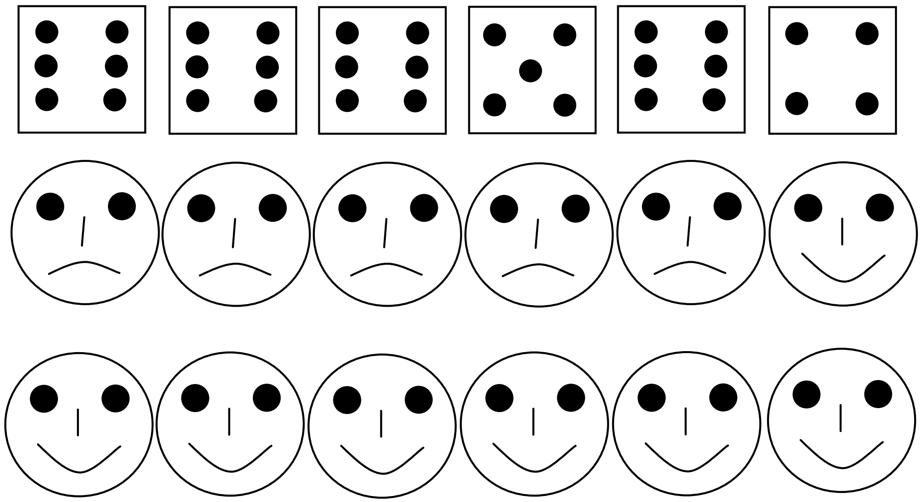
Markov Models

Lecture 10

The honest and the dishonest casino

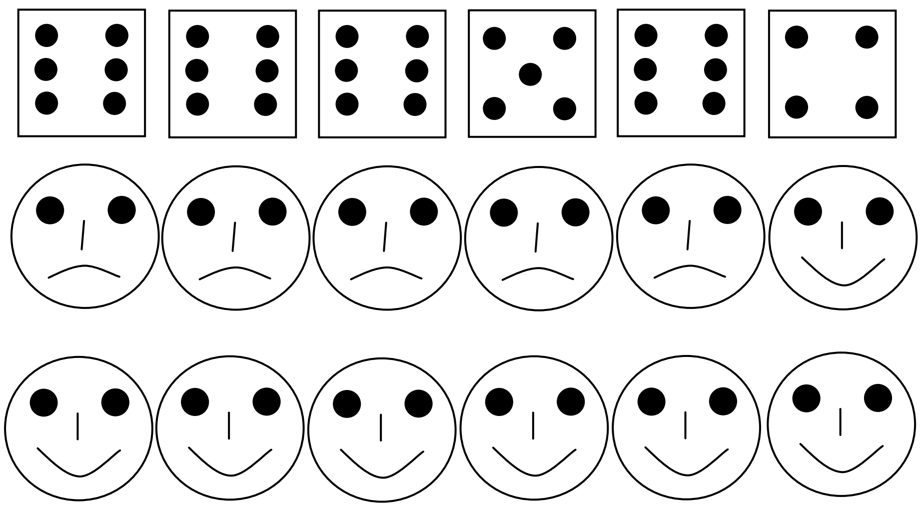


We can use the conditional probabilities for discrimination



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

We can use the 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 and model L)=0.5*0.5*0.5*0.1*0.5*0.1=0.000625

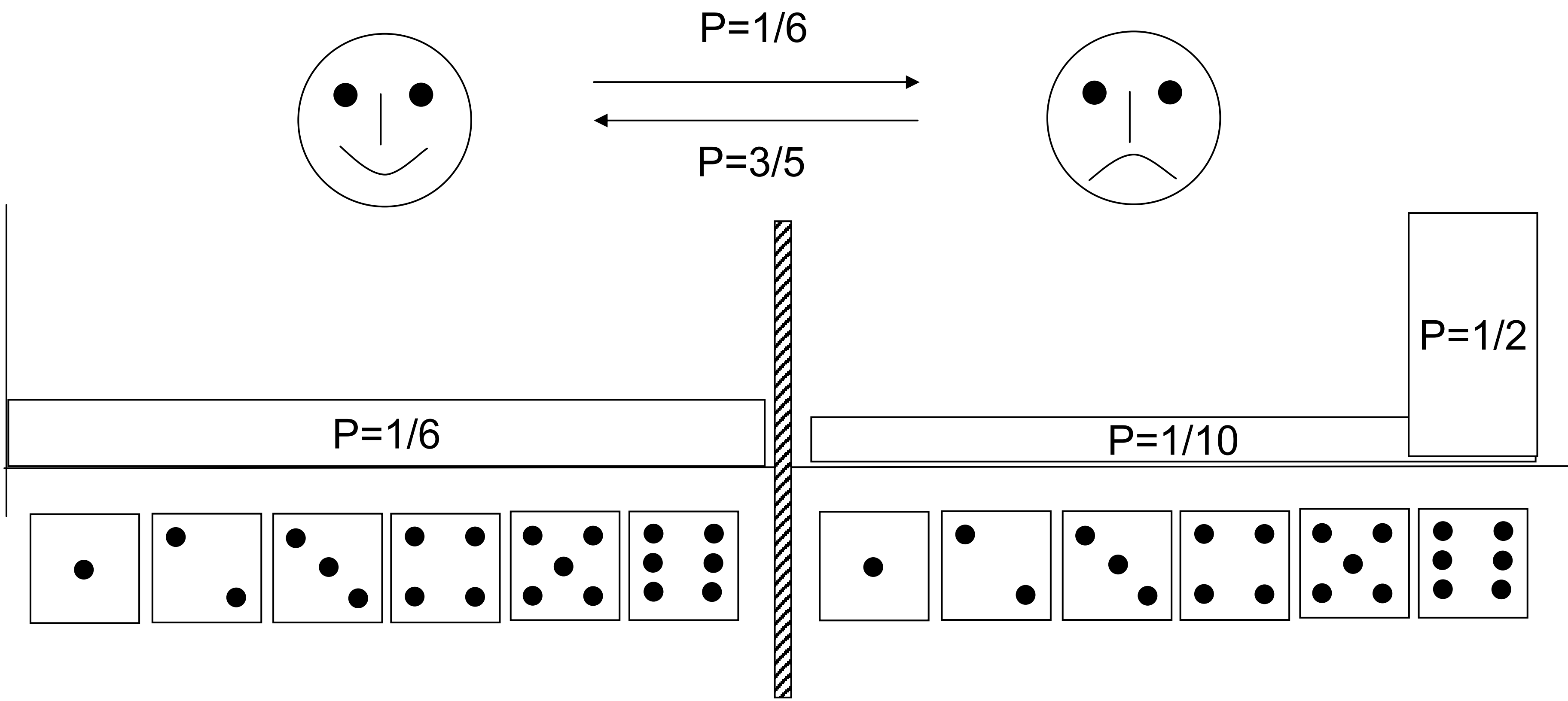
P(M and model F)=0.17*0.17*0.17*0.17*0.17*0.17=0.000024

How confident we are that this sequence was produced by a loaded die?

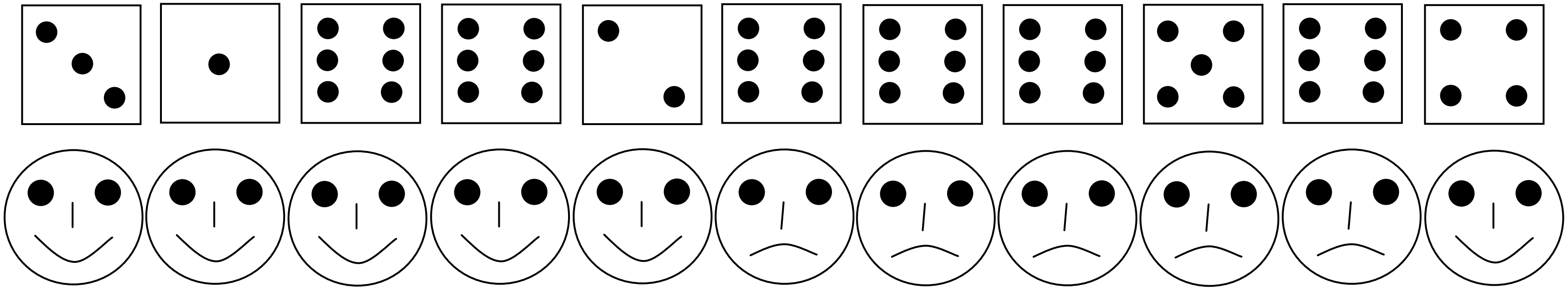
P(M and model L)/ P(M and model F)=25.89

