Hidden Markov Models

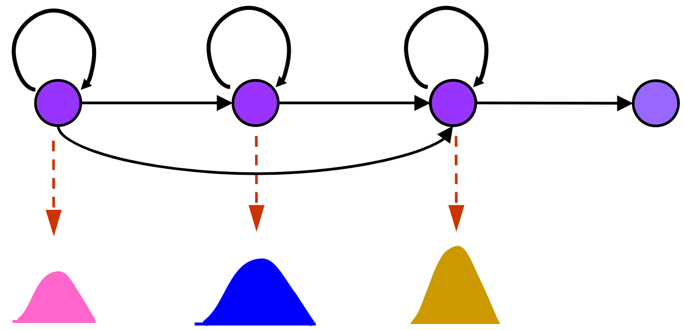
Class 15. 12 Oct 2010

Administrivia

- HW2 – due Tuesday

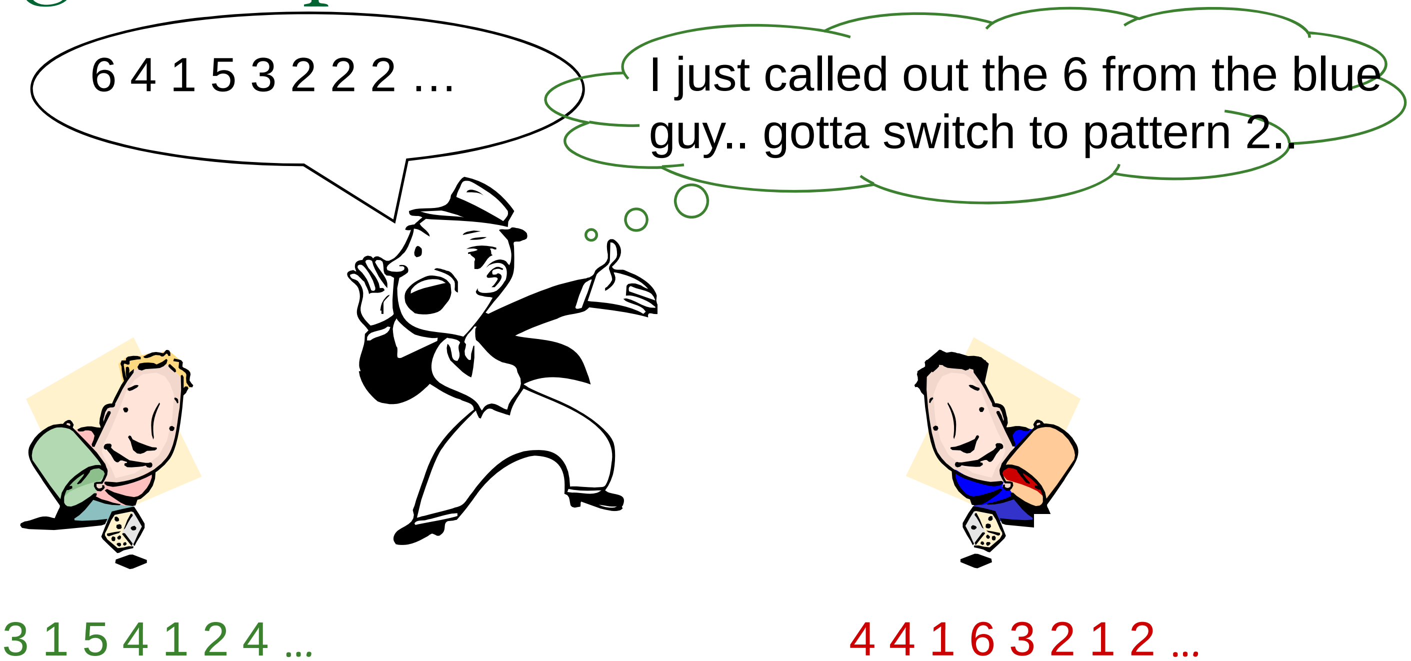
- Is everyone on the “projects” page?
  - Where are your project proposals?

Recap: What is an HMM



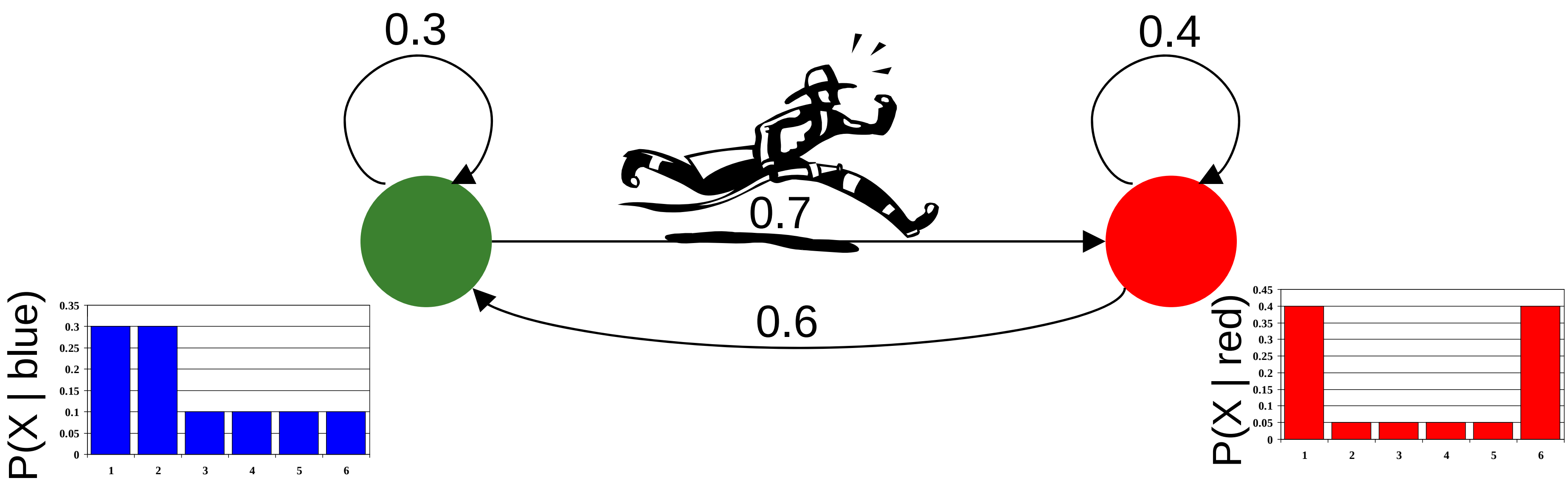
- "Probabilistic function of a markov chain"
- Models a dynamical system
- System goes through a number of states
  - Following a Markov chain model
- On arriving at any state it generates observations according to a state-specific probability distribution

A Thought Experiment



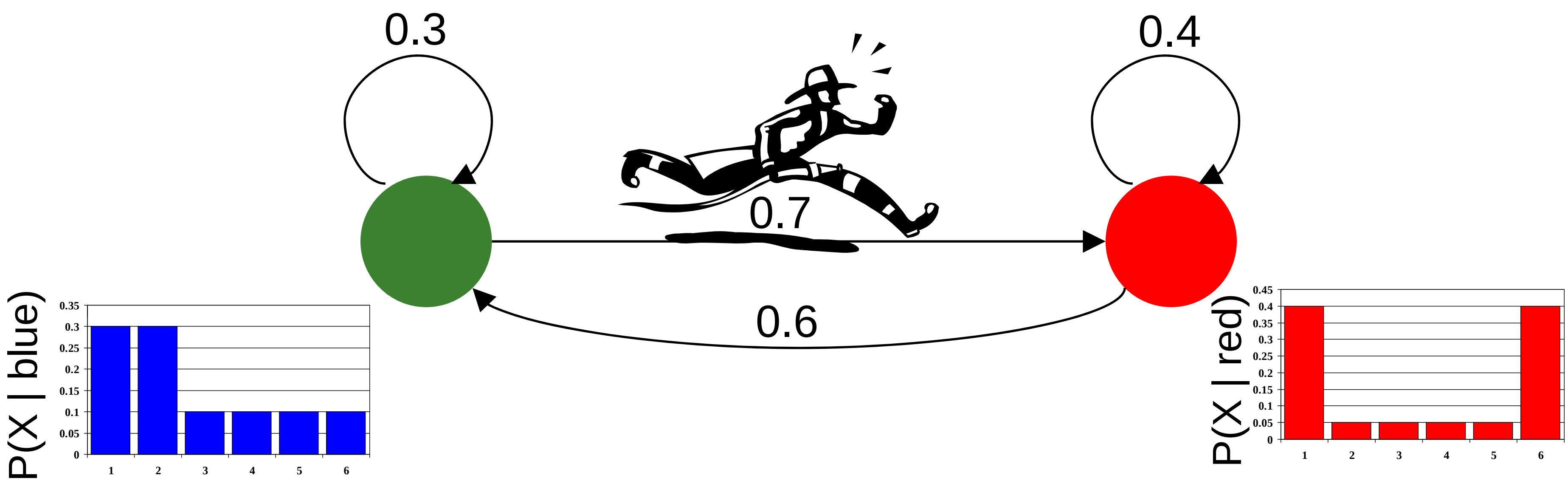
- Two "shooters" roll dice
- A caller calls out the number rolled. We only get to hear what he calls out
- The caller behaves randomly
  - If he has just called a number rolled by the blue shooter, his next call is that of the red shooter 70% of the time
  - But if he has just called the red shooter, he has only a 40% probability of calling the red shooter again in the next call
- How do we characterize this?

A Thought Experiment



- The dots and arrows represent the “states” of the caller
  - When he’s on the blue circle he calls out the blue dice
  - When he’s on the red circle he calls out the red dice
  - The histograms represent the probability distribution of the numbers for the blue and red dice

A Thought Experiment

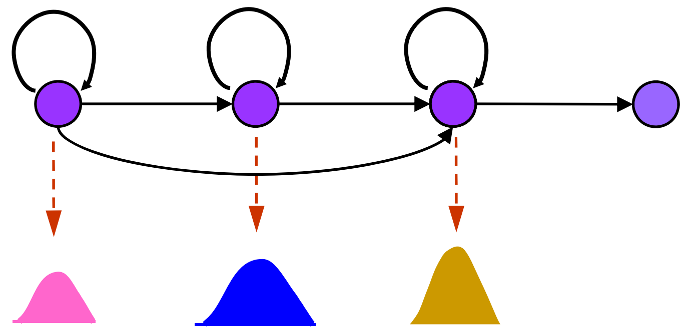


- When the caller is in any state, he calls a number based on the probability distribution of that state
  - We call these state output distributions
- At each step, he moves from his current state to another state following a probability distribution
  - We call these transition probabilities
- The caller is an HMM!!!

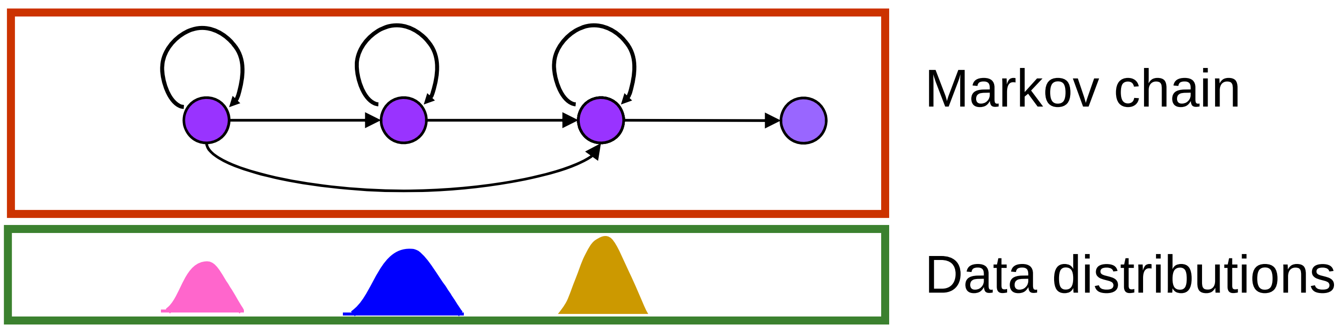
What is an HMM

- HMMs are statistical models for (causal) processes
- The model assumes that the process can be in one of a number of states at any time instant
- The state of the process at any time instant depends only on the state at the previous instant (causality, Markovian)
- At each instant the process generates an observation from a probability distribution that is specific to the current state
- The generated observations are all that we get to see
    - the actual state of the process is not directly observable
        - Hence the qualifier hidden

Hidden Markov Models



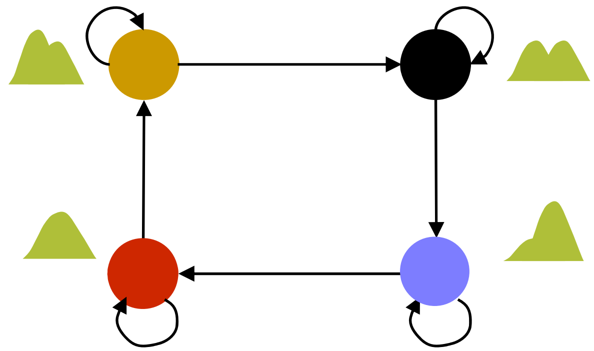
- A Hidden Markov Model consists of two components
  - A state/transition backbone that specifies how many states there are, and how they can follow one another
  - A set of probability distributions, one for each state, which specifies the distribution of all vectors in that state



- This can be factored into two separate probabilistic entities
  - A probabilistic Markov chain with states and transitions
  - A set of data probability distributions, associated with the states

How an HMM models a process

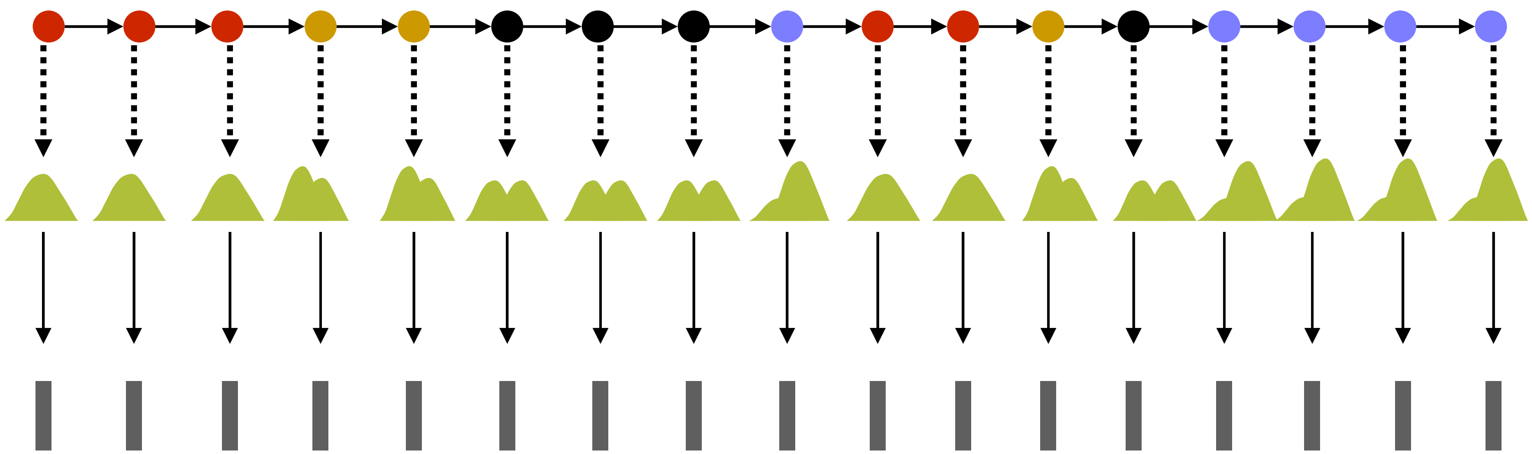
HMM assumed to be generating data



state sequence

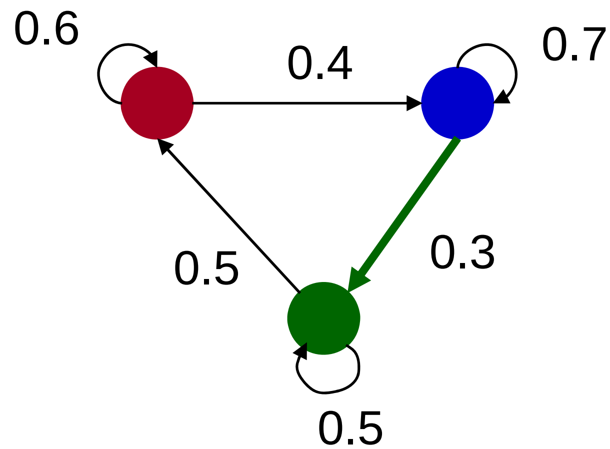
state distributions

observation sequence

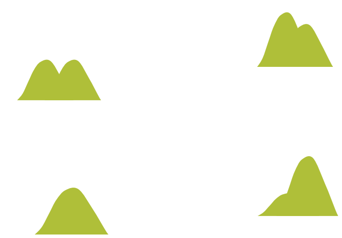


HMM Parameters

- The *topology* of the HMM
  - Number of states and allowed transitions
  - E.g. here we have 3 states and cannot go from the blue state to the red
- The transition probabilities
  - Often represented as a matrix as here
  - T_{ij} is the probability that when in state i, the process will move to j
- The probability π_i of beginning at any state s_i
  - The complete set is represented as π
- The *state output distributions*



$$T = \begin{pmatrix} .6 & .4 & 0 \\ 0 & .7 & .3 \\ .5 & 0 & .5 \end{pmatrix}$$



HMM state output distributions

- The state output distribution is the distribution of data produced from any state
- Typically modelled as Gaussian

$$P(x \mid s_i) = Gaussian(x; \mu_i, \Theta_i) = \frac{1}{\sqrt{(2\pi)^d |\Theta_i|}} e^{-0.5(x-\mu_i)^T \Theta_i^{-1} (x-\mu_i)}$$

- The paremeters are μ_i and Θ_i

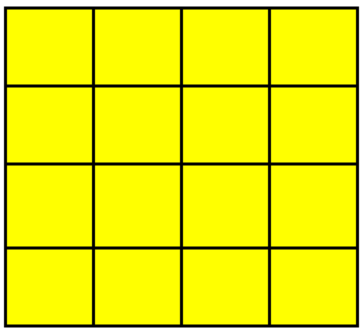
- More typically, modelled as Gaussian mixtures

$$P(x \mid s_i) = \sum_{j=0}^{K-1} w_{i,j} Gaussian(x; \mu_{i,j}, \Theta_{i,j})$$

  - Other distributions may also be used
  - E.g. histograms in the dice case

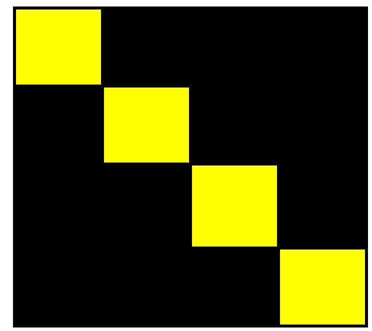
The Diagonal Covariance Matrix

Full covariance:
all elements are
non-zero



$-0.5(x-\mu)^T\Theta^{-1}(x-\mu)$

Diagonal covariance:
off-diagonal elements
are zero



$-\Sigma_i (x_i-\mu_i)^2 / 2\sigma_i^2$

- For GMMs it is frequently assumed that the feature vector dimensions are all *independent* of each other
- *Result*: The covariance matrix is reduced to a diagonal form
  - The determinant of the diagonal Θ matrix is easy to compute

Three Basic HMM Problems

- What is the probability that it will generate a specific observation sequence

- Given a observation sequence, how do we determine which observation was generated from which state
  - The state segmentation problem

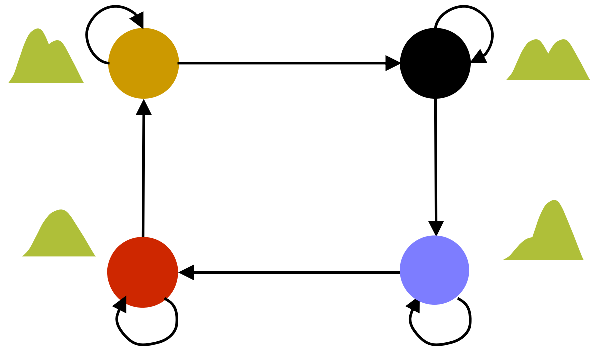
- How do we *learn* the parameters of the HMM from observation sequences

Computing the Probability of an Observation Sequence

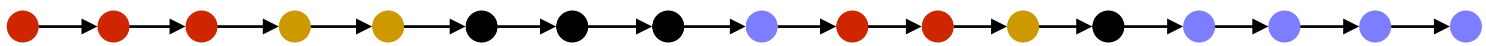
- Two aspects to producing the observation:
  - Progressing through a sequence of states
  - Producing observations from these states

Progressing through states

HMM assumed to be generating data



state sequence



- The process begins at some state (red) here
- From that state, it makes an allowed transition
  - To arrive at the same or any other state
- From that state it makes another allowed transition
  - And so on

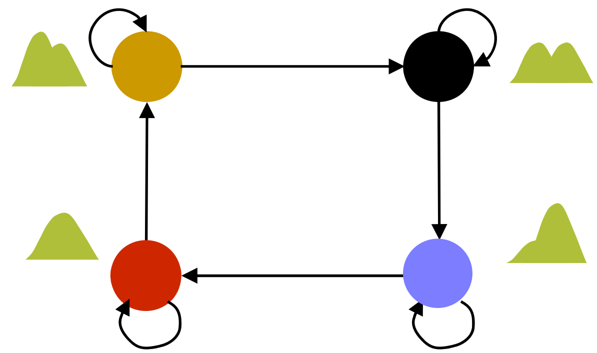
Probability that the HMM will follow a particular state sequence

$$P(s_1, s_2, s_3, \ldots) = P(s_1)P(s_2|s_1)P(s_3|s_2)\ldots$$

- $P(s_1)$ is the probability that the process will initially be in state s_1
- $P(s_i \mid s_i)$ is the transition probability of moving to state s_i at the next time instant when the system is currently in s_i
  - Also denoted by T_{ij} earlier

Generating Observations from States

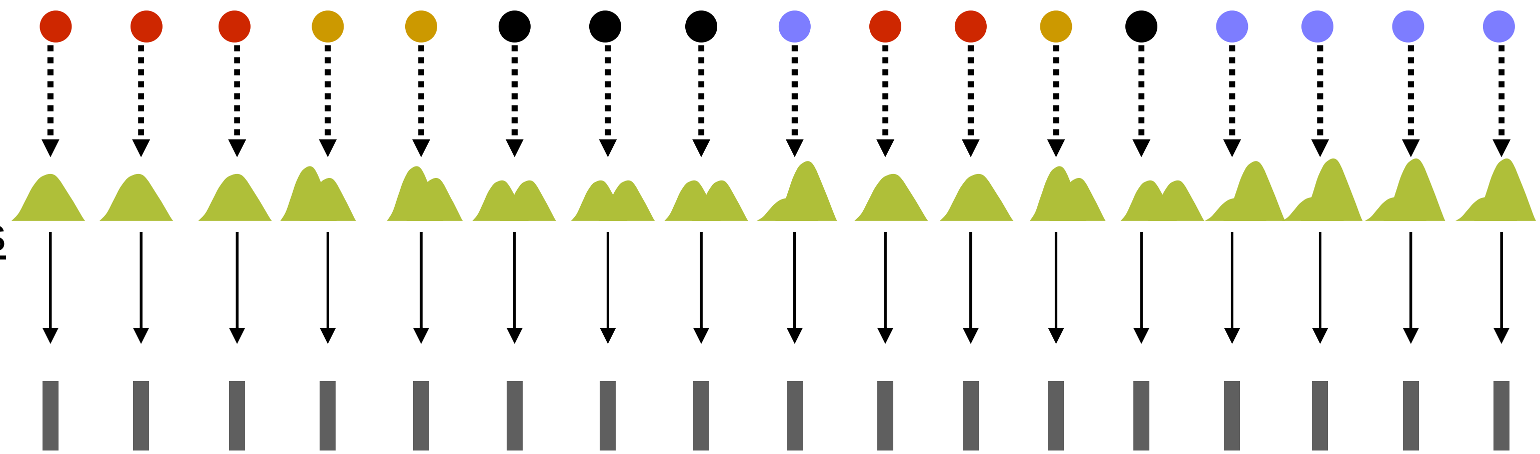
HMM assumed to be generating data



state sequence

state distributions

observation sequence



- At each time it generates an observation from the state it is in at that time

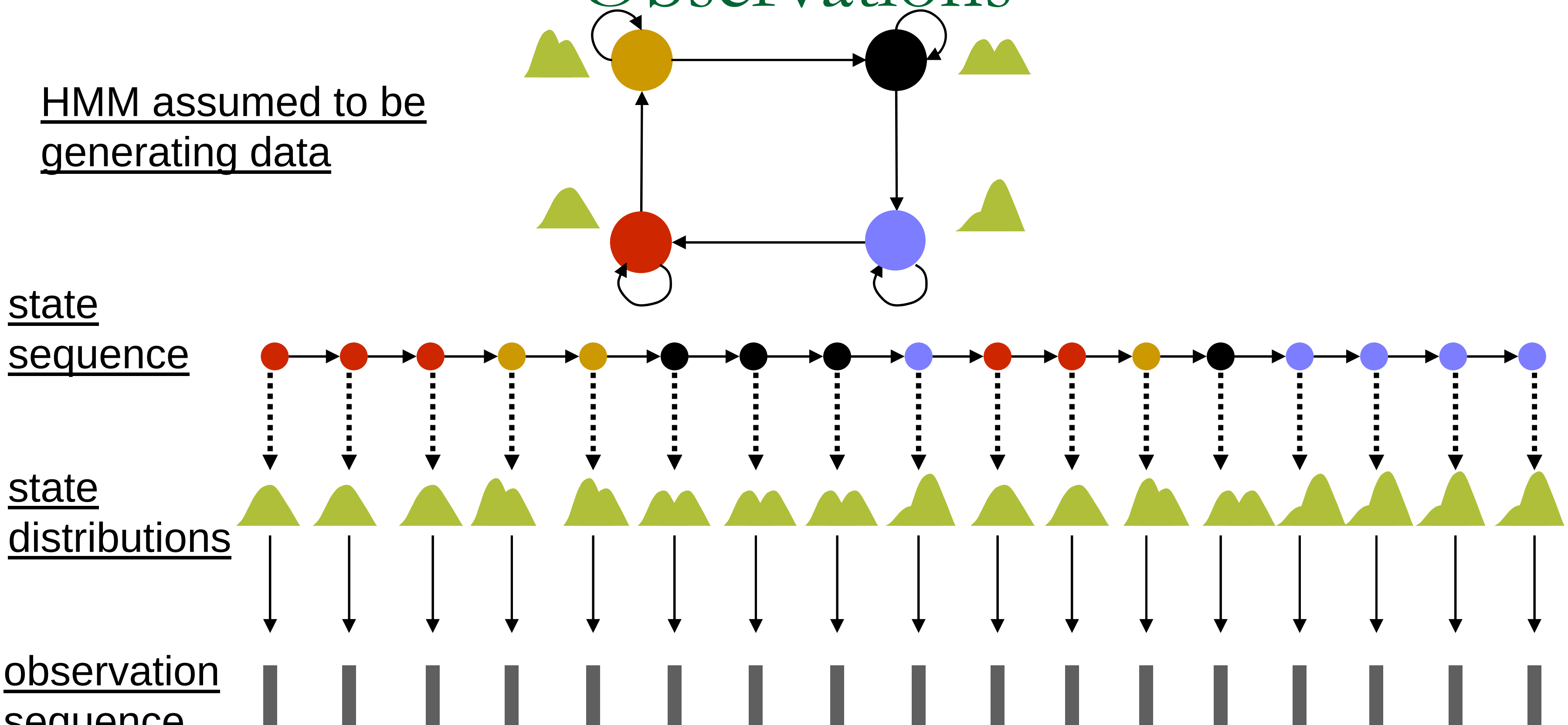
Probability that the HMM will generate a particular observation sequence given a state sequence (state sequence known)

$$P(o_1, o_2, o_3, \ldots | s_1, s_2, s_3, \ldots) = P(o_1|s_1)P(o_2|s_2)P(o_3|s_3)\ldots$$

Computed from the Gaussian or Gaussian mixture for state s_1

- $P(o_i \mid s_i)$ is the probability of generating observation o_i when the system is in state s_i

Proceeding through States and Producing Observations



- At each time it produces an observation and makes a transition

Probability that the HMM will generate a particular state sequence and from it, a particular observation sequence

$$P(o_1, o_2, o_3, \ldots, s_1, s_2, s_3, \ldots) =$$

$$P(o_1, o_2, o_3, \ldots | s_1, s_2, s_3, \ldots) P(s_1, s_2, s_3, \ldots) =$$

$$P(o_1|s_1) P(o_2|s_2) P(o_3|s_3) \ldots P(s_1) P(s_2|s_1) P(s_3|s_2) \ldots$$

Probability of Generating an Observation Sequence

- The precise state sequence is not known
- All possible state sequences must be considered

$$P(o_1, o_2, o_3, \ldots) = \sum_{\substack{all.possible \\ state.sequences}} P(o_1, o_2, o_3, \ldots, s_1, s_2, s_3, \ldots) =$$

$$\sum_{\substack{all.possible \\ state.sequences}} P(o_1|s_1) P(o_2|s_2) P(o_3|s_3) \ldots P(s_1) P(s_2|s_1) P(s_3|s_2) \ldots$$

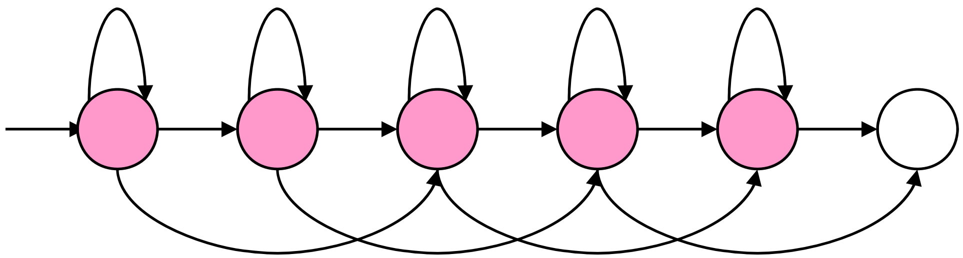
Computing it Efficiently

- Explicit summing over all state sequences is not tractable
  - A very large number of possible state sequences

- Instead we use the forward algorithm

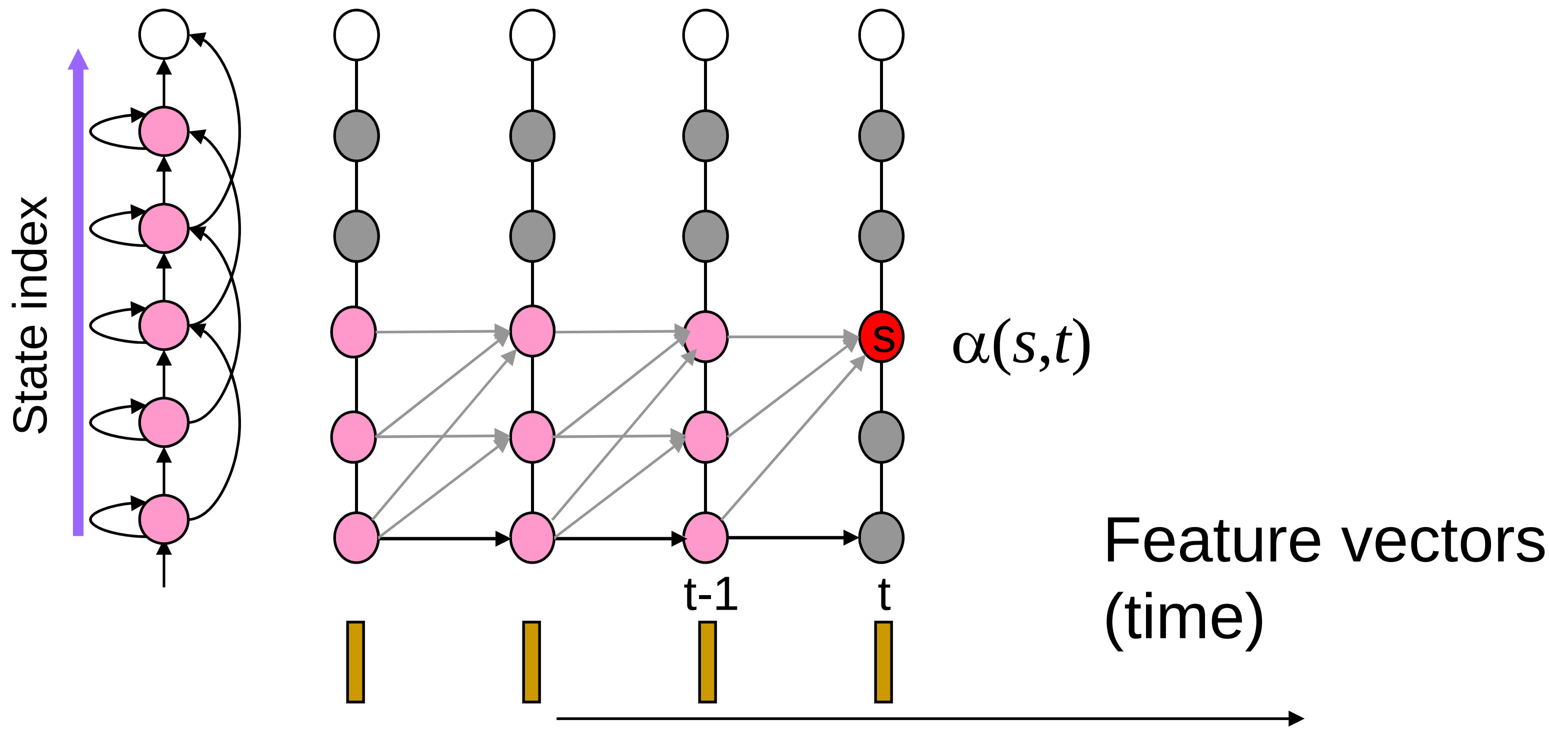
- A dynamic programming technique.

Illustrative Example



- Example: a generic HMM with 5 states and a "terminating state".
  - Left to right topology
    - $P(s_i)$ = 1 for state 1 and 0 for others
  - The arrows represent transition for which the probability is not 0
- *Notation:*
  - $P(s_i \mid s_j) = T_{ij}$
  - We represent $P(o_t \mid s_i) = b_i(t)$ for brevity

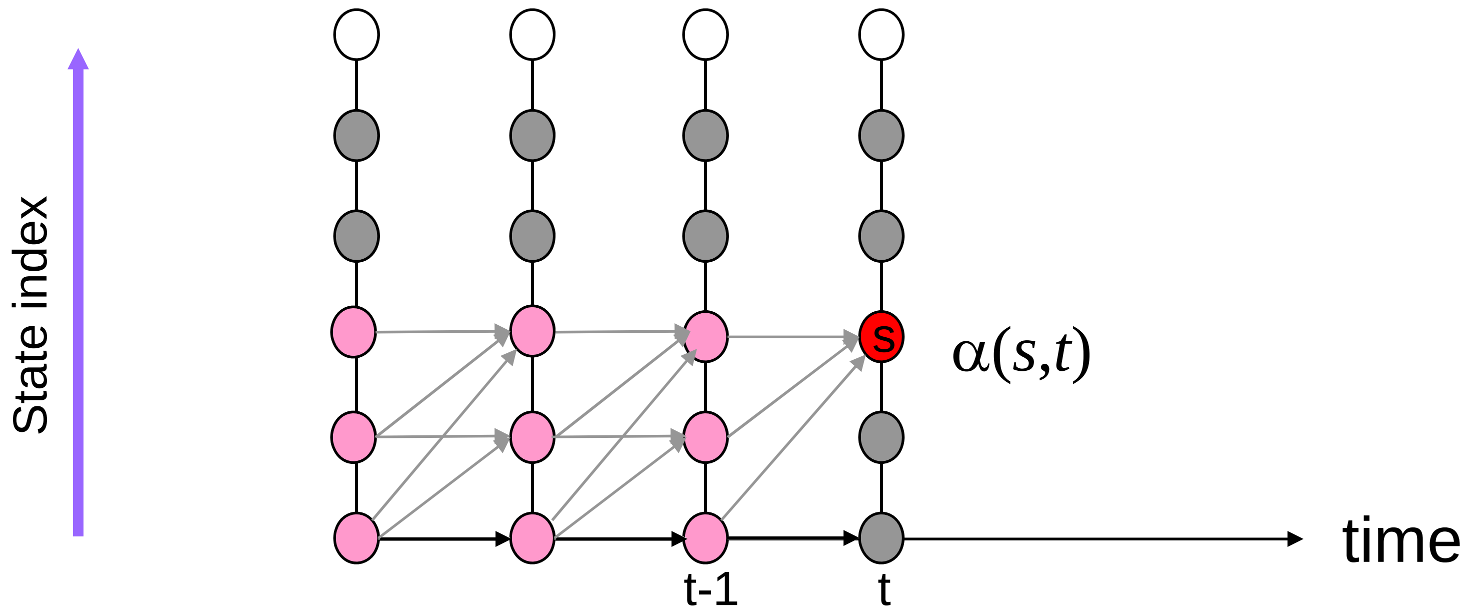
Diversion: The Trellis



- The trellis is a graphical representation of all possible paths through the HMM to produce a given observation
- The Y-axis represents HMM states, X axis represents observations
- Every edge in the graph represents a valid transition in the HMM over a single time step
- Every node represents the event of a particular observation being generated from a particular state