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

The occasionally dishonest casino



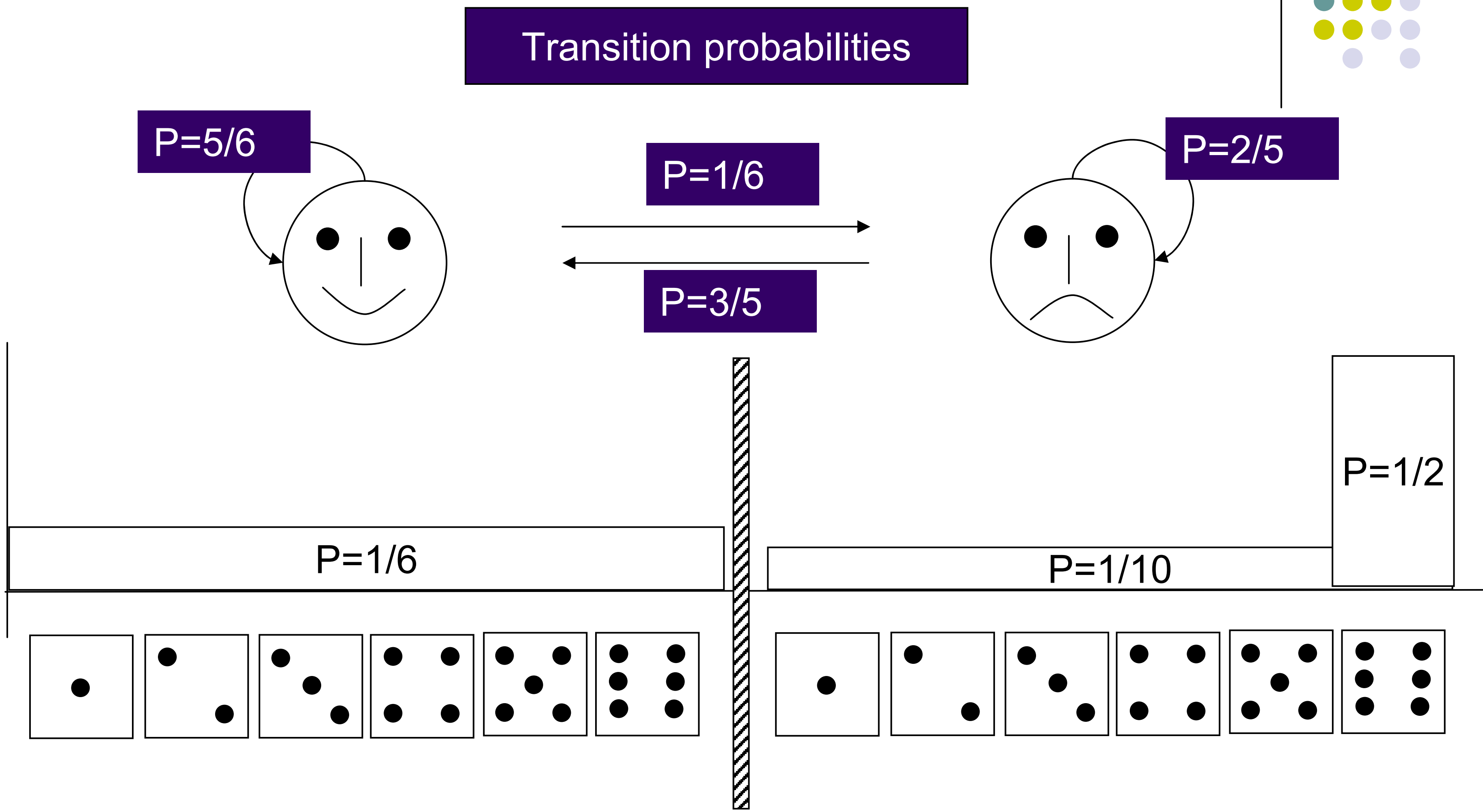
Sequence generated by a model of an occasionally dishonest casino



Markov chains

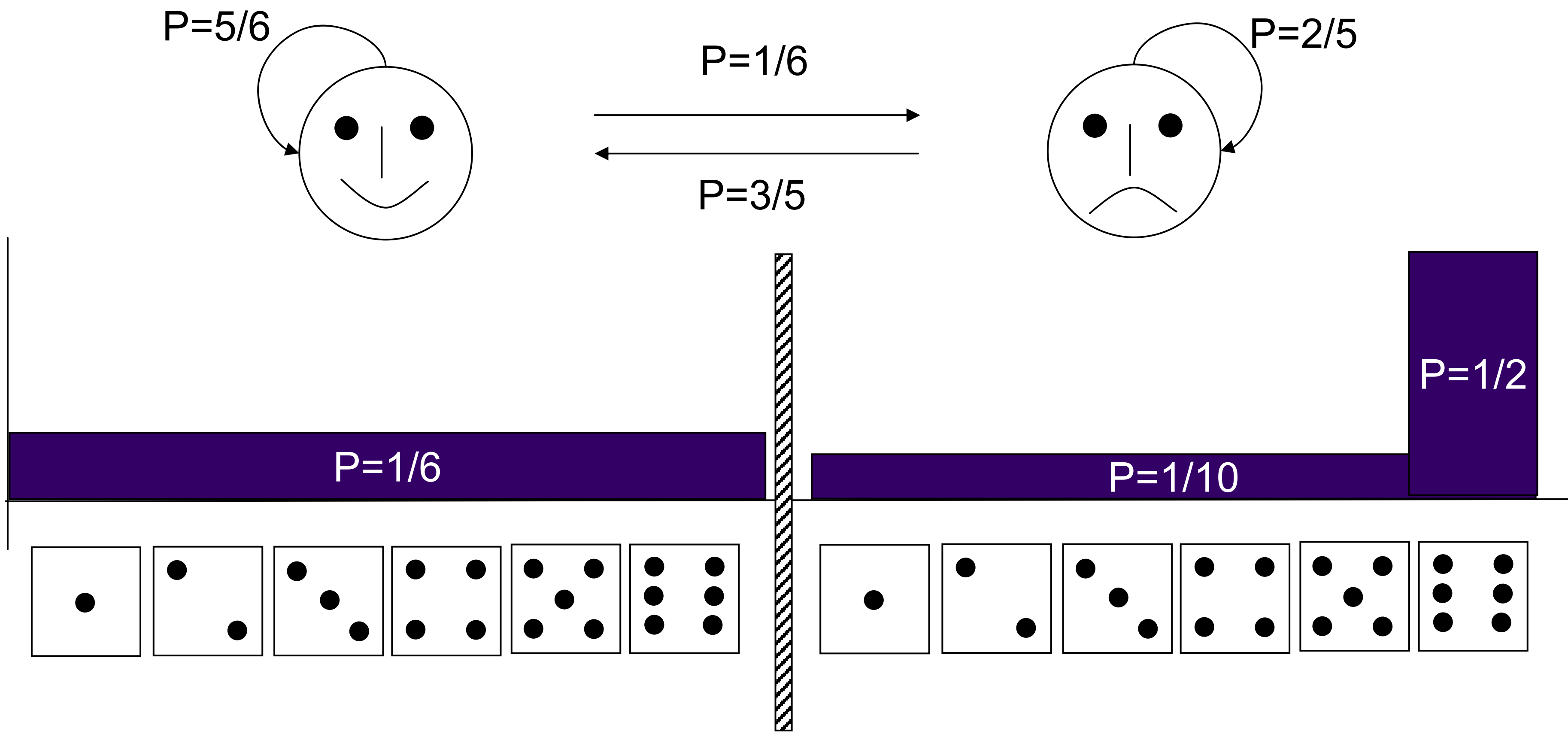
- A general model of a system which moves from state to state with some probability a_{ij}, called a *transition probability*
- While in a particular state, system emits a symbol m_k from a finite alphabet with the probability $e_i(m_k)$, called *an emission probability* of symbol m_k in state W_i
- 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, and the sequence of states which is called a *path* through system states