The Forward Algorithm

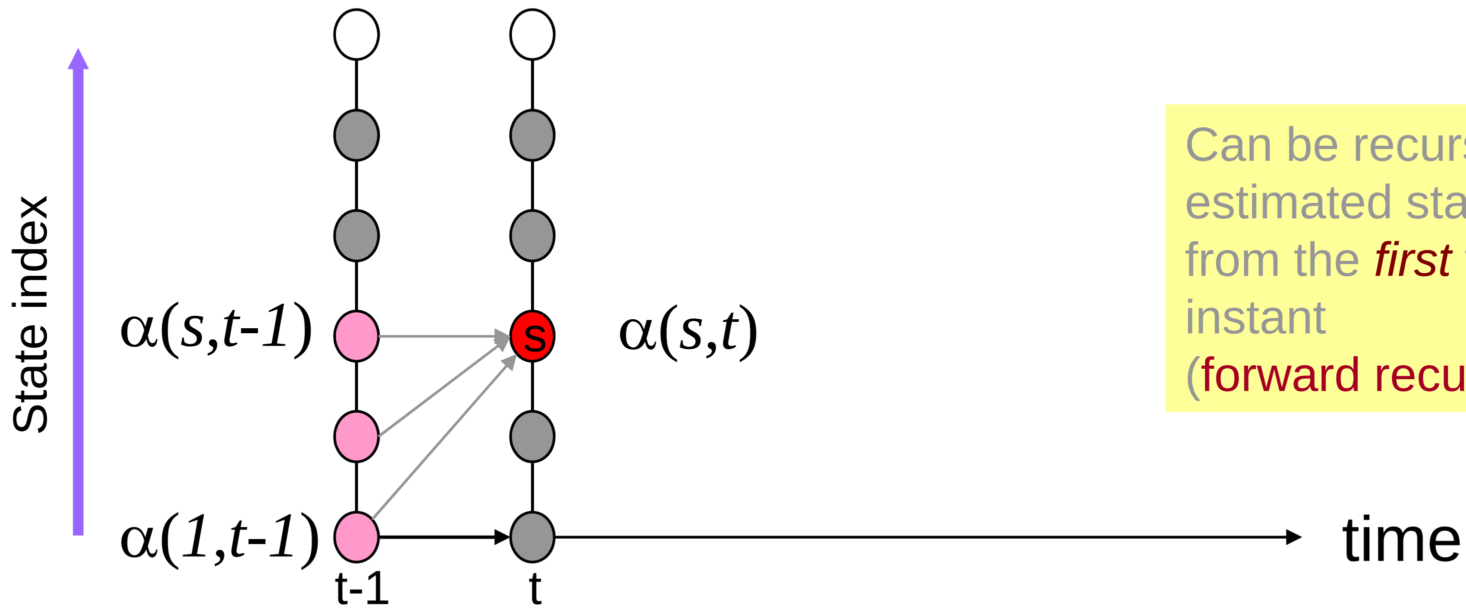
$$\alpha(s,t) = P(x_1, x_2, ..., x_t, state(t) = s)$$



- $\alpha(s,t)$ is the total probability of ALL state sequences that end at state *s* at time *t*, and all observations until x_t

The Forward Algorithm

$$\alpha(s,t) = P(x_1, x_2, \ldots, x_t, state(t) = s)$$



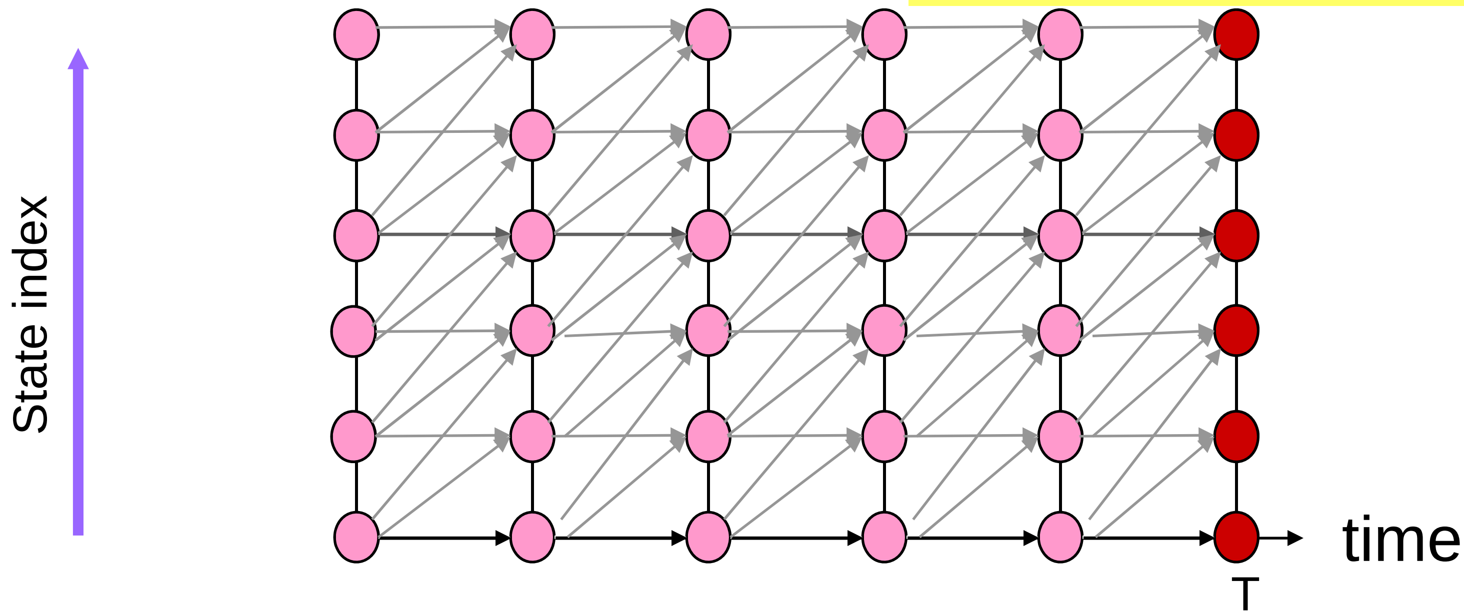
Can be recursively estimated starting from the *first* time instant (forward recursion)

$$\alpha(s,t) = \sum_{s'} \alpha(s',t-1) P(s \mid s') P(x_t \mid s)$$

- $\alpha(s,t)$ can be recursively computed in terms of $\alpha(s',t')$, the forward probabilities at time t-1

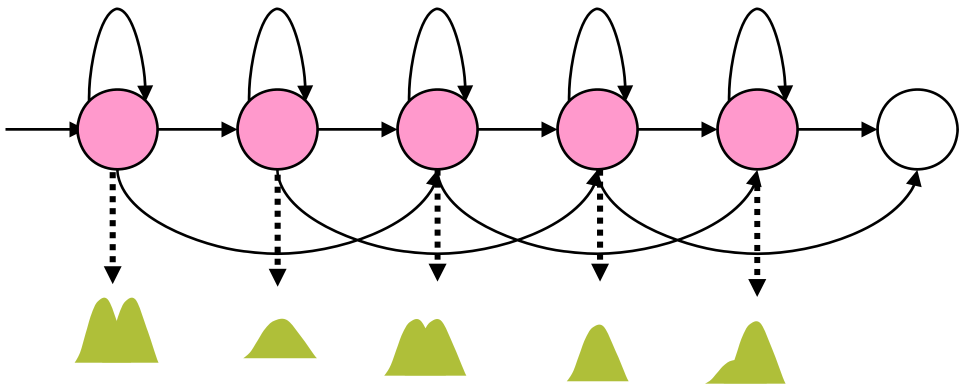
The Forward Algorithm

$$Totalprob = \sum_{s} \alpha(s,T)$$



- In the final observation the alpha at each state gives the probability of all state sequences ending at that state
- General model: The total probability of the observation is the sum of the alpha values at all states

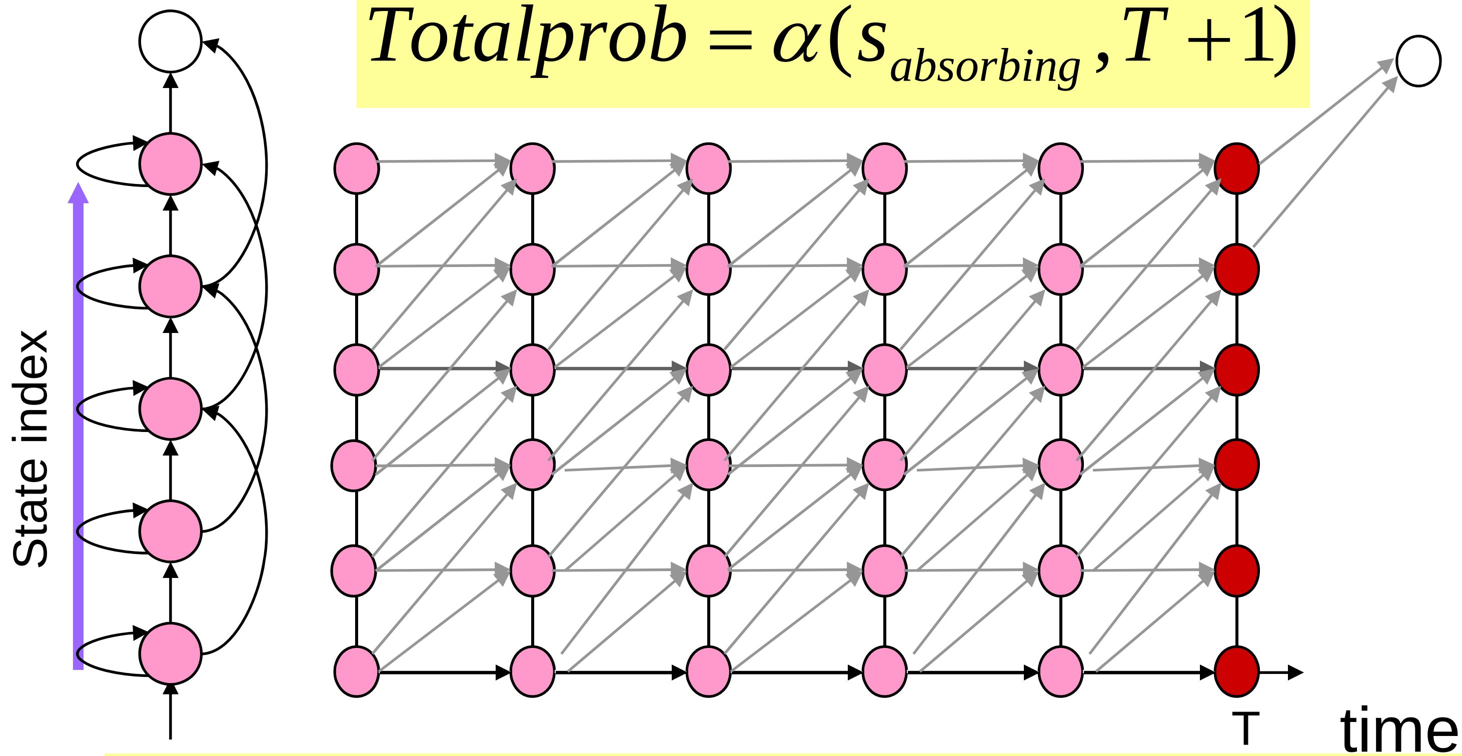
The absorbing state



- Observation sequences are assumed to end only when the process arrives at an absorbing state
  - No observations are produced from the absorbing state

The Forward Algorithm

$$Totalprob = \alpha(s_{absorbing}, T+1)$$



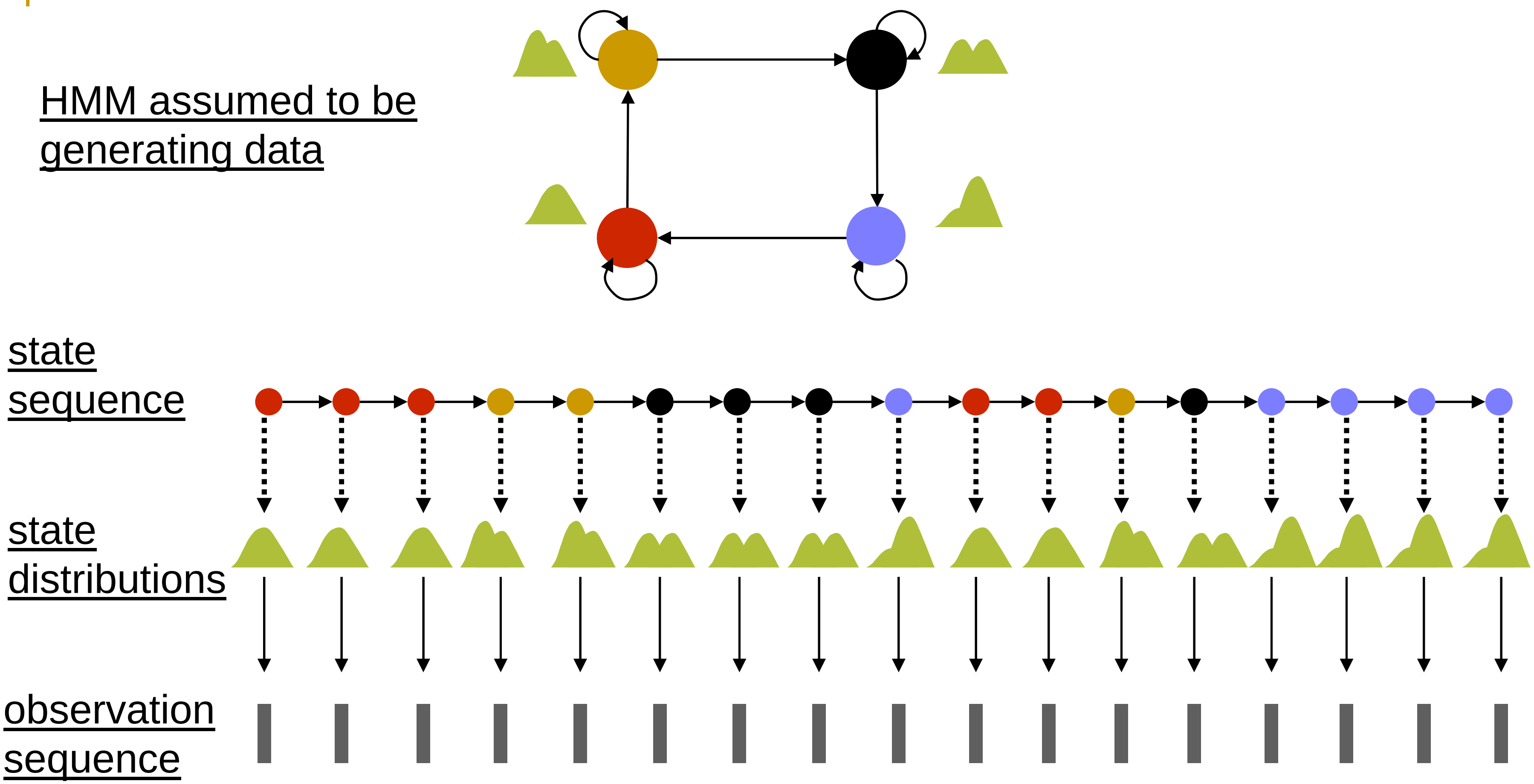
$$\alpha(s_{absorbing}, T+1) = \sum_{s'} \alpha(s', T) P(s_{absorbing} \mid s')$$

- Absorbing state model: The total probability is the alpha computed at the absorbing state after the final observation

Problem 2: State segmentation

- Given only a sequence of observations, how do we determine which sequence of states was followed in producing it?

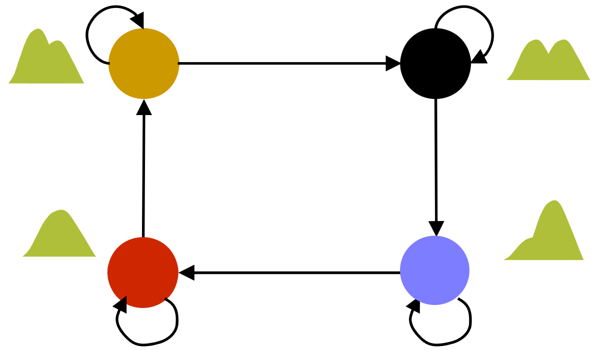
The HMM as a generator



- The process goes through a series of states and produces observations from them

States are hidden

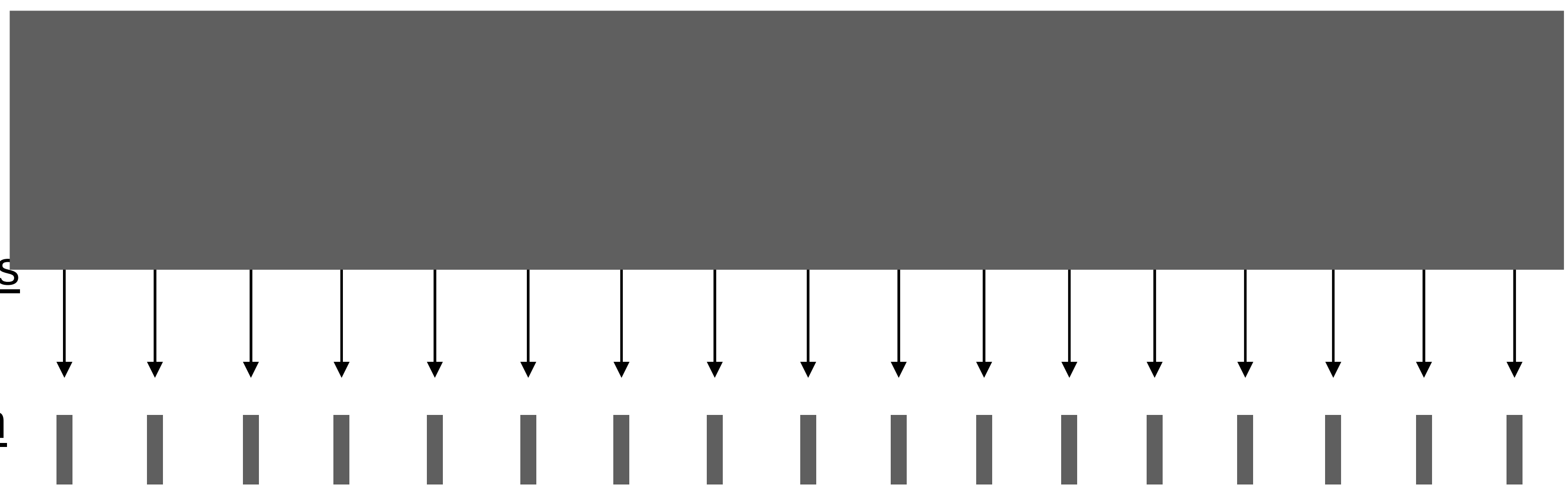
HMM assumed to be generating data



state sequence

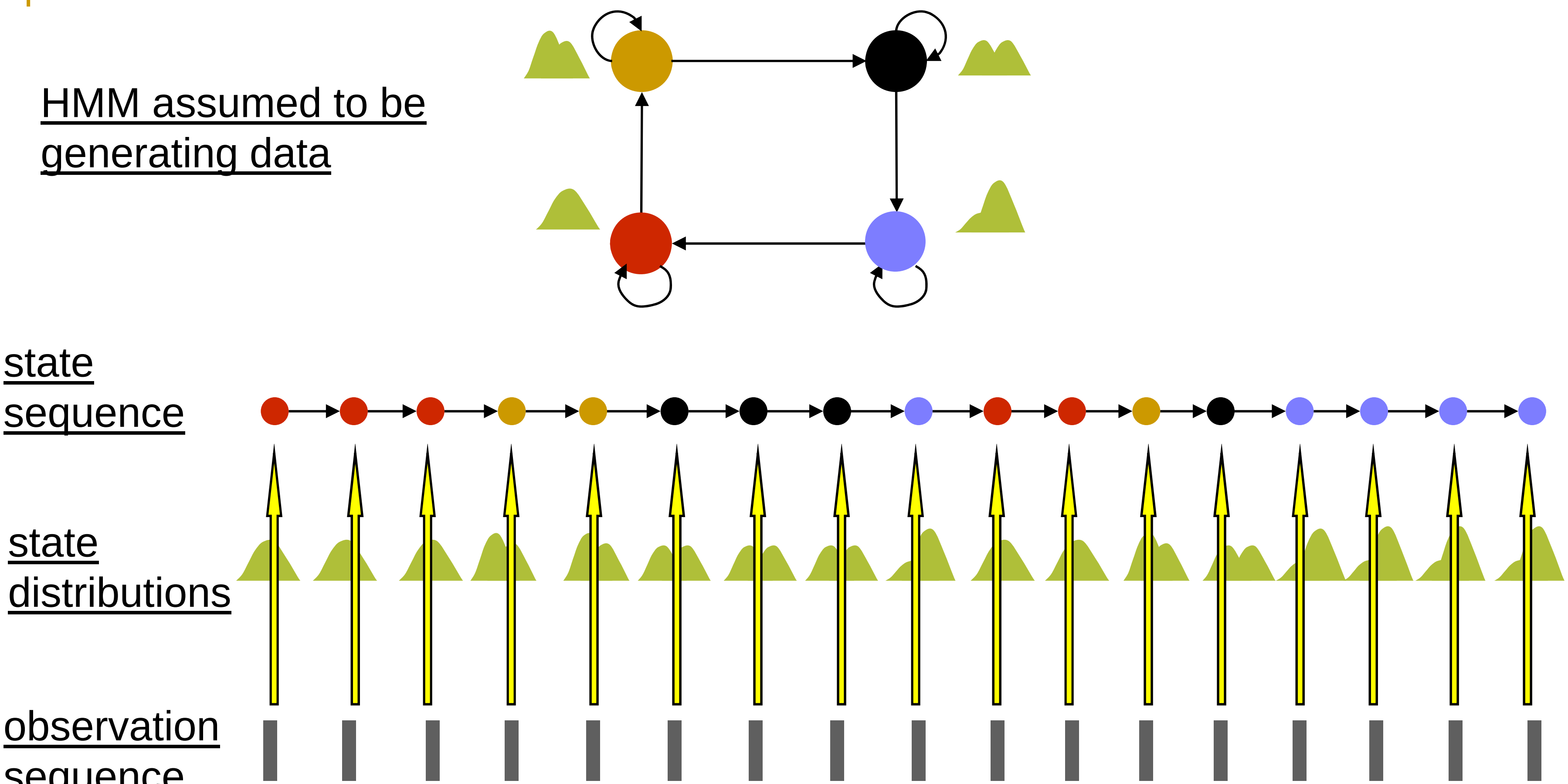
state distributions

observation sequence



- The observations do not reveal the underlying state

The state segmentation problem



- State segmentation: Estimate state sequence given observations

Estimating the State Sequence

- Many different state sequences are capable of producing the observation
- Solution: Identify the most *probable* state sequence
  - The state sequence for which the probability of progressing through that sequence and generating the observation sequence is maximum
  - i.e $P(o_1, o_2, o_3, \ldots, s_1, s_2, s_3, \ldots)$ is maximum

Estimating the state sequence

- Once again, exhaustive evaluation is impossibly expensive
- But once again a simple dynamic-programming solution is available

$$P(o_1, o_2, o_3, \ldots, s_1, s_2, s_3, \ldots) =$$

$$\underline{P(o_1|s_1)P(o_2|s_2)P(o_3|s_3)\ldots}\;\underline{P(s_1)P(s_2|s_1)P(s_3|s_2)\ldots}$$

- Needed:

$$\arg\max_{s_1,s_2,s_3,\ldots} P(o_1 \mid s_1)P(s_1)P(o_2 \mid s_2)P(s_2 \mid s_1)P(o_3 \mid s_3)P(s_3 \mid s_2)$$

Estimating the state sequence

- Once again, exhaustive evaluation is impossibly expensive
- But once again a simple dynamic-programming solution is available

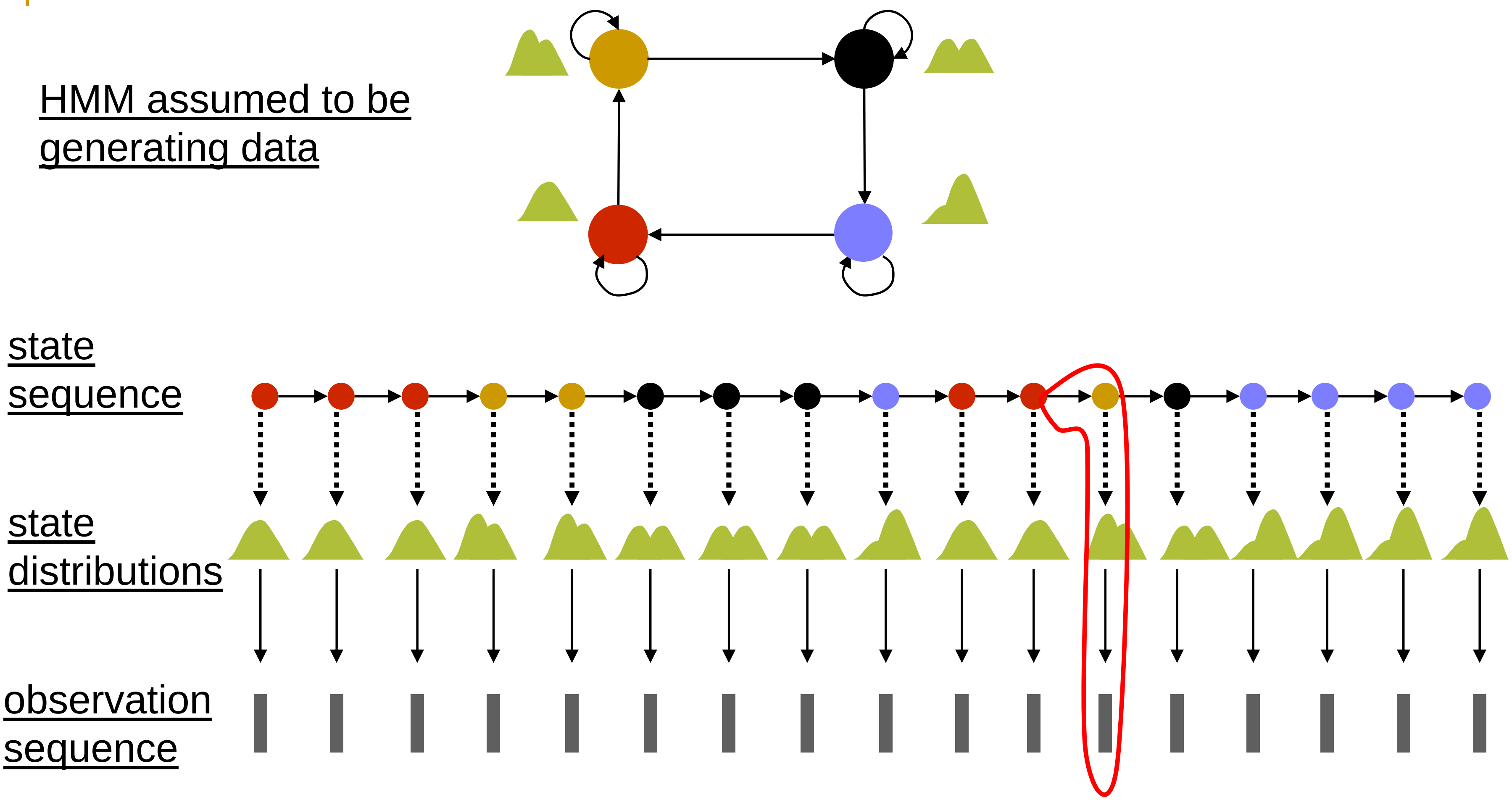
$$P(o_1, o_2, o_3, \ldots, s_1, s_2, s_3, \ldots) =$$

$$\underline{P(o_1|s_1)P(o_2|s_2)P(o_3|s_3)\ldots}\ \underline{P(s_1)P(s_2|s_1)P(s_3|s_2)\ldots}$$

- Needed:

$$\arg\max_{s_1, s_2, s_3, \ldots} P(o_1 \mid s_1)P(s_1)P(o_2 \mid s_2)P(s_2 \mid s_1)P(o_3 \mid s_3)P(s_3 \mid s_2)$$

The HMM as a generator



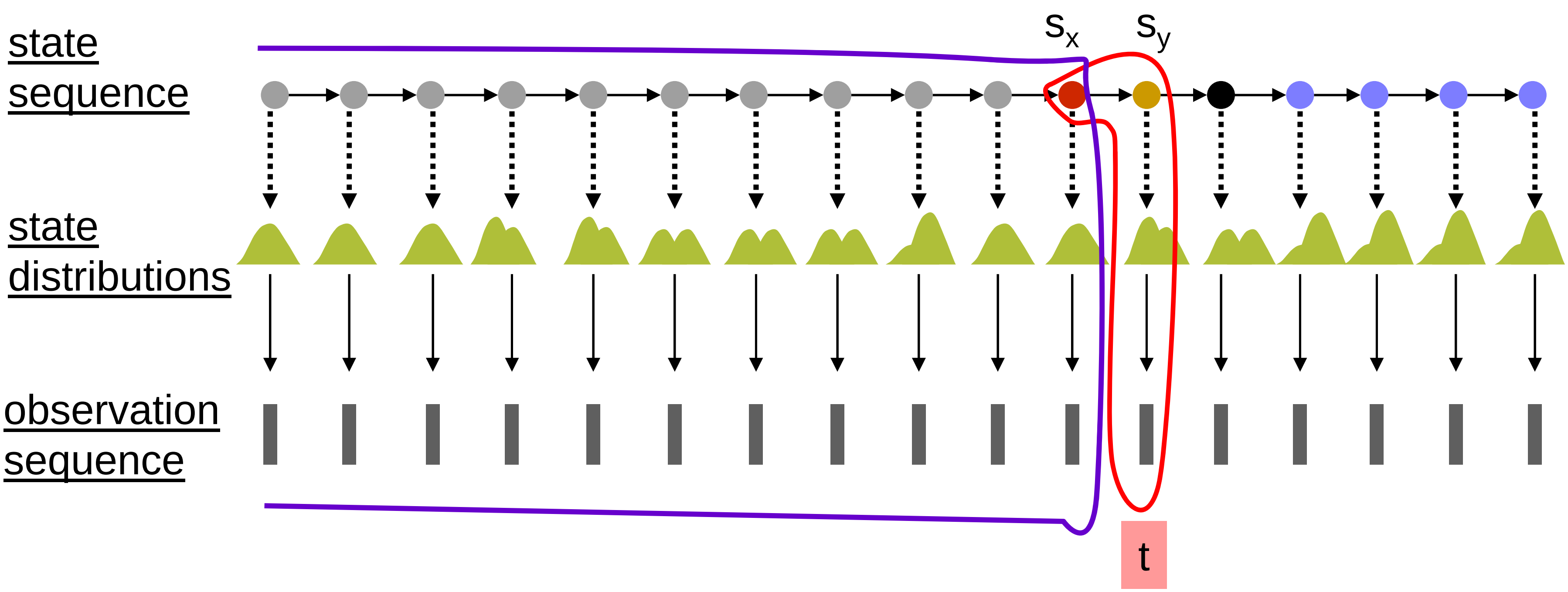
- Each enclosed term represents one forward transition and a subsequent emission

The state sequence

- The probability of a state sequence ?,?,?,?,s_x,s_y ending at time t , and producing all observations until o_t
  - $P(o_{1..t-1}, ?,?,?,?, s_x , o_t,s_y) = P(o_{1..t-1},?,?,?,?, s_x) \underline{P(o_t|s_y)P(s_y|s_x)}$

- The *best* state sequence that ends with s_x,s_y at t will have a probability equal to the probability of the best state sequence ending at $t\text{-}1$ at s_x times $P(o_t|s_y)P(s_y|s_x)$

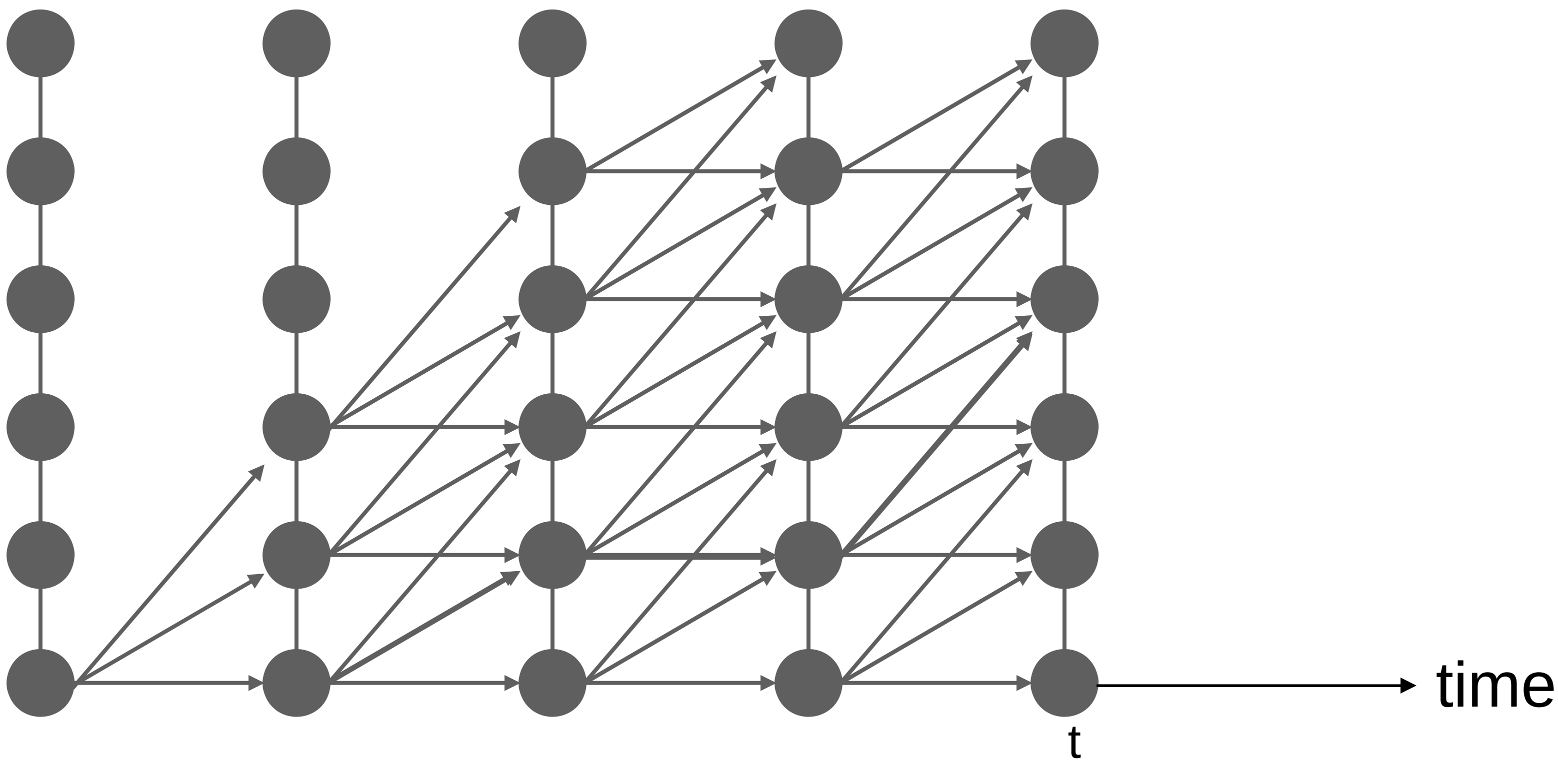
Extending the state sequence



- The probability of a state sequence ?,?,?,?,s_x,s_y ending at time *t* and producing observations until o_t
  - $P(o_{1..t-1}, o_t, ?,?,?,?, s_x, s_y) = P(o_{1..t-1}, ?,?,?,?, s_x)P(o_t|s_y)P(s_y|s_x)$