Markov chain terminology

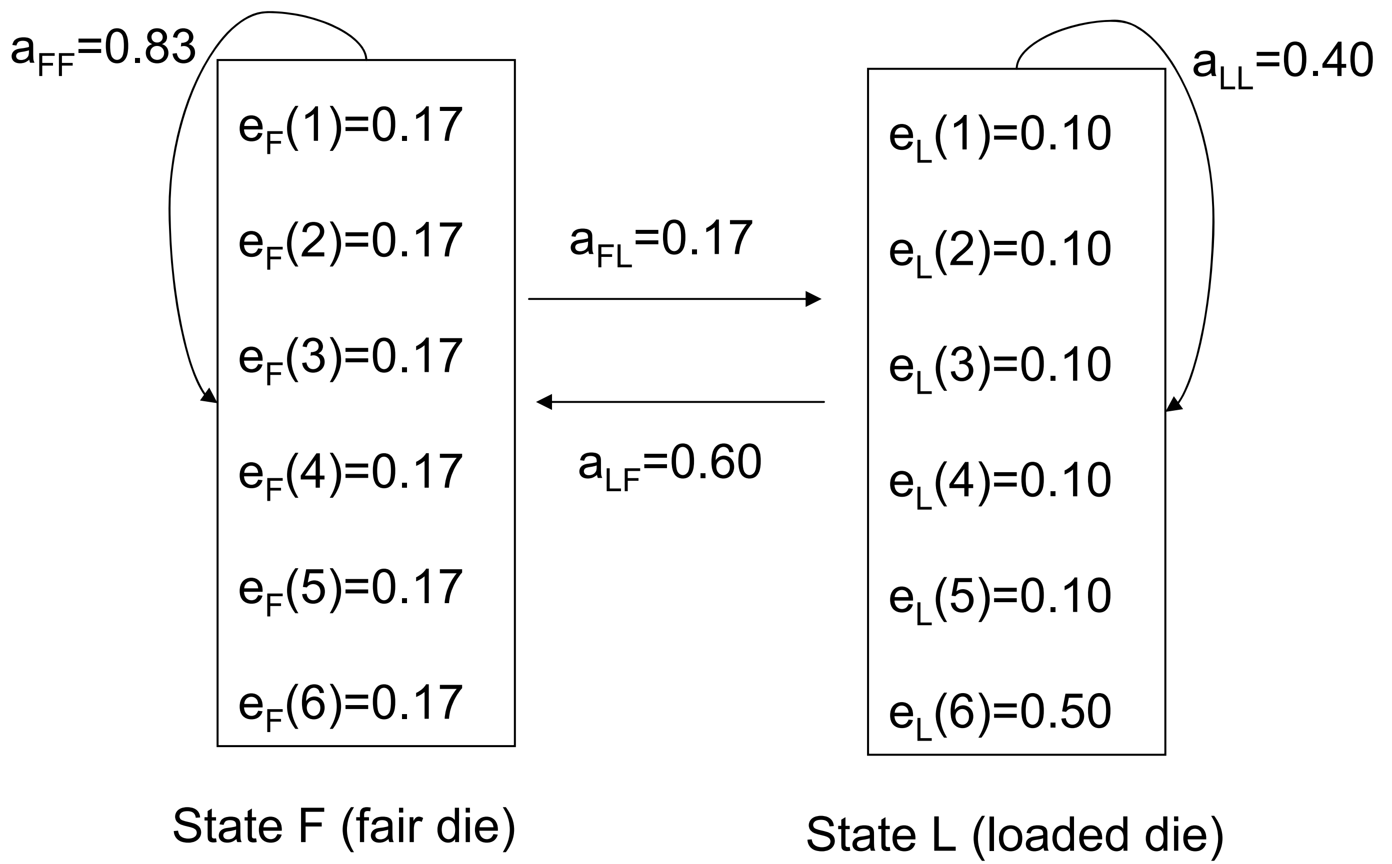


Markov chain terminology

Emission probabilities



Markov model diagram



Markov model parameters

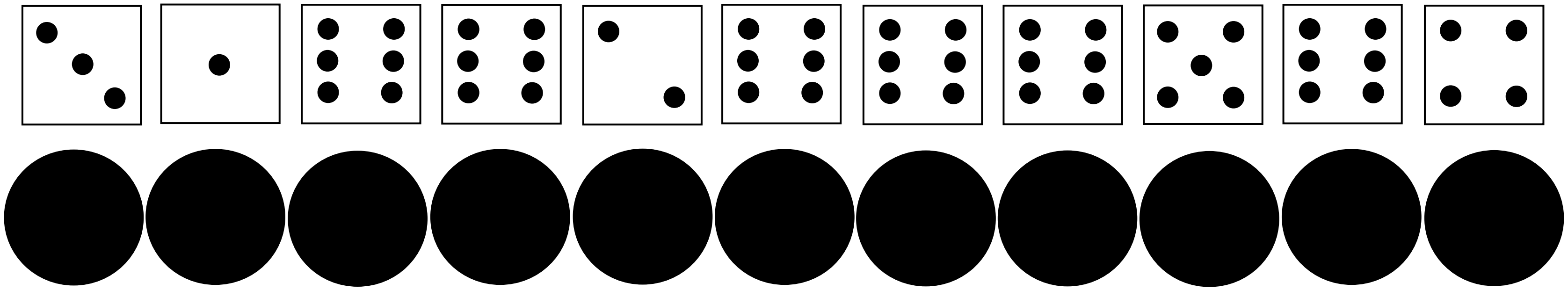
Emission probabilities

	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

The transition matrix

	F	L
F	0.83	0.17
L	0.60	0.40

Hidden Markov Model (HMM)



- States are unknown (hidden)

Questions to HMM

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

The probability that the sequence was generated given a particular path

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

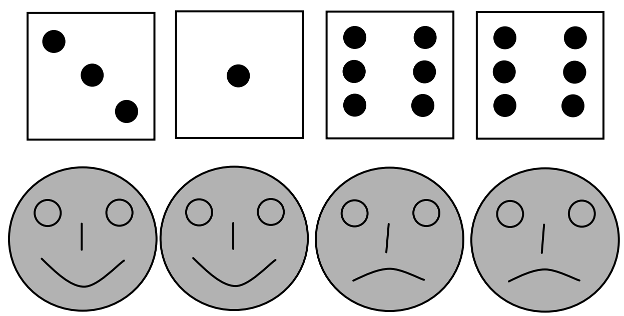
The probability that the sequence M was generated following a path π

- 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

	F	L
F	0.83	0.17
L	0.60	0.40

A suggested path



$P(M \text{ and } \pi)=0.17 * 0.83*0.17 * 0.17*0.50 * 0.60*0.50=0.0006$

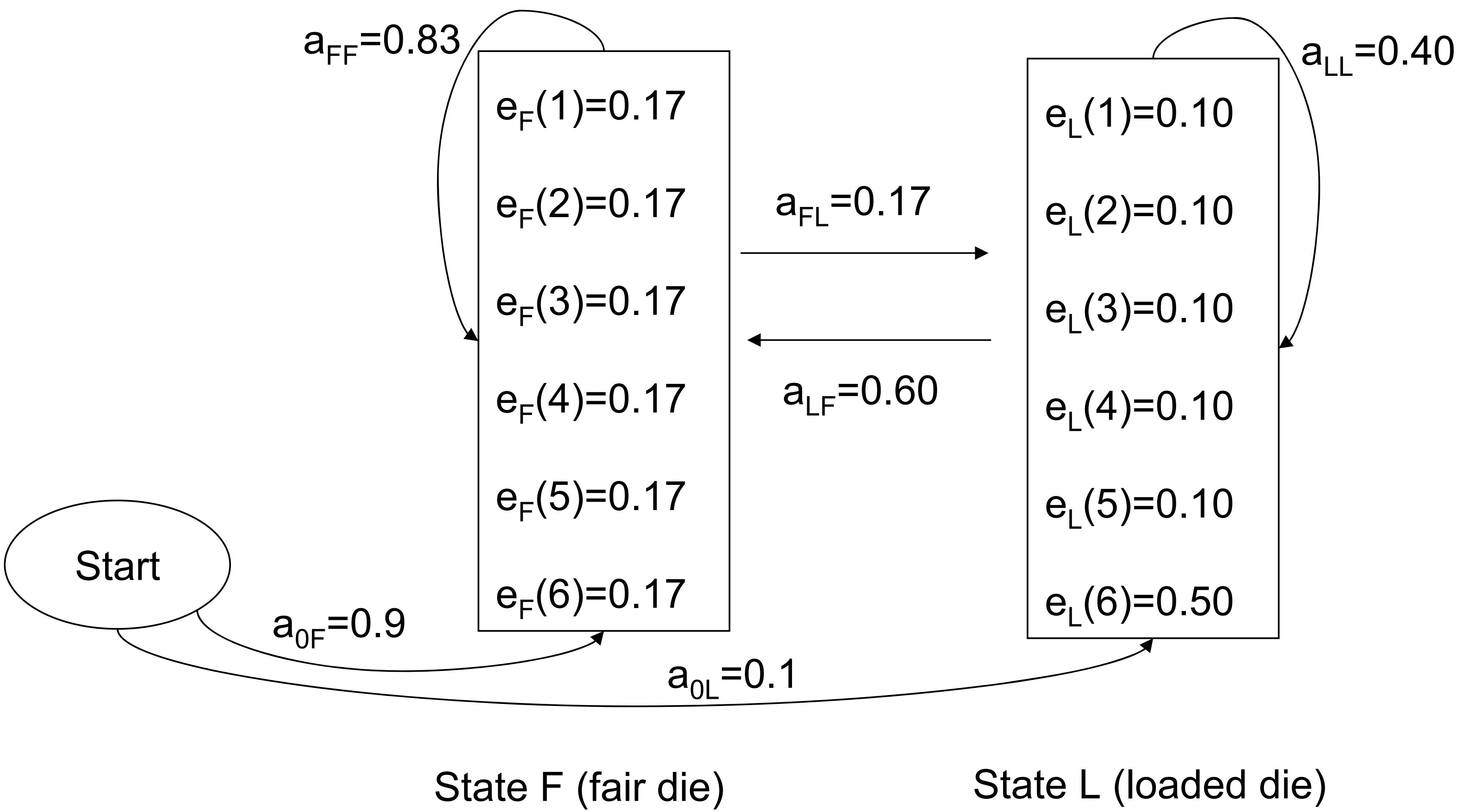
- Repeat for each possible path and choose a path which maximizes
  P(π and M). Total 2^N calculations

Viterbi algorithm for the most probable path

Dynamic programming

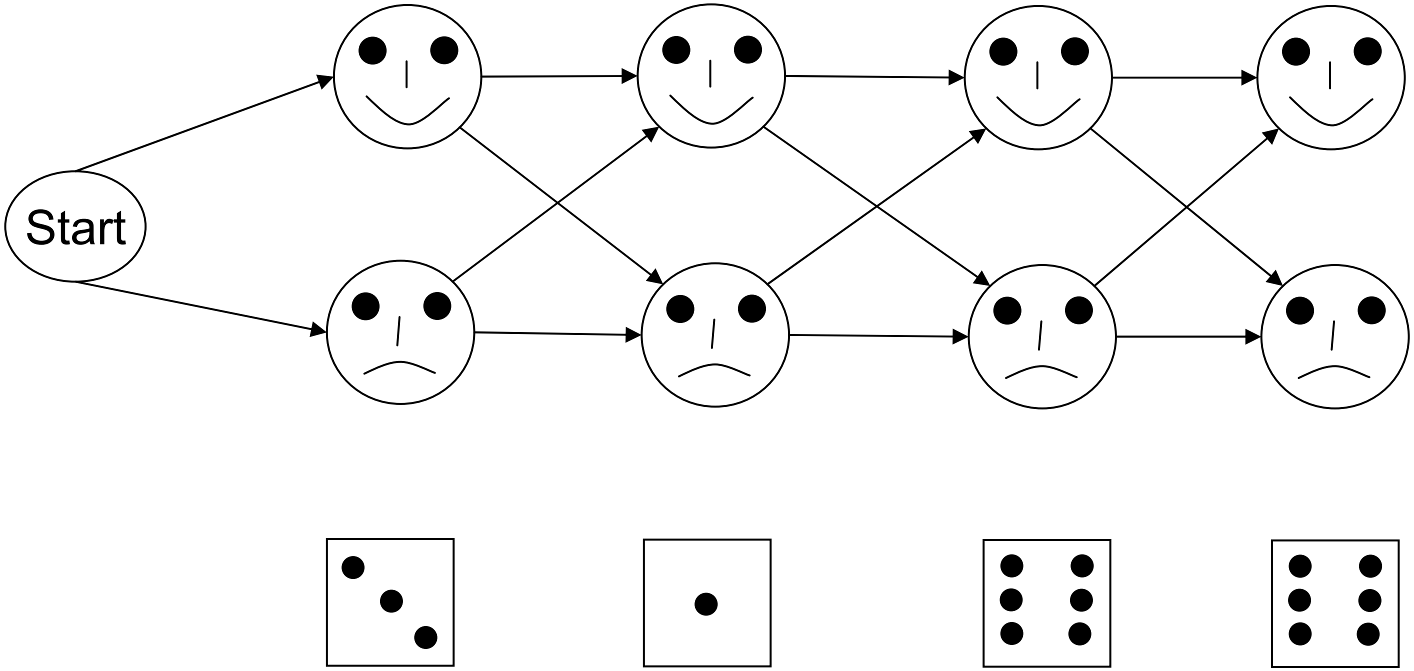
Dynamic programming. Initialization – the probability of choosing a die for the first time

- 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

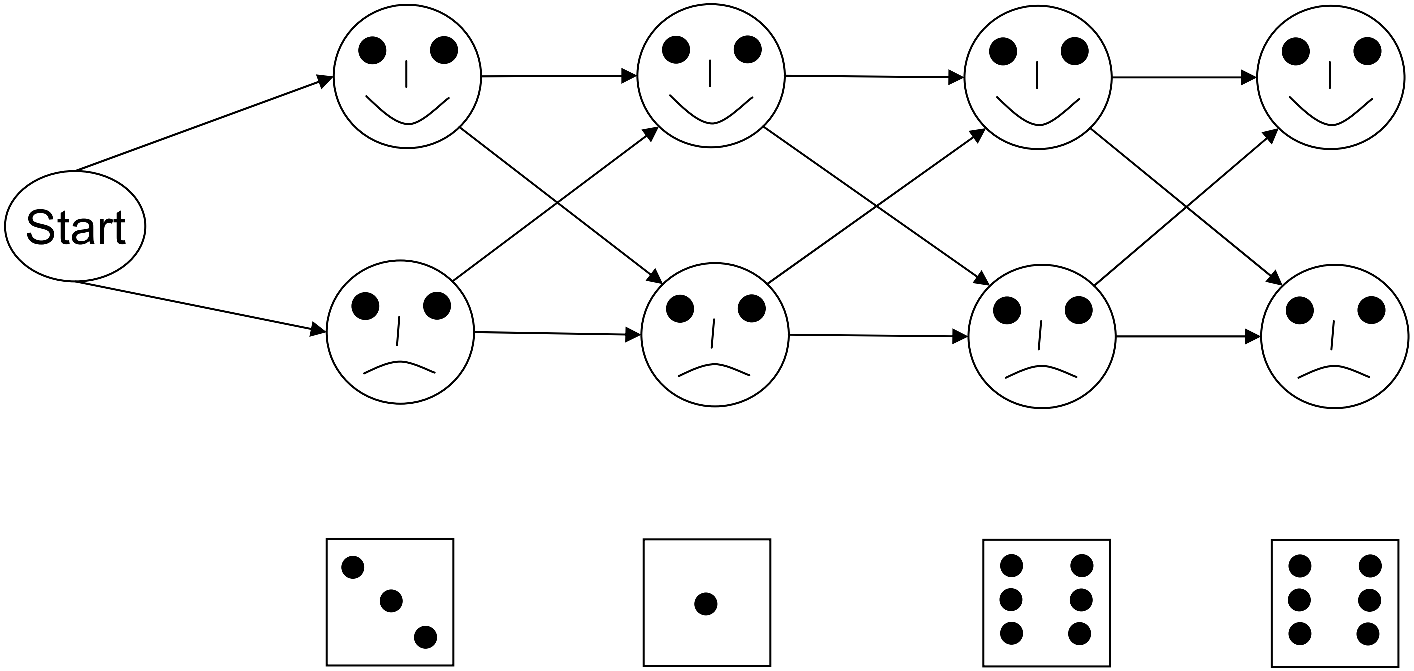
The graph of a process.



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

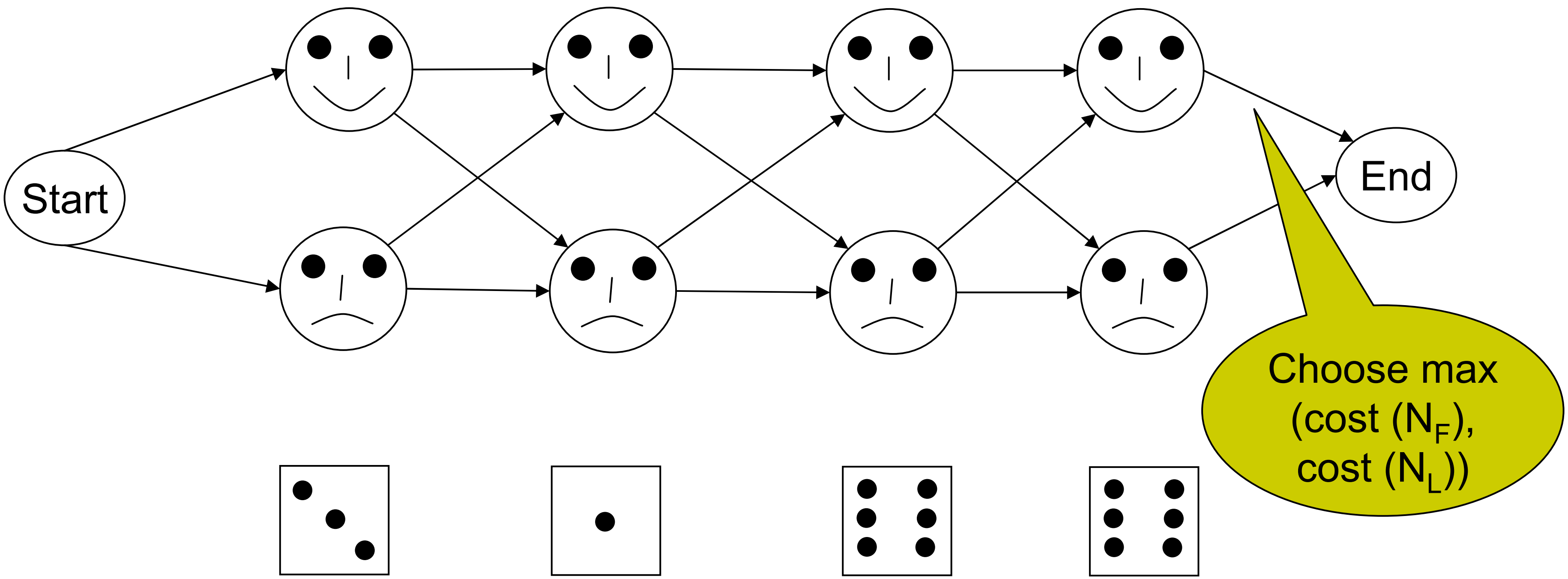
Dynamic programming. Recursion

The graph of a process. We are looking for a path which maximizes the probability of emission M