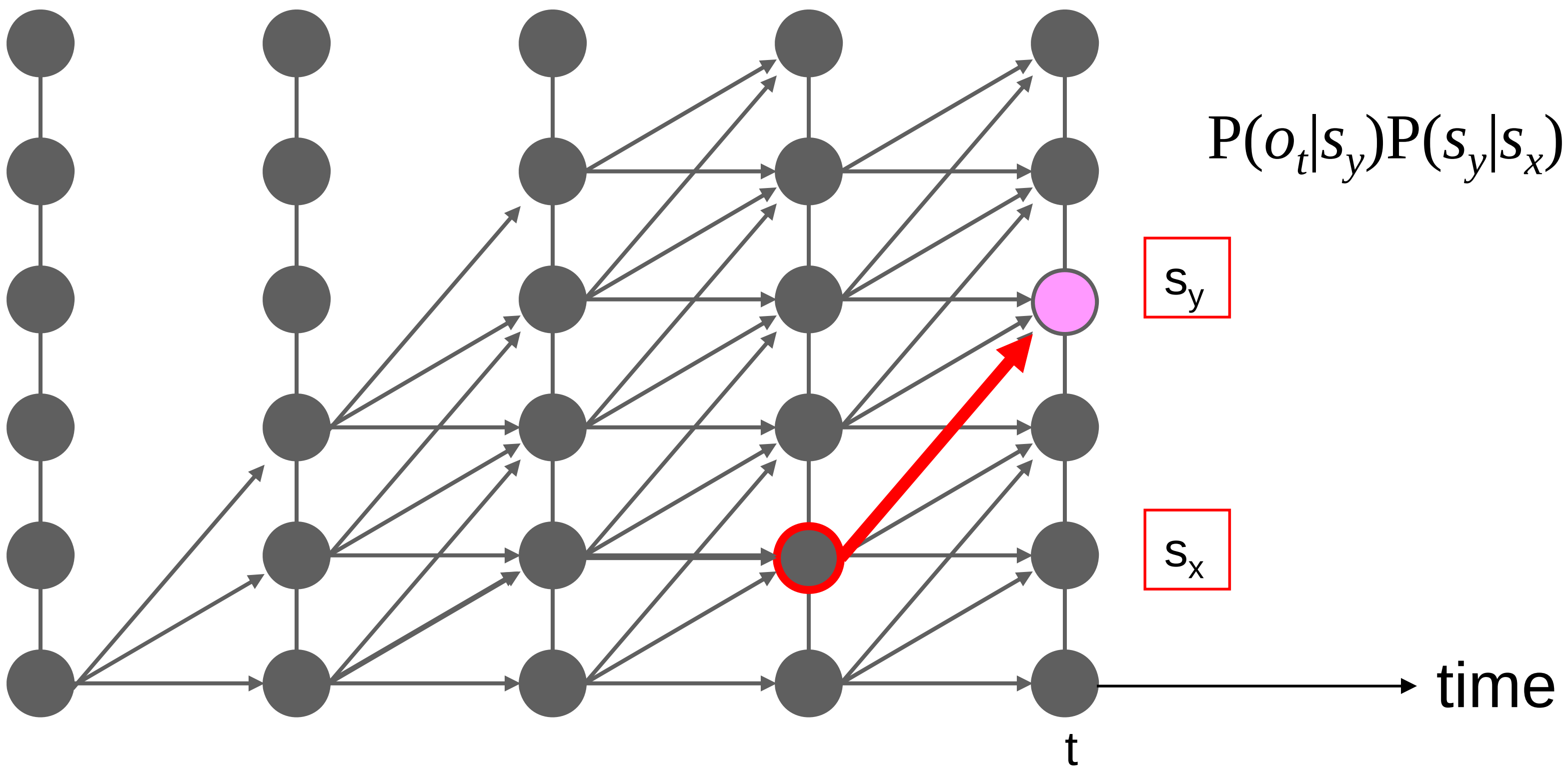
Trellis

- The graph below shows the set of all possible state sequences through this HMM in five time instants



The cost of extending a state sequence

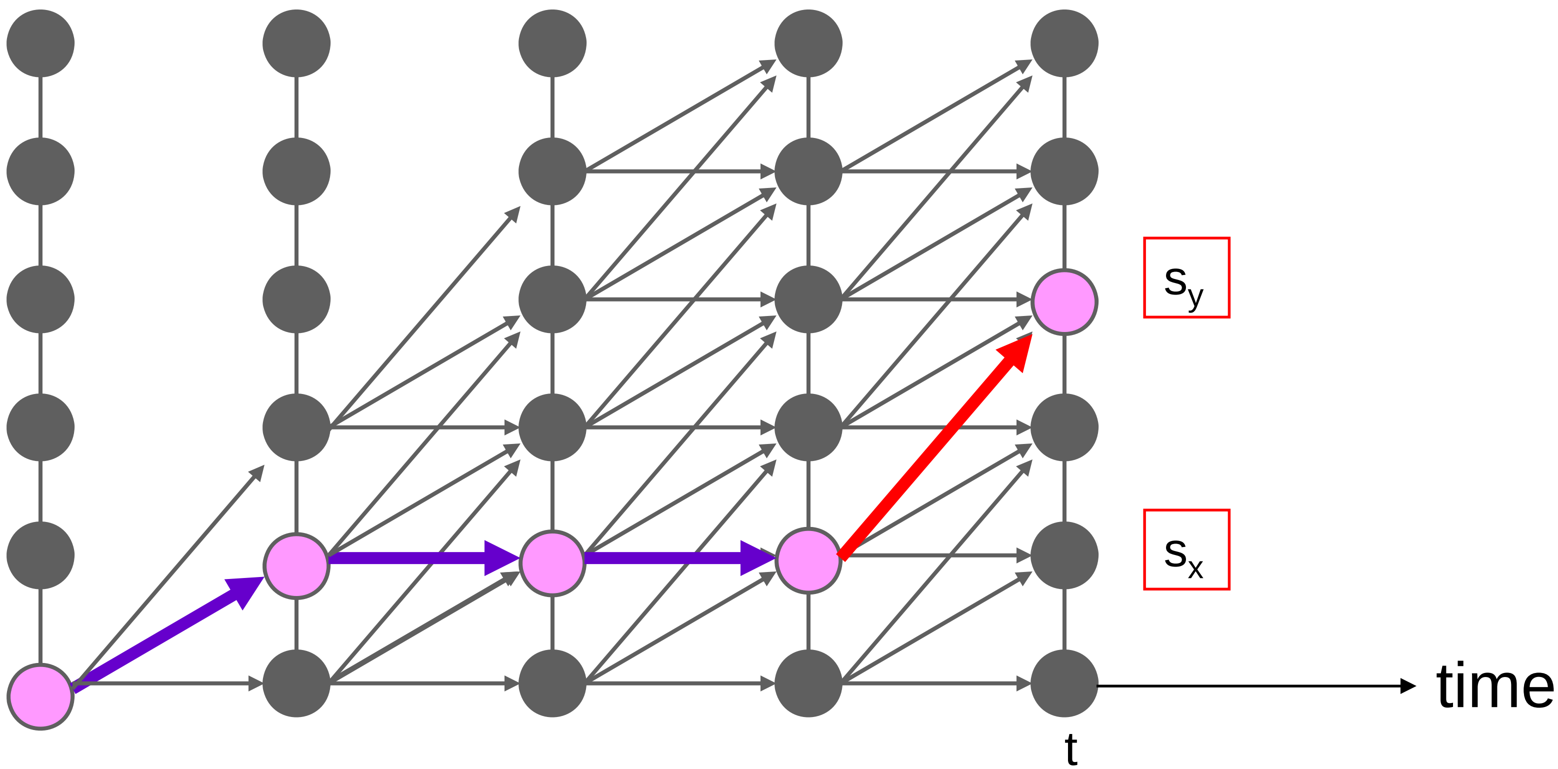
- The cost of *extending* a state sequence ending at s_x is only dependent on the transition from s_x to s_y, and the observation probability at s_y



The cost of extending a state sequence

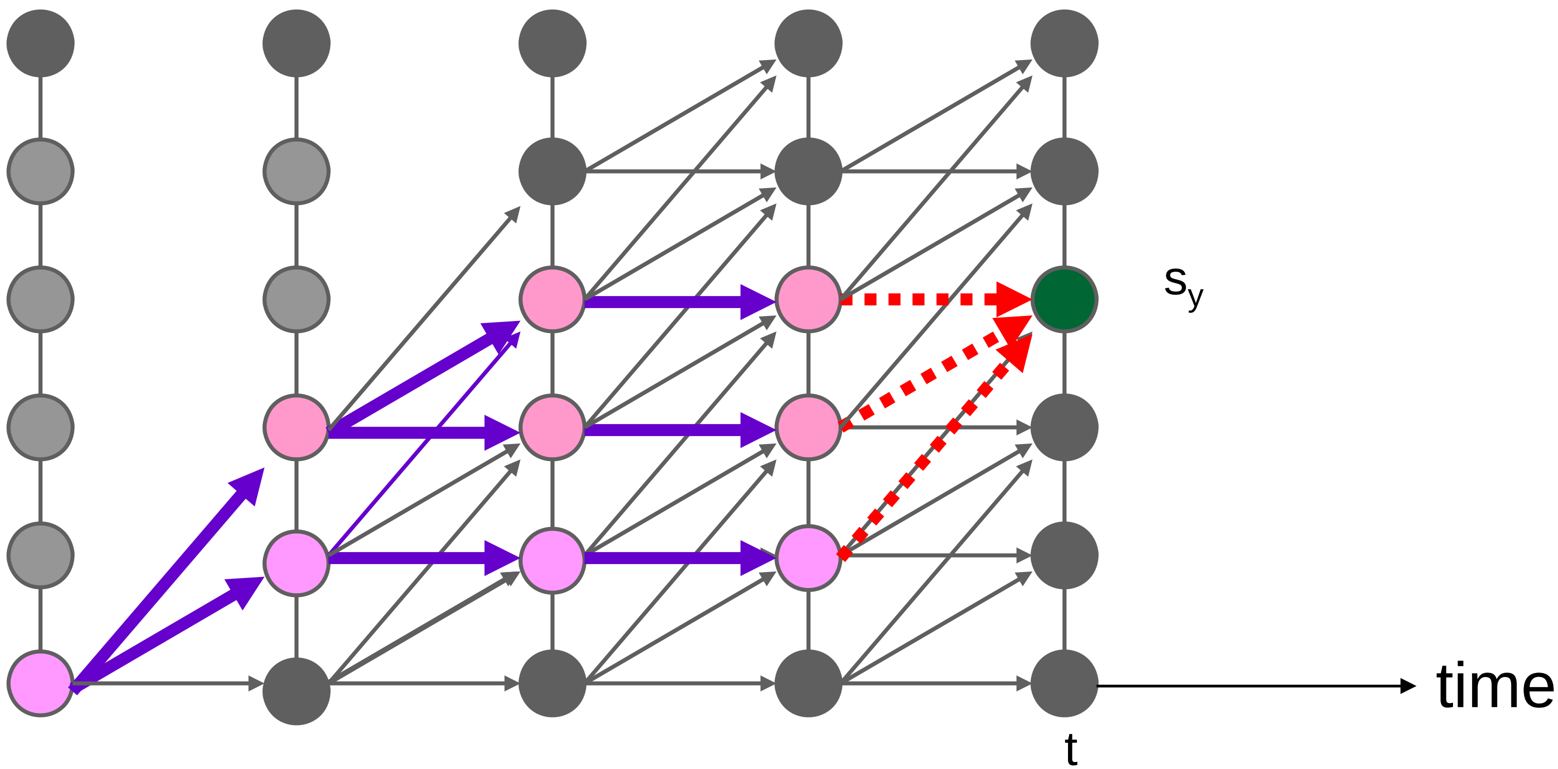
- The best path to s_y through s_x is simply an extension of the best path to s_x

BestP($o_{1..t-1}$,?,?,?,?, s_x)
$P(o_t|s_y)P(s_y|s_x)$



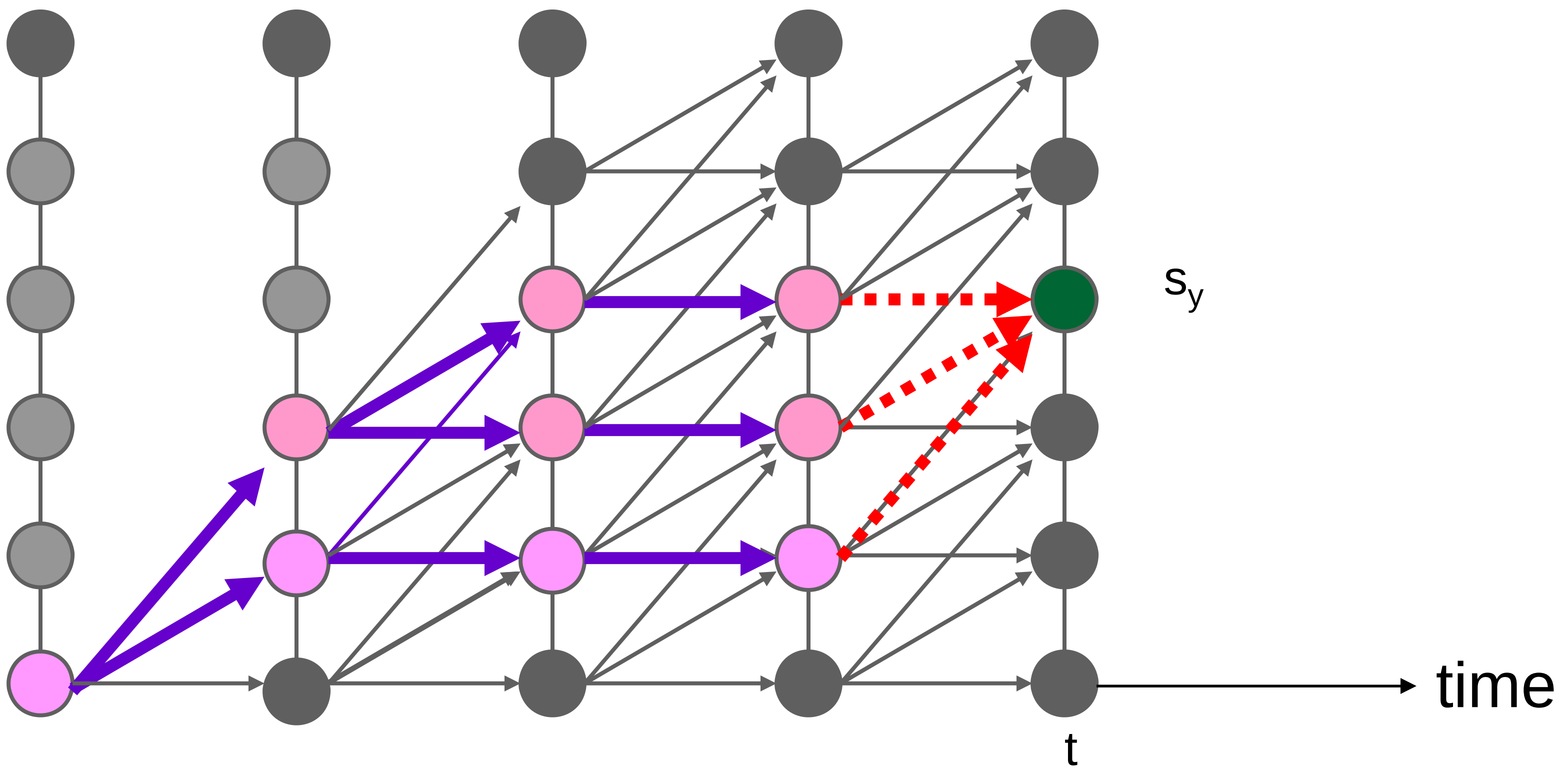
The Recursion

- The overall best path to s_y is an extension of the best path to one of the states at the previous time



The Recursion

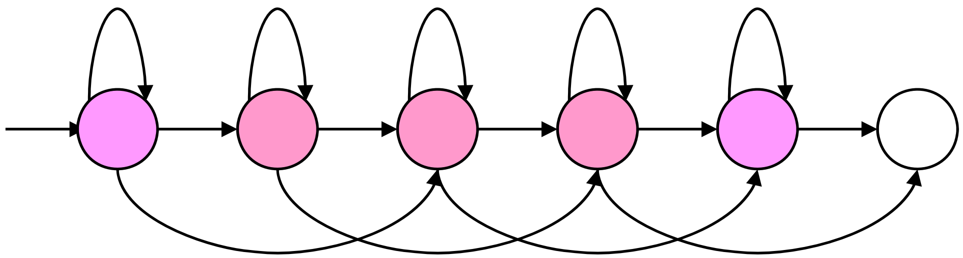
- Prob. of best path to s_y =
  Max_{s_x} $\text{BestP}(o_{1..t-1},?,?,?,?, s_x)\ P(o_t|s_y)P(s_y|s_x)$

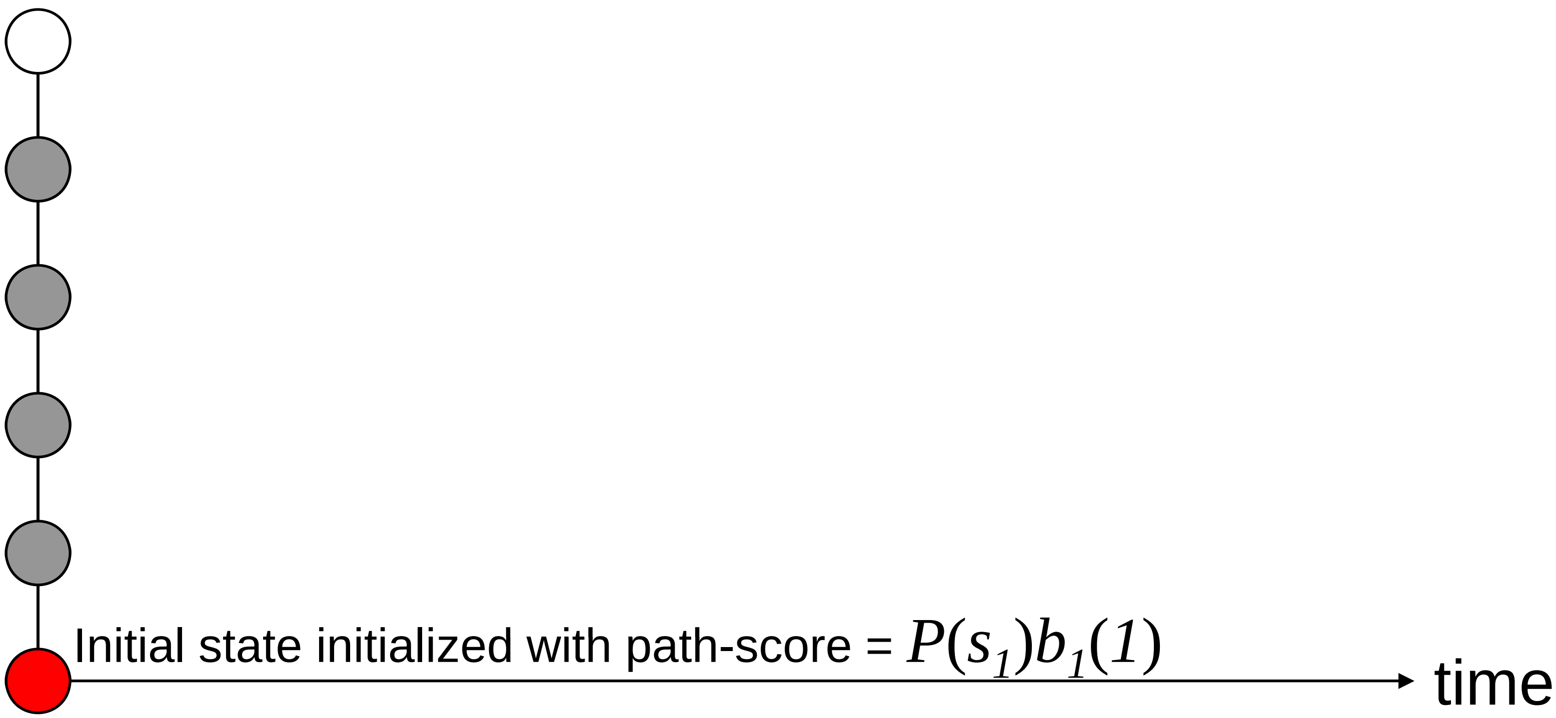


Finding the best state sequence

- The simple algorithm just presented is called the VITERBI algorithm in the literature
  - After A.J.Viterbi, who invented this dynamic programming algorithm for a completely different purpose: decoding error correction codes!