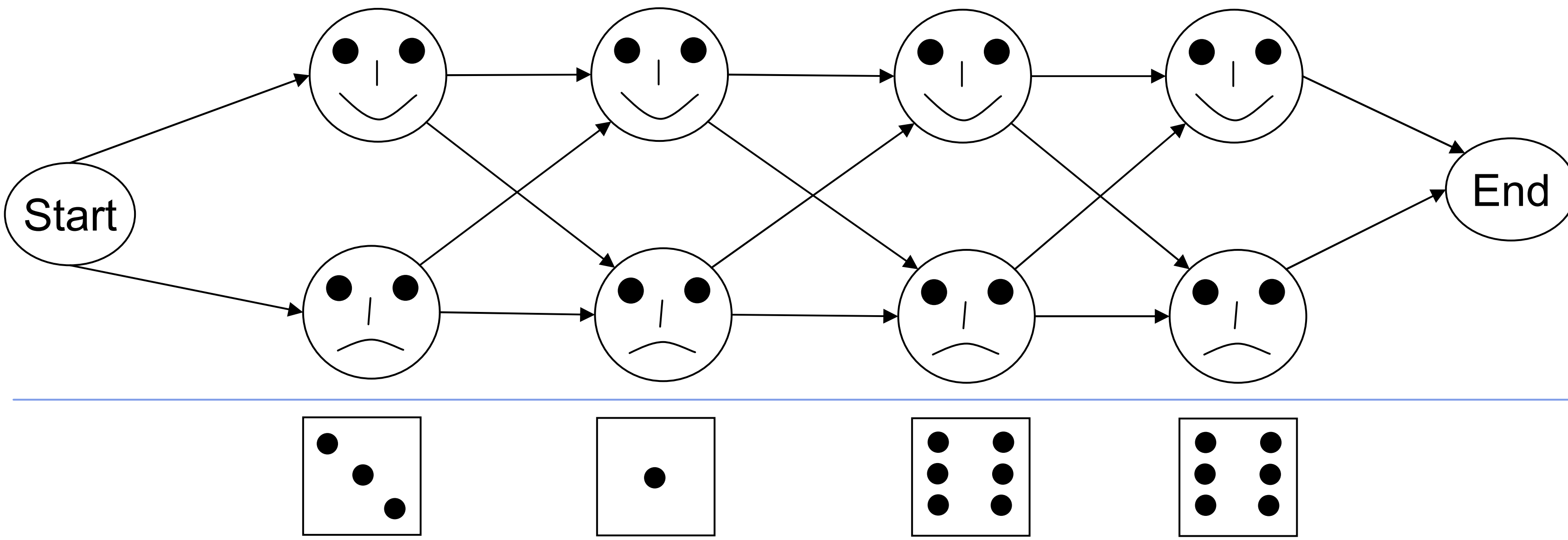
Dynamic programming. Recursion

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. Recursion

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



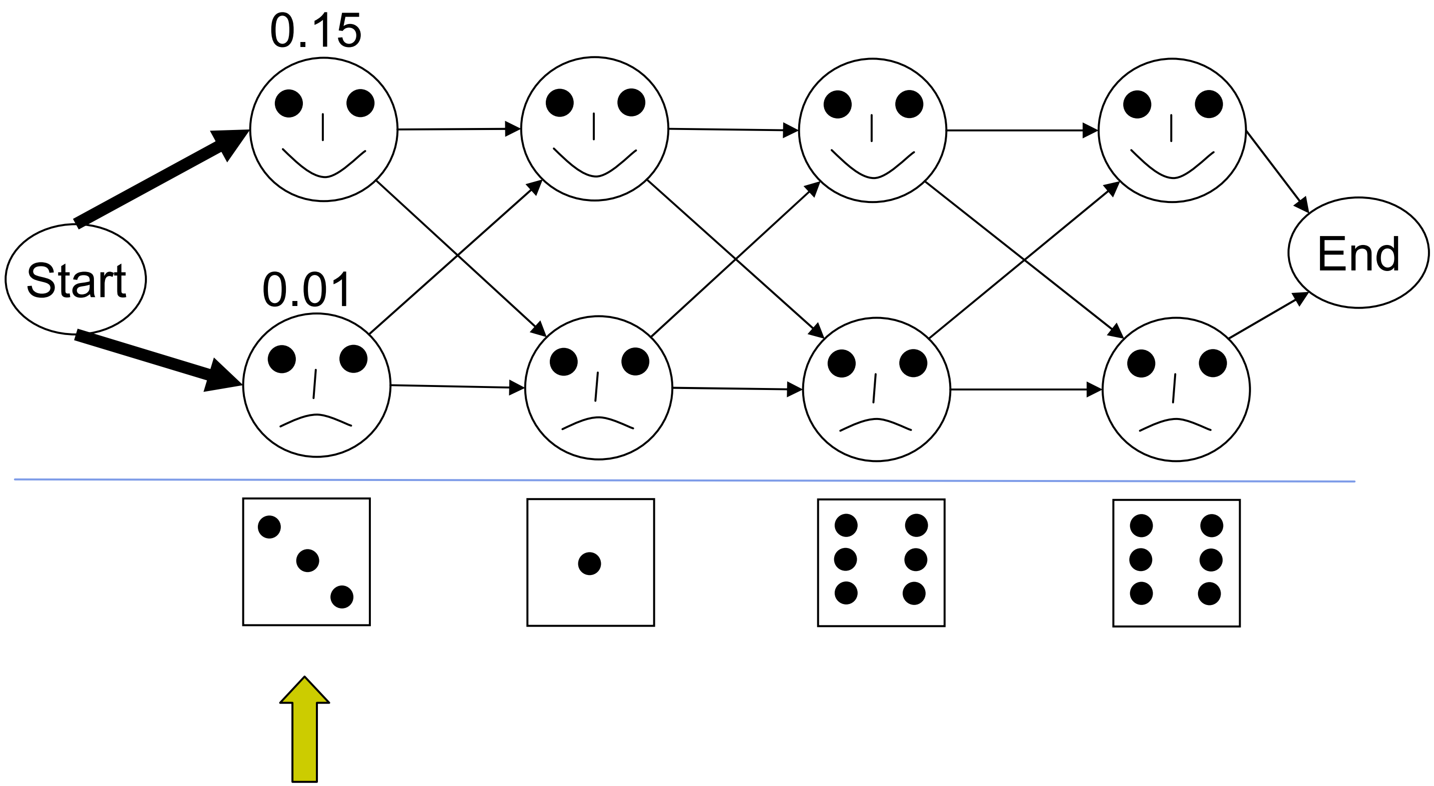
$$P(\pi_{F,i+1}) = \max\{\ P(\pi_{F,i}) * a_{FF},\ \ P(\pi_{L,I}) * a_{LF}\ \} * e_F(M[i+1])$$

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

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

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

Viterbi algorithm. Demo 1

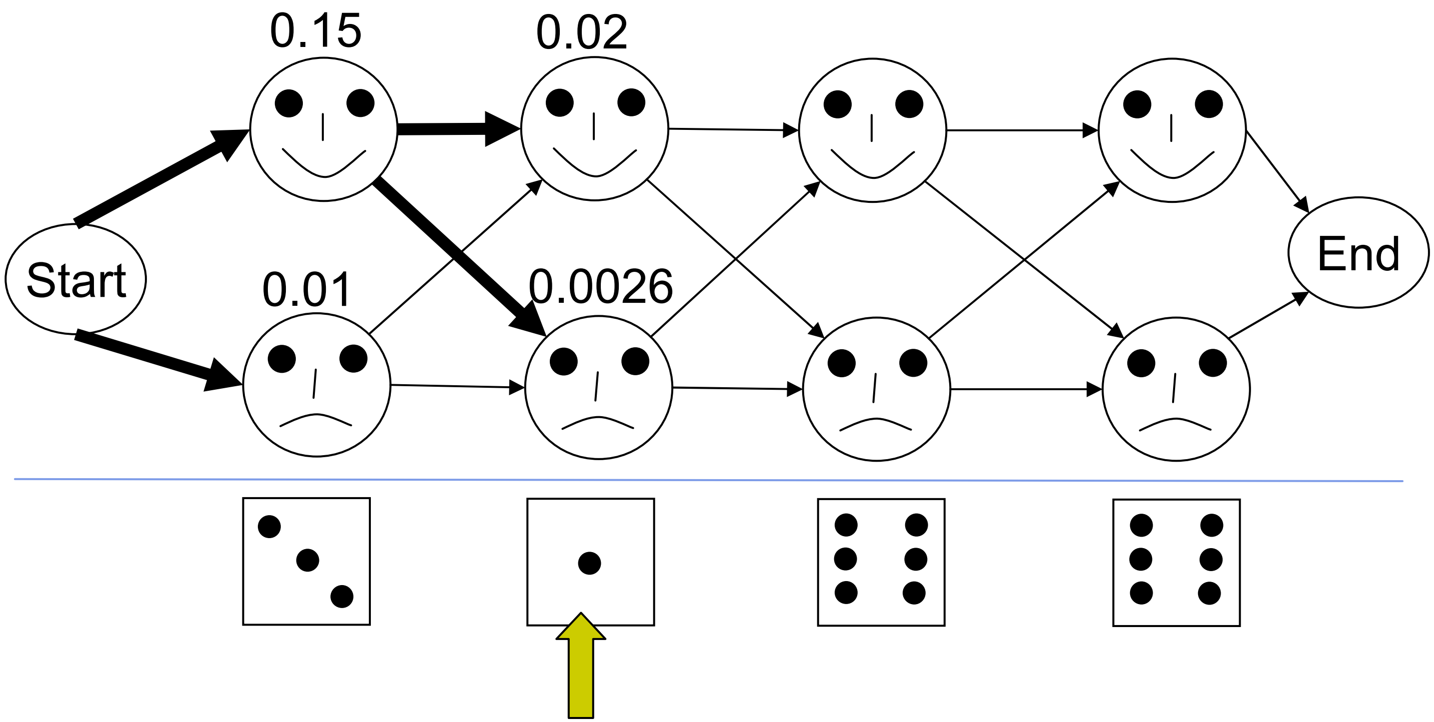


	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	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



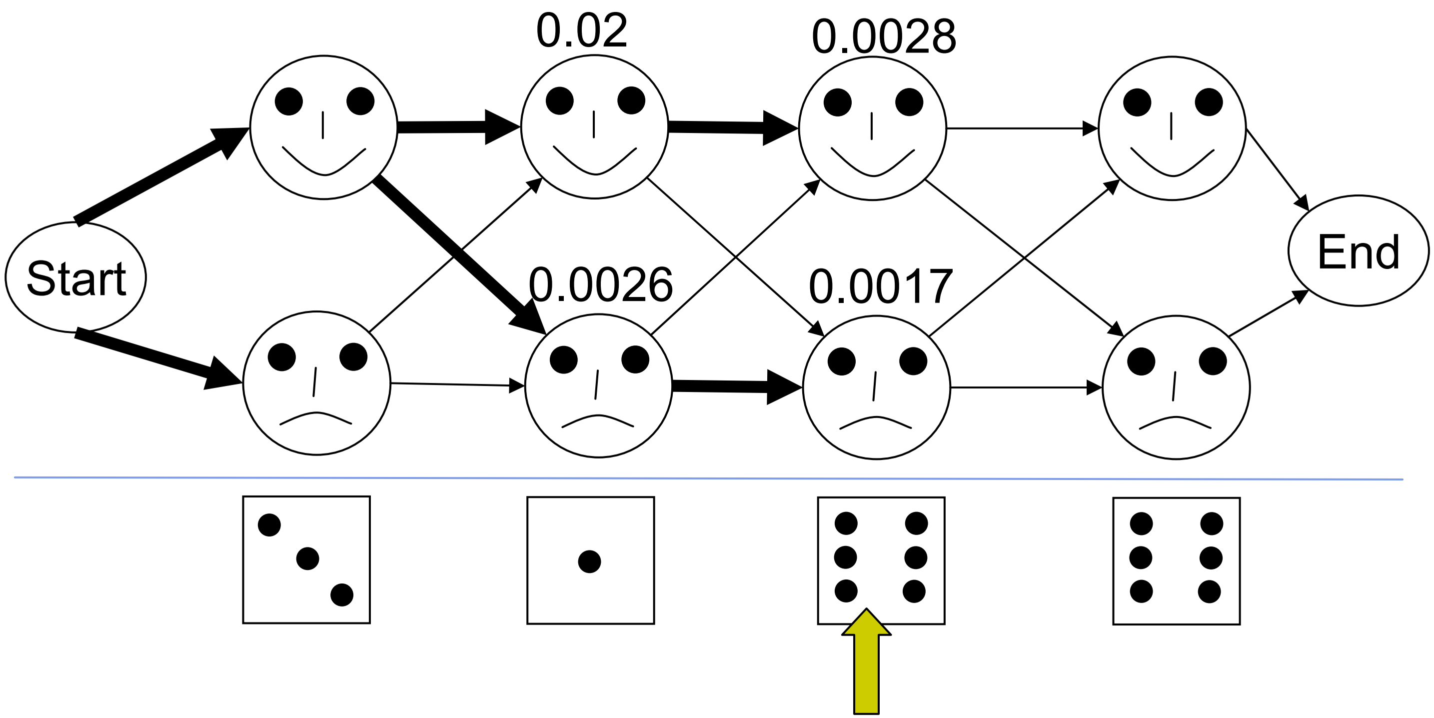
	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	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



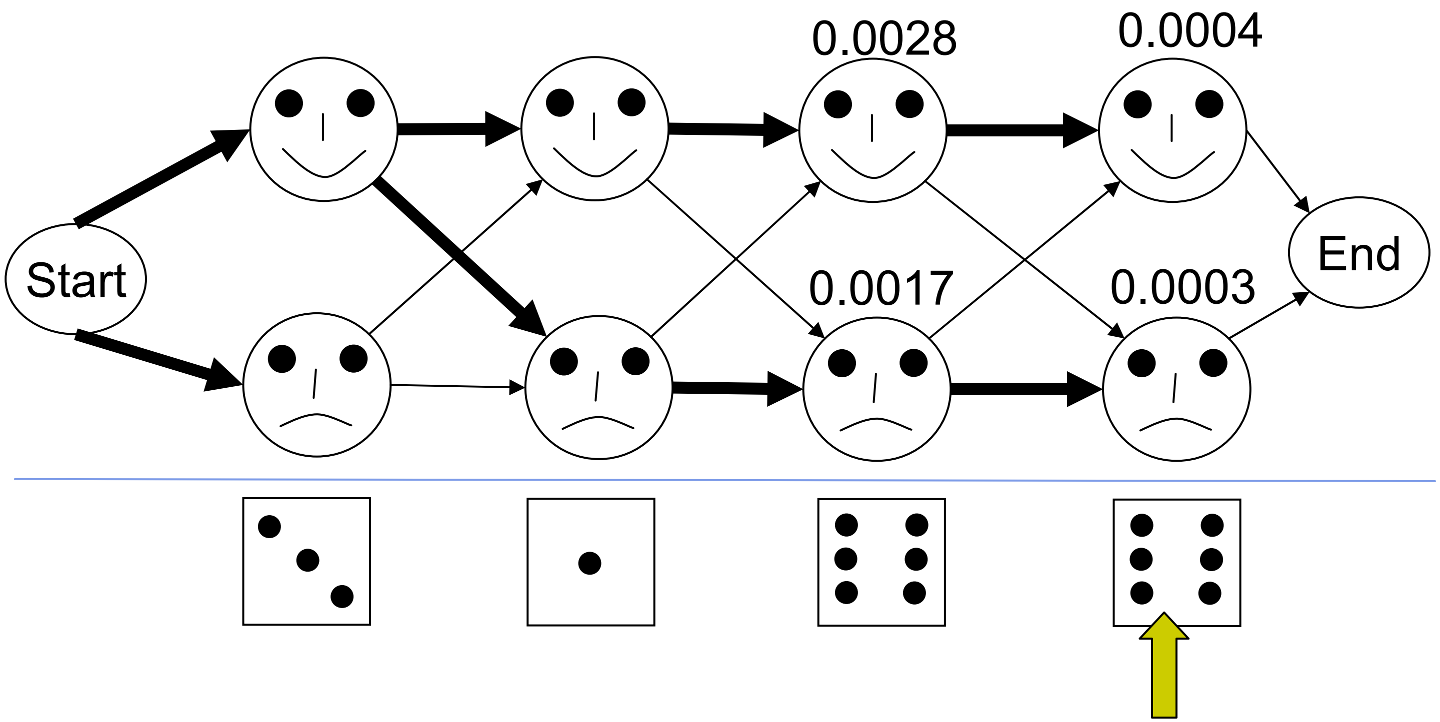
	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	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



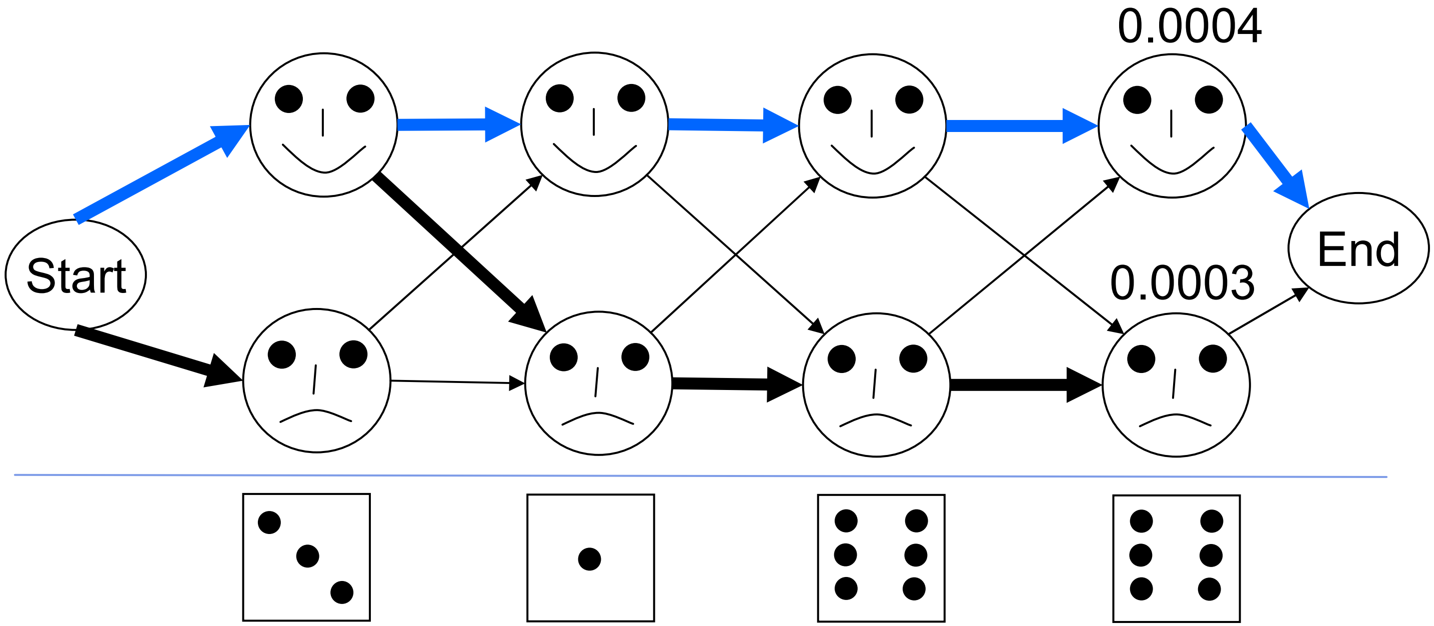
	F	L
1	0.17	0.10
2	0.17	0.10
3	0.17	0.10
4	0.17	0.10
5	0.17	0.10
6	0.17	0.50

	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

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

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

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

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

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

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

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

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

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

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

```
Rolls   315116246446644245311321631164152133625144543631656626566666
Die     FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLL

Rolls   651166453132551245636664631636663162326455236265666625151631
Die     LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLLFFFFFFFFF
Viterbi LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFF

Rolls   222555441666566563564324364131513465146353411126414626253356
Die     FFFFFFFFLLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls   366163666466232534413661661163252562462255265252266435353336
Die     LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls   233121625364414432335163243633665562466662632666612355245242
Die     FFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFF
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLFFFFFFFFFF
```

delay

Missing short stretches

Overall, an underlying hidden pathway explains the given sequence well – the model is good

Exercise 1. Markov models

- In Vancouver, if it rains today, then it rains tomorrow 3 times out of 5. If it is sunny today, it is also sunny tomorrow 1 time out of 3. Build a Markov model for the weather in Vancouver.