Viterbi Search (contd.)

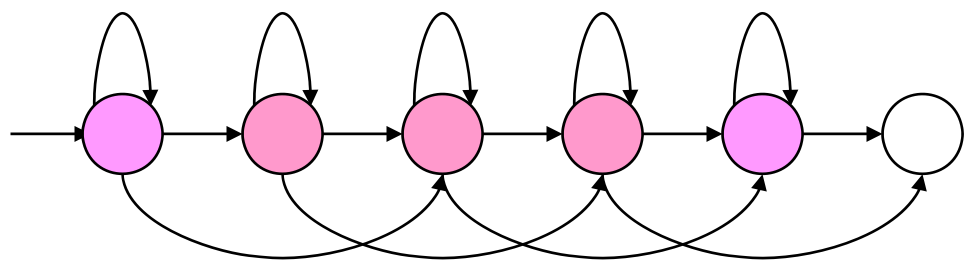


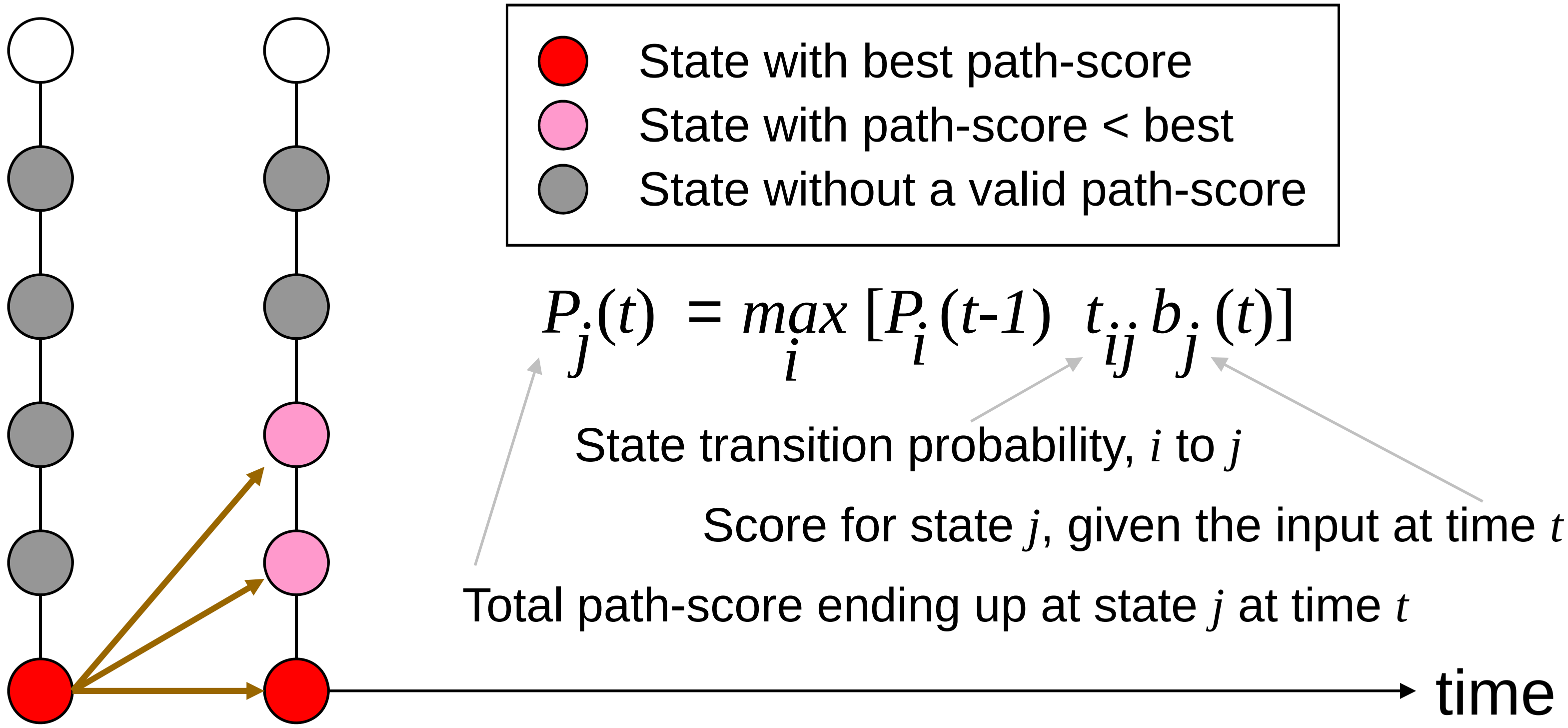


Initial state initialized with path-score = $P(s_1)b_1(1)$

All other states have score 0 since $P(s_i) = 0$ for them

Viterbi Search (contd.)





- State with best path-score
- State with path-score < best
- State without a valid path-score

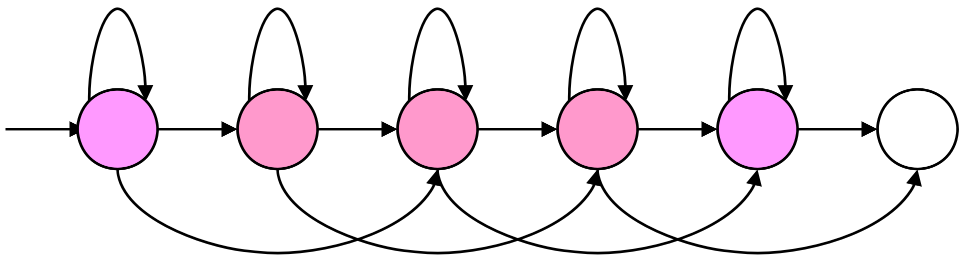
$$P_j(t) = \max_i \left[P_i(t-1)\ t_{ij}\ b_j(t)\right]$$

State transition probability, i to j

Score for state j, given the input at time t

Total path-score ending up at state j at time t

Viterbi Search (contd.)

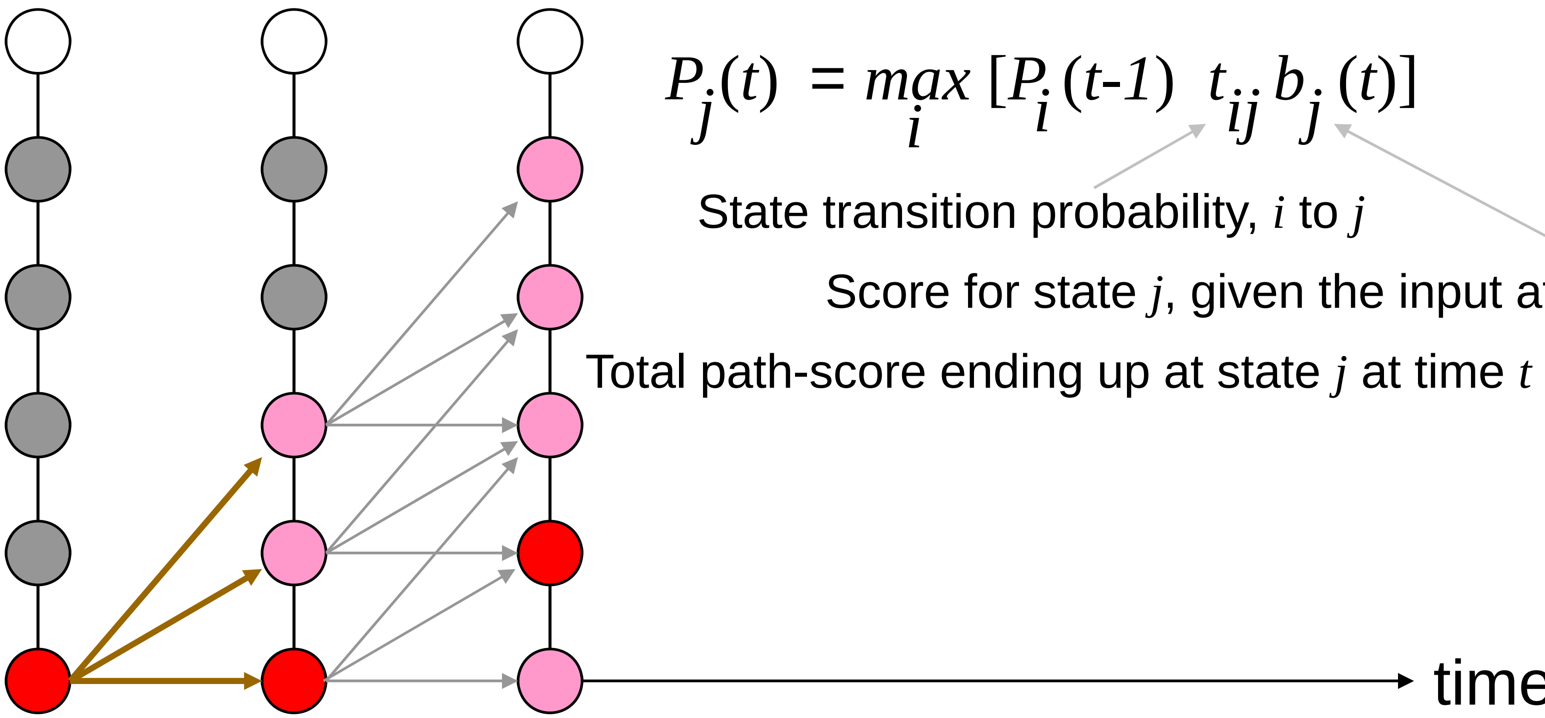


$$P_j(t) = \max_i [P_i(t\text{-}1)\ t_{ij}\ b_j(t)]$$

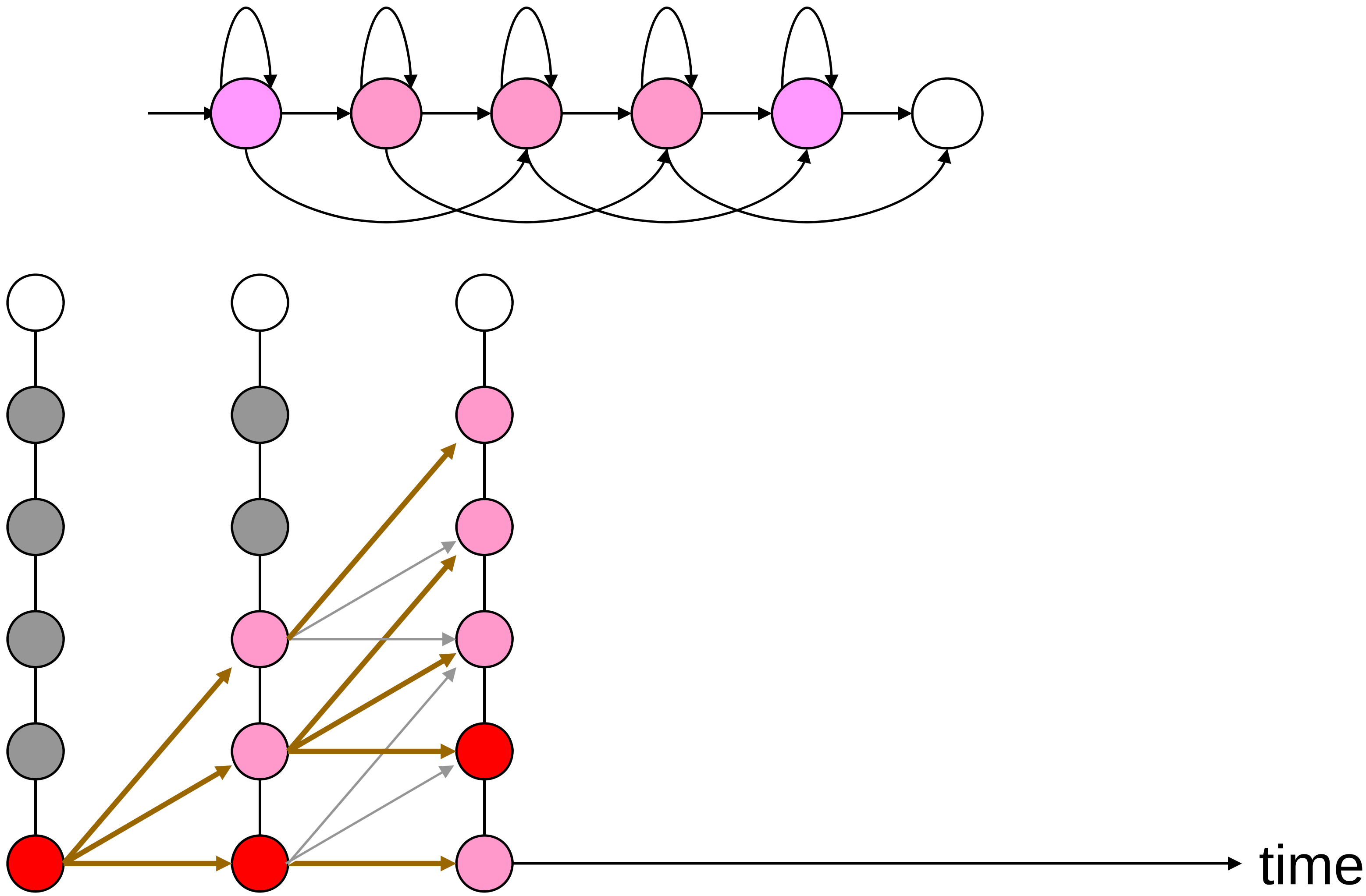
State transition probability, i to j

Score for state j, given the input at time t

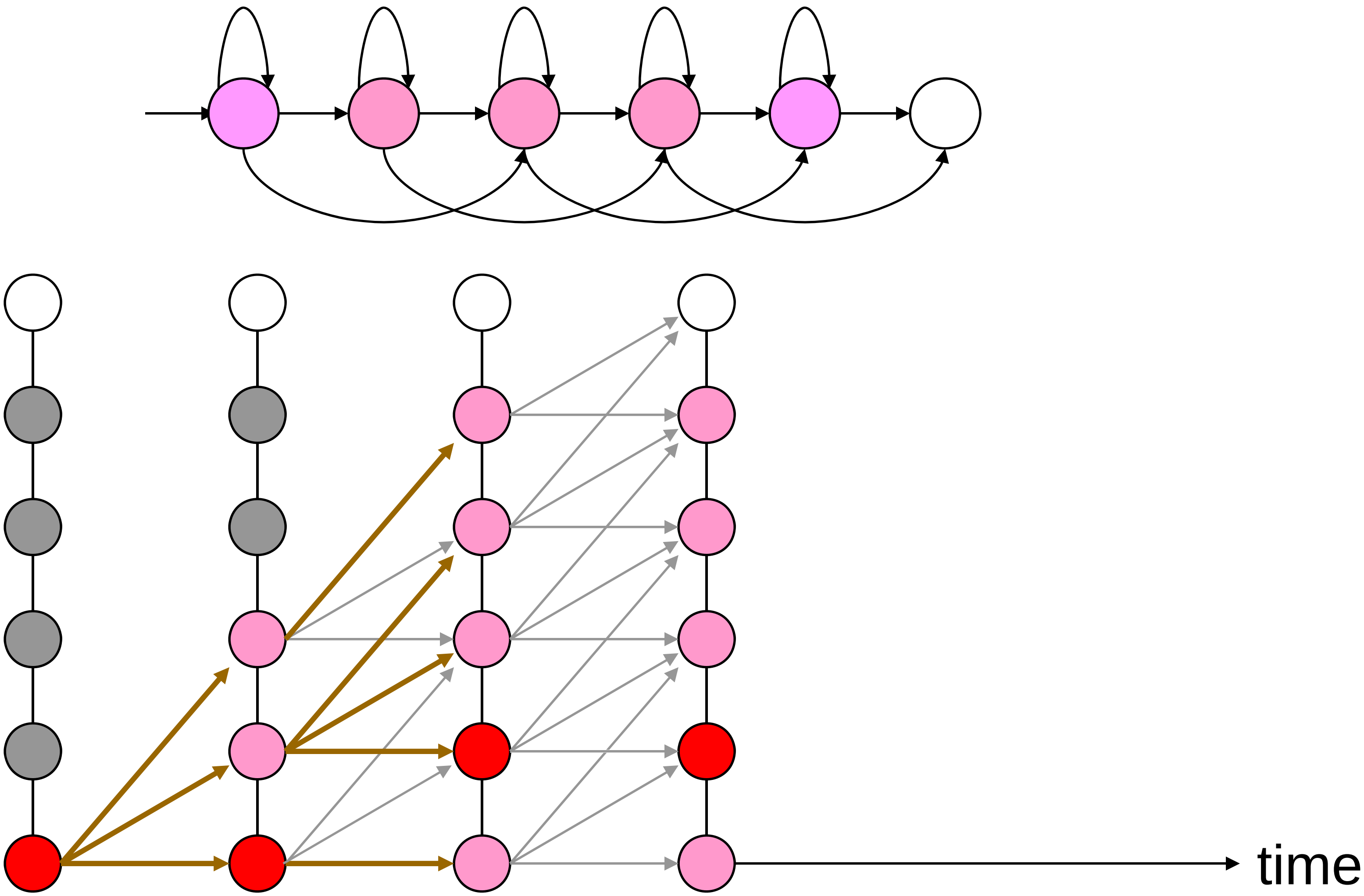
Total path-score ending up at state j at time t