Exercise 2. Discrimination by probability

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

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

Find out what of the coins has more probably produced the following sequence of observations

HHHTTHT

Exercise 2. When the coin is biased

- For sequence M of length N with k heads:
- $P(M\ I\ \text{fair coin}) = \Pi_n(1/2) = 1/2^N$
- $P\ (M \mid \text{biased coin}) = \Pi_k(3/4) * \Pi_{N-k}(1/4) = 3^k/4^k * 1/4^{N-k}$
- For this simple model, we can find when

$P(M\ I\ \text{fair coin}) < P\ (M \mid \text{biased coin})$

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

$2^N < 3K$

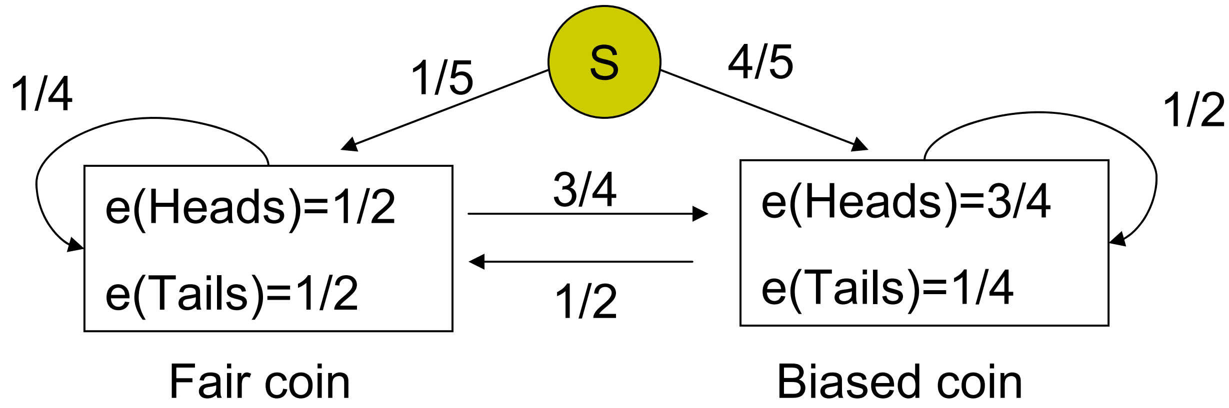
$N\log_2 2 < k\log_2 3$

$K > (\log_2 2/\log_2 3)\ N$

$K > 0.63\ N$

Exercise 3.

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



Observed sequence: HTTHHH

We can answer 2 questions

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

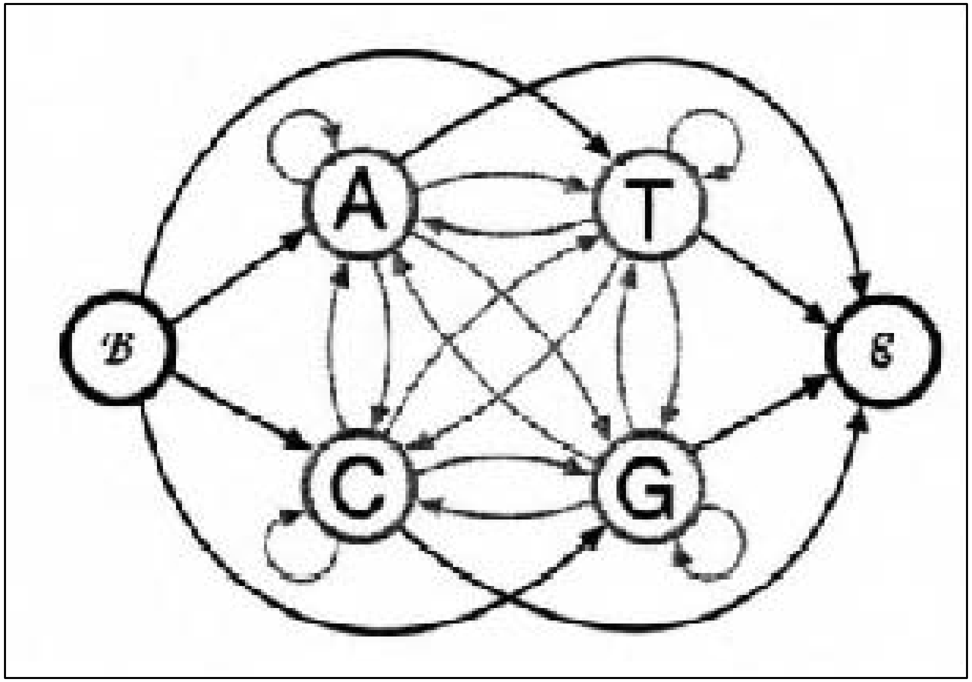
CpG islands

- C nucleotide followed by G is easily methylated
- Methylated C easily becomes T
- The methylation is suppressed in important regulatory regions – around promoters (starting sites of transcription)
- Thus, an overall low frequency of CG di-nucleotide is significantly increased in the gene promoter regions

Biological questions

- Given a short stretch of DNA sequence, how can we determine whether it came from a CpG island or not
- Given a lon un-annotated DNA sequence, find CpG islands in it

Markov model for DNA sequence



- Usually, the end of sequence is not modelled in Markov chain – sequence can end anywhere

Transition probability estimation from real DNA sequences

- From 48 CpG islands of a total length 60,000 nucleotides, and from a regular DNA stretches, the transition probabilities for each pair of nucleotides were estimated (expected 0.25 if at random)

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

+	A	C	G	T
A	0.18	0.27	0.43	0.12
C	0.17	0.37	0.27	0.19
G	0.16	0.34	0.38	0.12
T	0.08	0.36	0.38	0.18

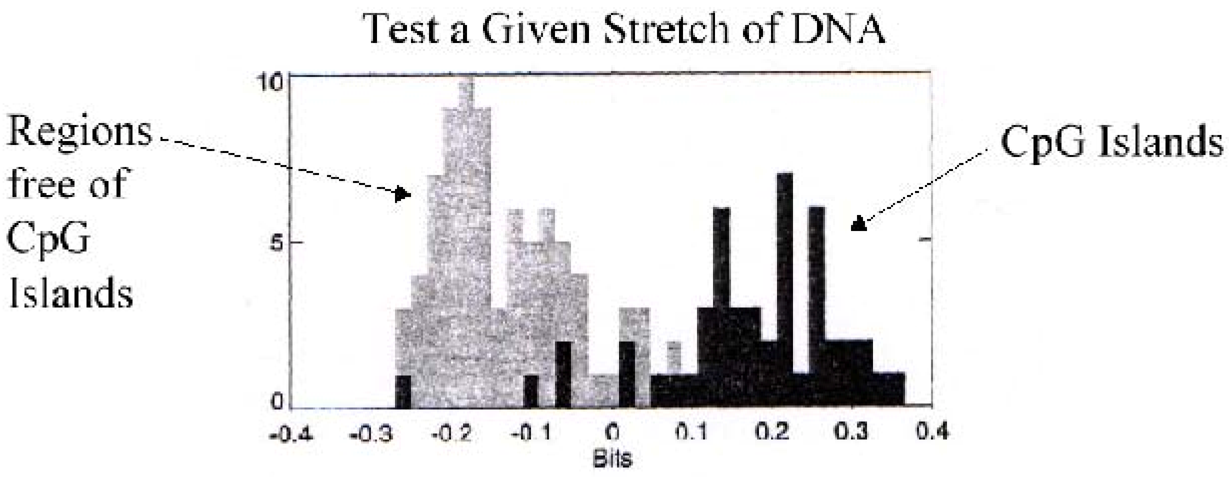
-	A	C	G	T
A	0.30	0.20	0.29	0.21
C	0.32	0.30	0.08	0.30
G	0.25	0.25	0.30	0.20
T	0.18	0.24	0.29	0.29

Am I in the CpG island?

- To use these (+) and (-) models for discrimination for a given sequence we calculate the log-odds ratio:

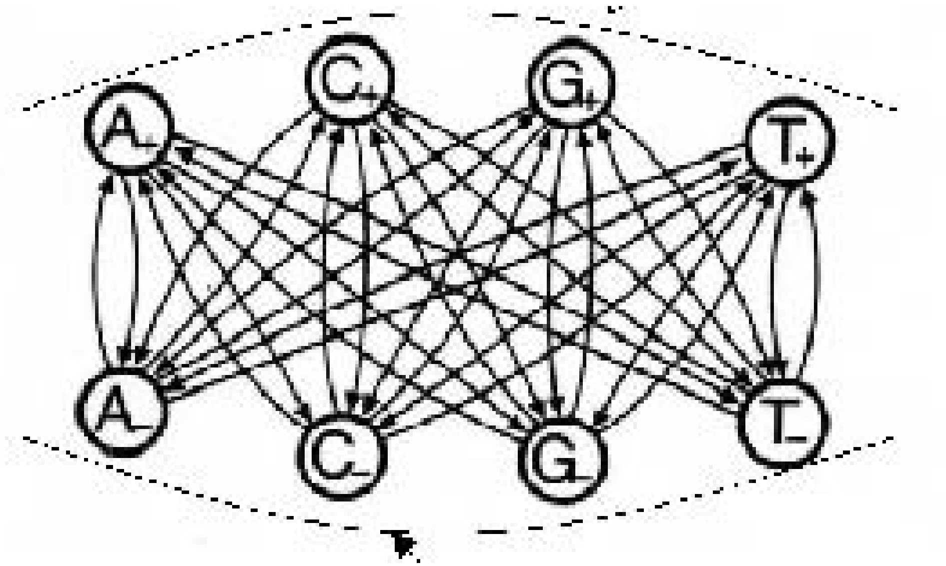
Score(M)=log [P(M|given model +)/P(M|given model -)]