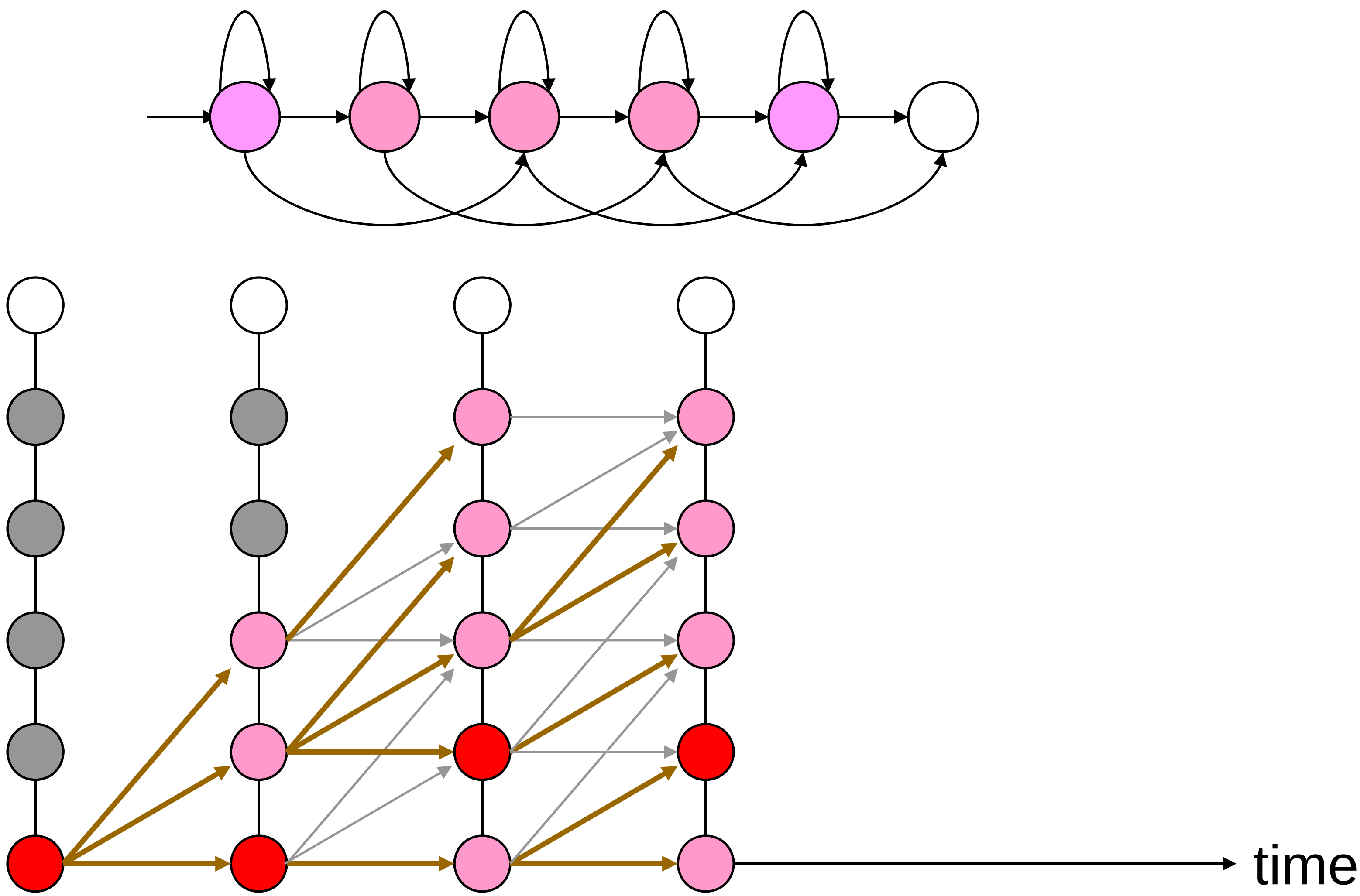
Viterbi Search (contd.)



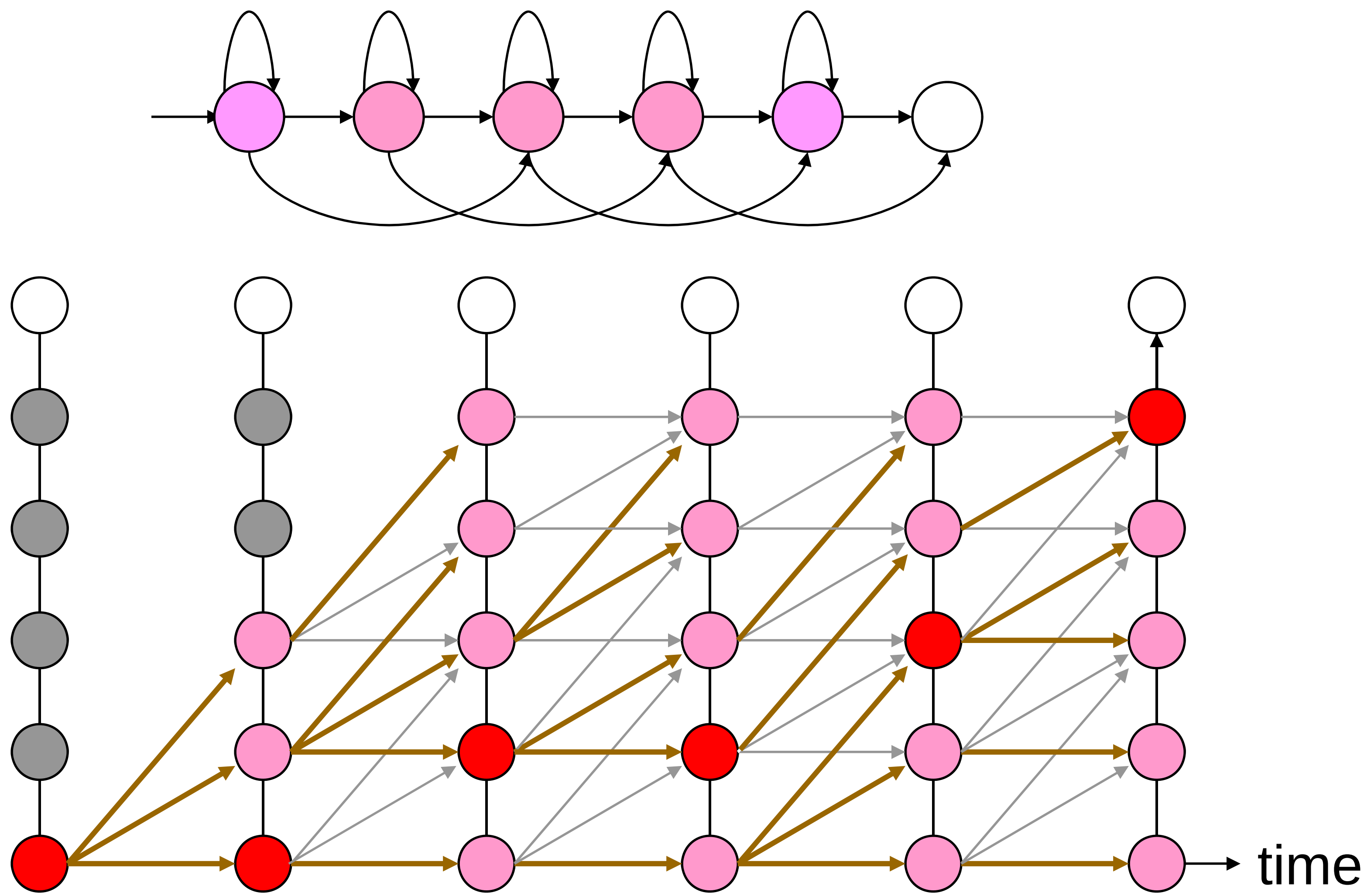
Viterbi Search (contd.)



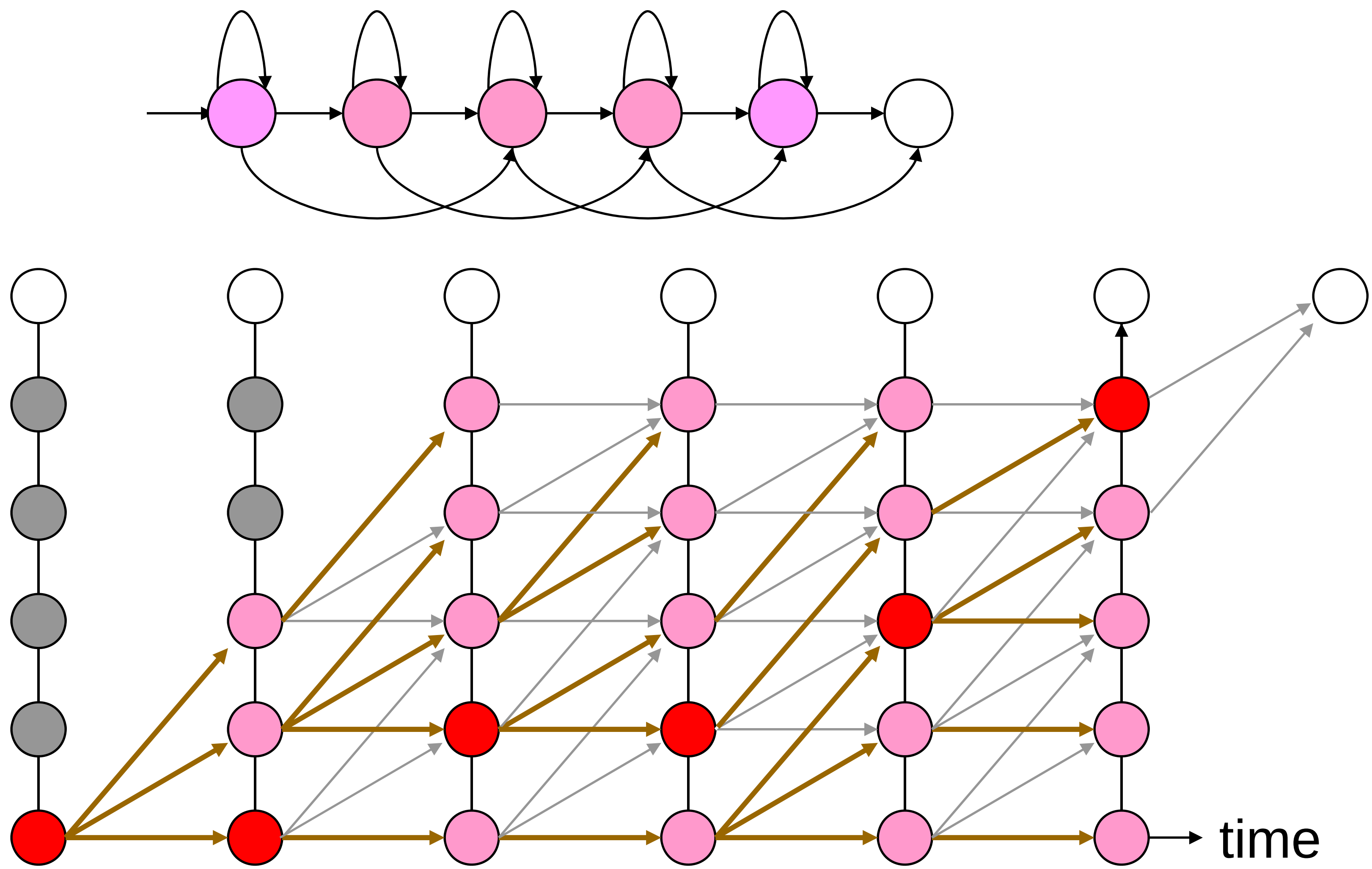
Viterbi Search (contd.)



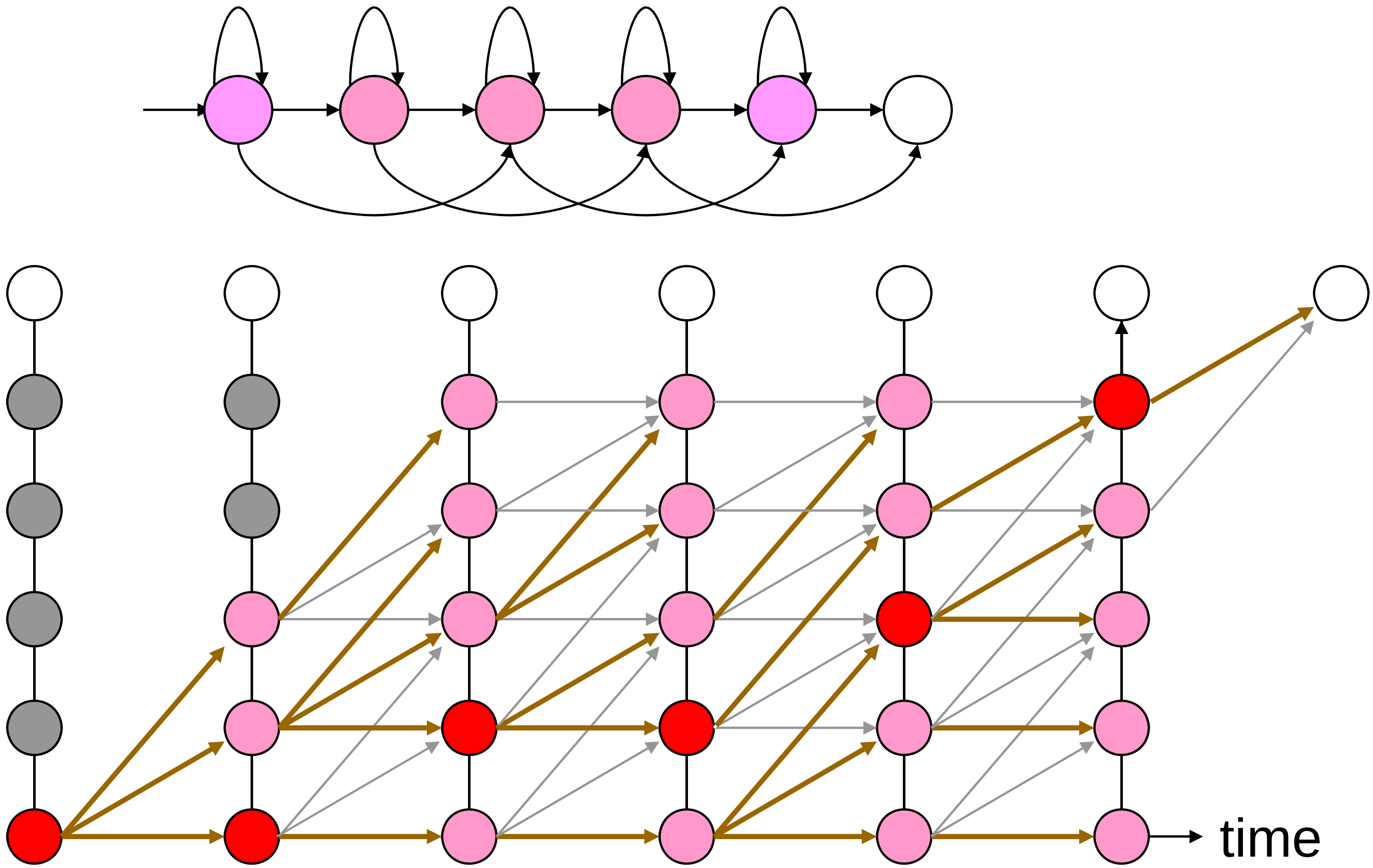
Viterbi Search (contd.)



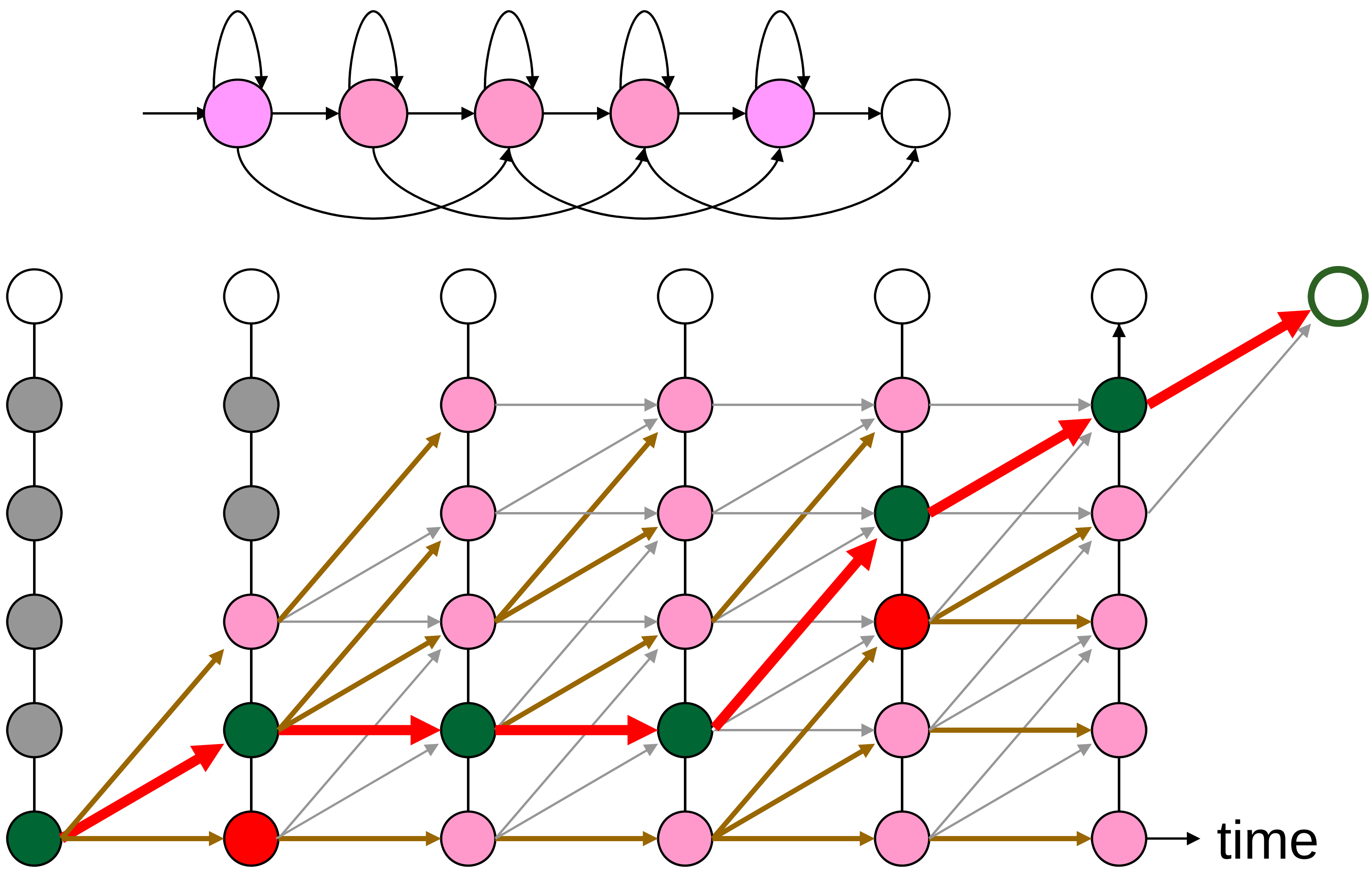
Viterbi Search (contd.)