If this value is positive, we are in the CpG island, if not, we are not



Test on another set of labeled DNA sequences

Finding CpG islands - HMM



- The relabeling is the critical step. The essential difference between a Markov chain and an HMM is that for HMM there is no 1-to-1 correspondence between the states and the symbols
- By looking at a single symbol, there is no way to tell whether it came from state C+ or C-

The most probable path through the sequence of states

- The most probable path for sequence CGCG

v		C	G	C	G
$\mathcal{B}$	1	0	0	0	0
A_+	0	0	0	0	0
C_+	0	$\boxed{0.13}$	0	$\boxed{0.012}$	0
G_+	0	0	$\boxed{0.034}$	0	$\boxed{0.0032}$
T_+	0	0	0	0	0
A_-	0	0	0	0	0
C_-	0	0.13	0	0.0026	0
G_-	0	0	0.010	0	0.00021
T_-	0	0	0	0	0

When we apply the Viterbi algorithm to a long un-annotated DNA sequence, the states will switch between + and -, giving suggested boundaries for CpG islands

Defining the model for HMM

- 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
    - $a_{from,to}=count_{from,to}/\Sigma_x count_{from,x}$
    - $e_{state\ i}(symbol\ j)=count_{state\ i}(symbol\ j)/\Sigma_y(symbol\ y)$
  - 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

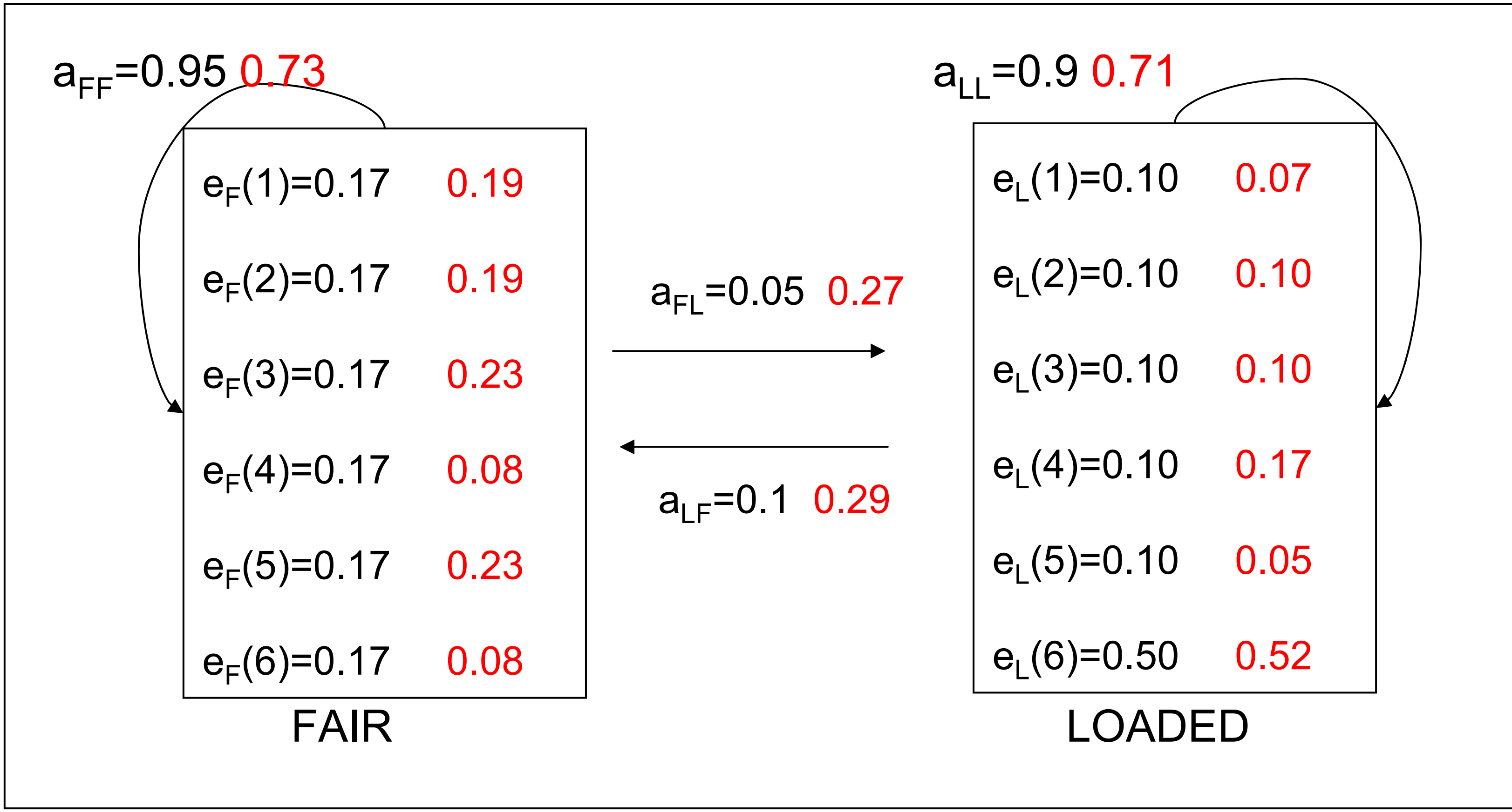
- Pick a set of random parameters
- Find the most probable path of states according to this set of parameters
- This path partitions the sequences into partitions according to the states
- Calculate new set of parameters, now from the known states
- Repeat – find the most probable path with the new parameters etc. – 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(observed data| Θ, π^*)

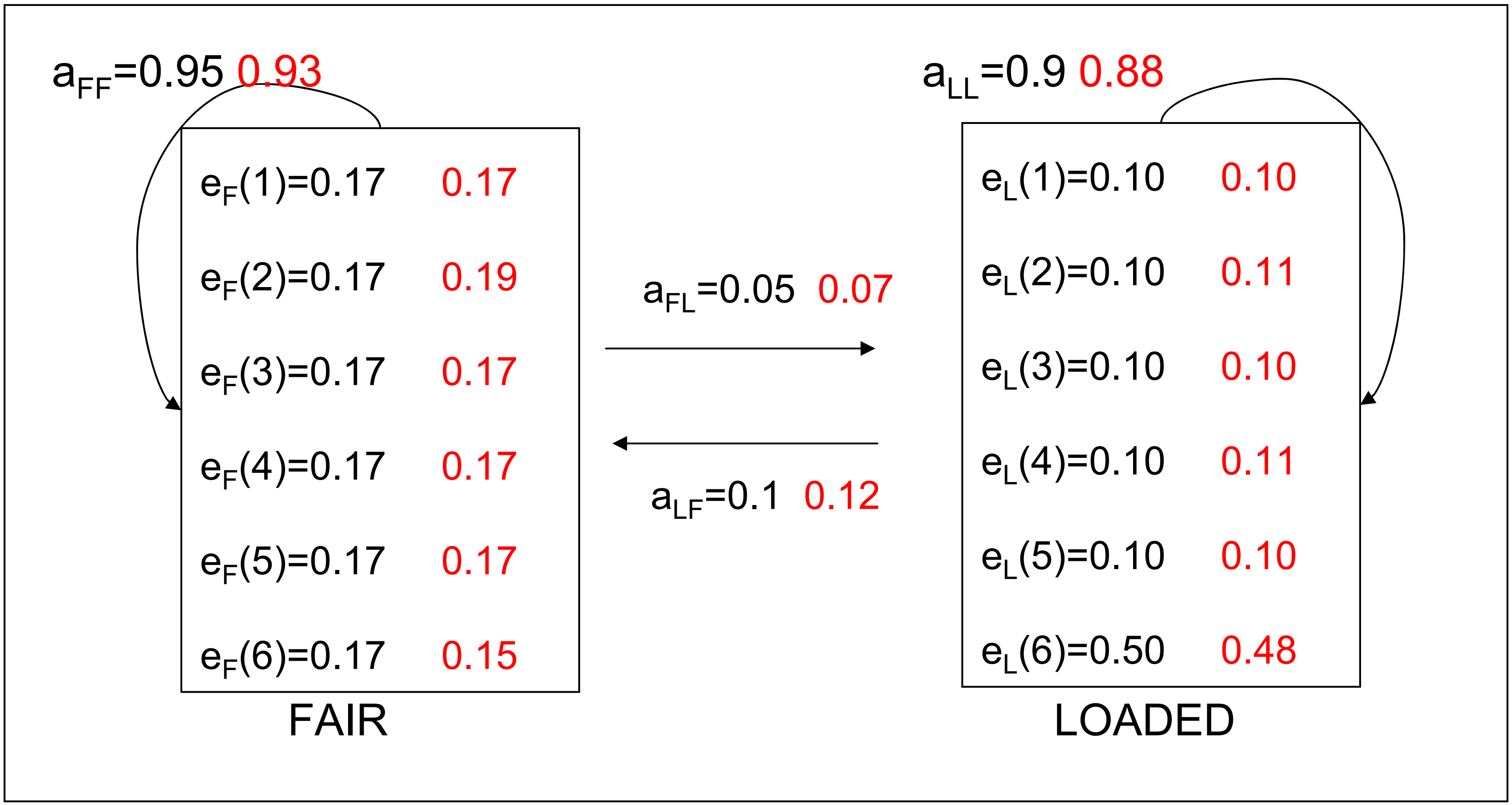
  and not P(observed data | Θ) which we ideally want

Parameter estimation – illustration 1



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

Parameter estimation – illustration 2



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