Viterbi Search (contd.)

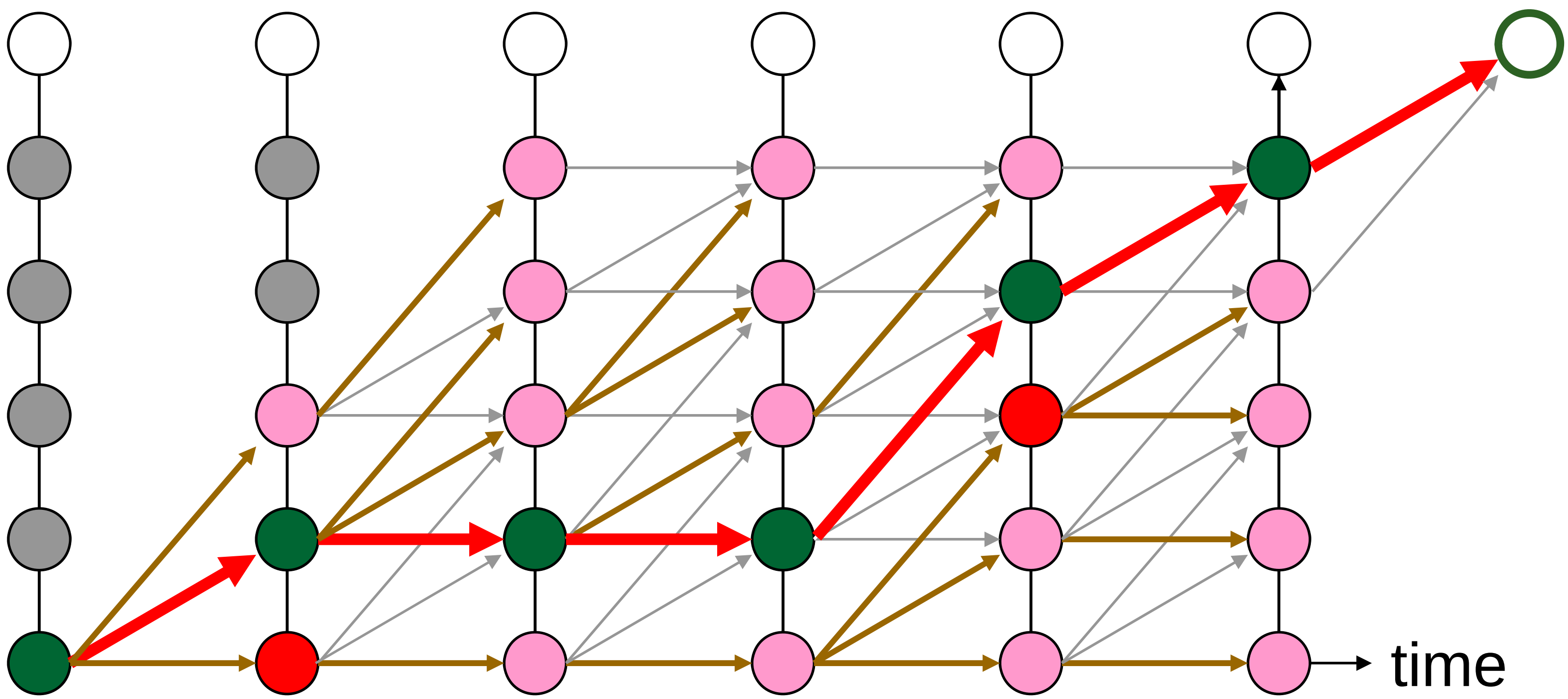


Viterbi Search (contd.)



Viterbi Search (contd.)

THE BEST STATE SEQUENCE IS THE ESTIMATE OF THE STATE SEQUENCE FOLLOWED IN GENERATING THE OBSERVATION



Problem3: Training HMM parameters

- We can compute the probability of an observation, and the best state sequence given an observation, using the HMM's parameters
- But where do the HMM parameters come from?
- They must be learned from a collection of observation sequences

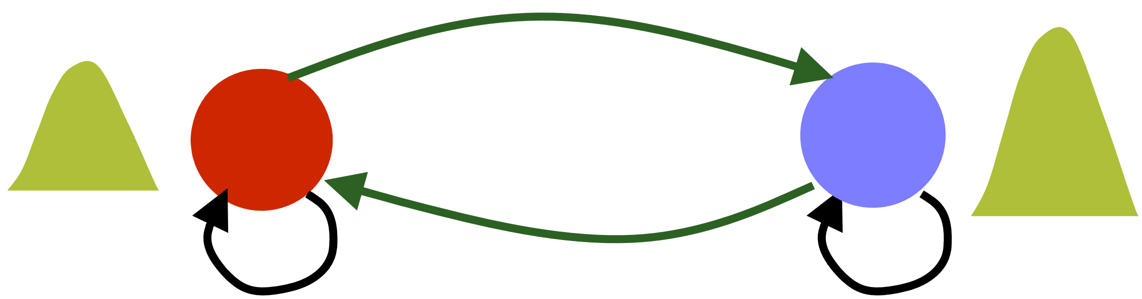
Learning HMM parameters: Simple procedure – counting

- Given a set of training instances
- Iteratively:

1. Initialize HMM parameters
2. Segment all training instances
3. Estimate transition probabilities and state output probability parameters by counting

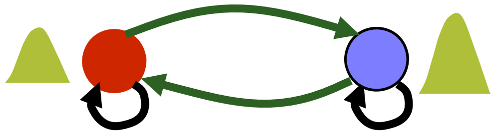
Learning by counting example

- Explanation by example in next few slides
- 2-state HMM, Gaussian PDF at states, 3 observation sequences
- Example shows ONE iteration
  - How to count after state sequences are obtained



Example: Learning HMM Parameters

- We have an HMM with two states s1 and s2.
- Observations are vectors x_{ij}
  - i-th sequence, j-th vector
- We are given the following three observation sequences
  - And have already estimated state sequences



Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

Observation 2

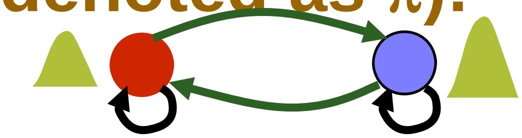
Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters

- **Initial state probabilities (usually denoted as π):**
  - We have 3 observations
  - 2 of these begin with S1, and one with S2
  - $\pi(S1) = 2/3$, $\pi(S2) = 1/3$



Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

Observation 2

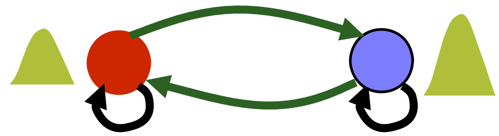
Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters

- **Transition probabilities:**
  - State S1 occurs 11 times in non-terminal locations



Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

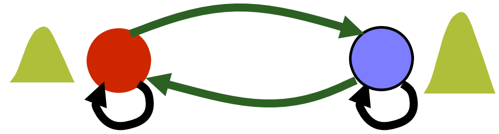
Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters



- **Transition probabilities:**
  - State S1 occurs 11 times in non-terminal locations
  - Of these, it is followed immediately by S1 6 times

1

Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

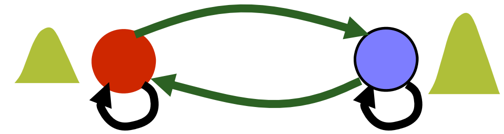
Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters



- **Transition probabilities:**
  - State S1 occurs 11 times in non-terminal locations
  - Of these, it is followed immediately by S1 6 times
  - It is followed immediately by S2 5 times

Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

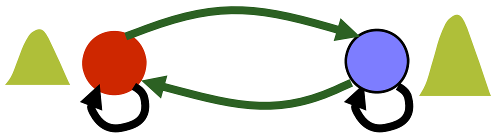
Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters



- **Transition probabilities:**
  - State S1 occurs 11 times in non-terminal locations
  - Of these, it is followed immediately by S1 6 times
  - It is followed immediately by S2 5 times
  - P(S1 | **S1**) = 6/ 11;   P(S2 | **S1**) = 5 / 11

Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

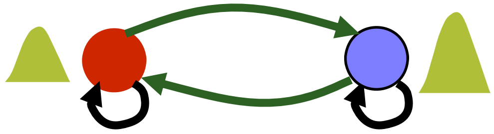
Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters



- **Transition probabilities:**
  - State S2 occurs 13 times in non-terminal locations

Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs.	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

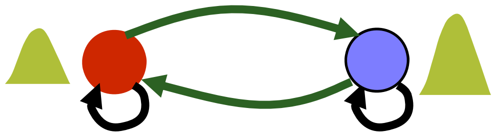
Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters



- **Transition probabilities:**
  - State S2 occurs 13 times in non-terminal locations
  - Of these, it is followed immediately by S1 5 times

Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

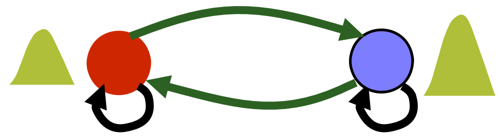
Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters



- **Transition probabilities:**
  - State S2 occurs 13 times in non-terminal locations
  - Of these, it is followed immediately by S1 5 times
  - It is followed immediately by S2 8 times

Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

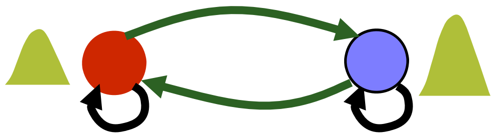
Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters



- **Transition probabilities:**
  - State S2 occurs 13 times in non-terminal locations
  - Of these, it is followed immediately by S1 5 times
  - It is followed immediately by S2 8 times
  - P(S1 | S2) = 5 / 13;   P(S2 | S2) = 8 / 13

Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

Observation 2

Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Parameters learnt so far

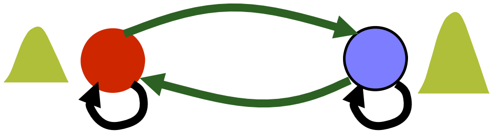
- State initial probabilities, often denoted as π
  - $\pi(S1) = 2/3 = 0.66$
  - $\pi(S2) = 1/3 = 0.33$
- State transition probabilities
  - P(S1 | S1) = 6/11 = 0.545; P(S2 | S1) = 5/11 = 0.455
  - P(S1 | S2) = 5/13 = 0.385; P(S2 | S2) = 8/13 = 0.615
  - Represented as a transition matrix

$$A = \begin{pmatrix} P(S1|S1) & P(S2|S1) \\ P(S1|S2) & P(S2|S2) \end{pmatrix} = \begin{pmatrix} 0.545 & 0.455 \\ 0.385 & 0.615 \end{pmatrix}$$

Each row of this matrix must sum to 1.0

Example: Learning HMM Parameters

- State output probability for S1
  - There are 13 observations in S1



Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

Observation 2

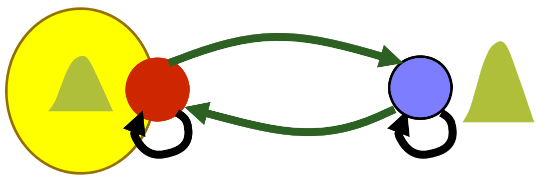
Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters

- State output probability for S1
  - There are 13 observations in S1
  - Segregate them out and count
    - Compute parameters (mean and variance) of Gaussian output density for state S1



Time	1	2	6	7	9	10
state	S1	S1	S1	S1	S1	S1
Obs	X_{a1}	X_{a2}	X_{a6}	X_{a7}	X_{a9}	X_{a10}

$$P(X \mid S_1) = \frac{1}{\sqrt{(2\pi)^d \mid \Theta_1 \mid}} \exp\left(-0.5(X-\mu_1)^T \Theta_1^{-1} (X-\mu_1)\right)$$

Time	3	4	9
state	S1	S1	S1
Obs	X_{b3}	X_{b4}	X_{b9}

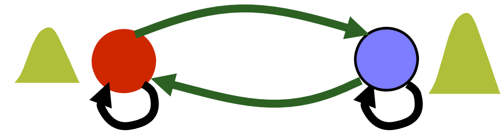
$$\mu_1 = \frac{1}{13}\left(\begin{array}{c} X_{a1} + X_{a2} + X_{a6} + X_{a7} + X_{a9} + X_{a10} + X_{b3} + \\ X_{b4} + X_{b9} + X_{c1} + X_{c2} + X_{c4} + X_{c5} \end{array}\right)$$

Time	1	3	4	5
state	S1	S1	S1	S1
Obs	X_{c1}	X_{c2}	X_{c4}	X_{c5}

$$\Theta_1 = \frac{1}{13}\left(\begin{array}{c} (X_{a1}-\mu_1)(X_{a1}-\mu_1)^T + (X_{a2}-\mu_1)(X_{a2}-\mu_1)^T + \ldots \\ (X_{b3}-\mu_1)(X_{b3}-\mu_1)^T + (X_{b4}-\mu_1)(X_{b4}-\mu_1)^T + \ldots \\ (X_{c1}-\mu_1)(X_{c1}-\mu_1)^T + (X_{c2}-\mu_1)(X_{c2}-\mu_1)^T + \ldots \end{array}\right)$$

Example: Learning HMM Parameters

- State output probability for S2
  - There are 14 observations in S2



Observation 1

Time	1	2	3	4	5	6	7	8	9	10
state	S1	S1	S2	S2	S2	S1	S1	S2	S1	S1
Obs	X_{a1}	X_{a2}	X_{a3}	X_{a4}	X_{a5}	X_{a6}	X_{a7}	X_{a8}	X_{a9}	X_{a10}

Observation 2

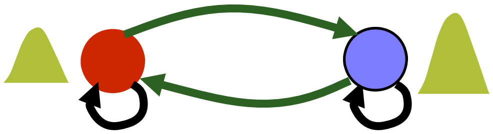
Time	1	2	3	4	5	6	7	8	9
state	S2	S2	S1	S1	S2	S2	S2	S2	S1
Obs	X_{b1}	X_{b2}	X_{b3}	X_{b4}	X_{b5}	X_{b6}	X_{b7}	X_{b8}	X_{b9}

Observation 3

Time	1	2	3	4	5	6	7	8
state	S1	S2	S1	S1	S1	S2	S2	S2
Obs	X_{c1}	X_{c2}	X_{c3}	X_{c4}	X_{c5}	X_{c6}	X_{c7}	X_{c8}

Example: Learning HMM Parameters

- State output probability for S2
  - There are 14 observations in S2
  - Segregate them out and count
    - Compute parameters (mean and variance) of Gaussian output density for state S2



Time	3	4	5	8
state	S2	S2	S2	S2
Obs	X_{a3}	X_{a4}	X_{a5}	X_{a8}

$$P(X \mid S_2) = \frac{1}{\sqrt{(2\pi)^d \mid \Theta_2 \mid}} \exp\left(-0.5(X-\mu_2)^T \Theta_2^{-1}(X-\mu_2)\right)$$

Time	1	2	5	6	7	8
state	S2	S2	S2	S2	S2	S2
Obs	X_{b1}	X_{b2}	X_{b5}	X_{b6}	X_{b7}	X_{b8}

$$\mu_2 = \frac{1}{14}\begin{pmatrix} X_{a3} + X_{a4} + X_{a5} + X_{a8} + X_{b1} + X_{b2} + X_{b5} + \\ X_{b6} + X_{b7} + X_{b8} + X_{c2} + X_{c6} + X_{c7} + X_{c8} \end{pmatrix}$$

Time	2	6	7	8
state	S2	S2	S2	S2
Obs	X_{c2}	X_{c6}	X_{c7}	X_{c8}

$$\Theta_1 = \frac{1}{14}\left((X_{a3}-\mu_2)(X_{a3}-\mu_2)^T + \ldots\right)$$

We have learnt all the HMM parmeters

- State initial probabilities, often denoted as π
  - $\pi(S1) = 0.66$ $\quad\quad$ $\pi(S2) = 1/3 = 0.33$
- State transition probabilities

$$A = \begin{pmatrix} 0.545 & 0.455 \\ 0.385 & 0.615 \end{pmatrix}$$

- State output probabilities

State output probability for S1

$$P(X \mid S_1) = \frac{1}{\sqrt{(2\pi)^d \mid \Theta_1 \mid}} \exp\left(-0.5(X-\mu_1)^T \Theta_1^{-1} (X-\mu_1)\right)$$

State output probability for S2

$$P(X \mid S_2) = \frac{1}{\sqrt{(2\pi)^d \mid \Theta_2 \mid}} \exp\left(-0.5(X-\mu_2)^T \Theta_2^{-1} (X-\mu_2)\right)$$

Update rules at each iteration

$$\pi(s_i) = \frac{\text{No. of observation sequences that start at state } s_i}{\text{Total no. of observation sequences}}$$

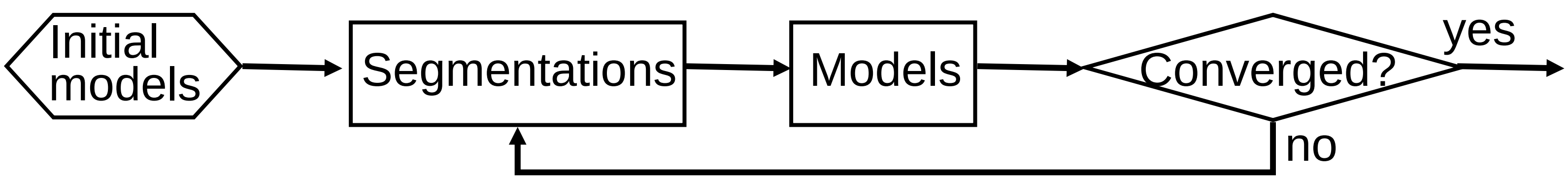
$$P(s_j \mid s_i) = \frac{\sum_{obs} \sum_{t:state(t)=s_i .\&. state(t+1)=s_j} 1}{\sum_{obs} \sum_{t:state(t)=s_i} 1}$$

$$\mu_i = \frac{\sum_{obs} \sum_{t:state(t)=s_i} X_{obs,t}}{\sum_{obs} \sum_{t:state(t)=s_i} 1}$$

$$\Theta_i = \frac{\sum_{obs} \sum_{t:state(t)=s_i} (X_{obs,t} - \mu_i)(X_{obs,t} - \mu_i)^T}{\sum_{obs} \sum_{t:state(t)=s_i} 1}$$

- Assumes state output PDF = Gaussian
  - For GMMs, estimate GMM parameters from collection of observations at any state

Training by segmentation: Viterbi training



- Initialize all HMM parameters
- Segment all training observation sequences into states using the Viterbi algorithm with the current models
- Using estimated state sequences and training observation sequences, reestimate the HMM parameters
- This method is also called a “segmental k-means” learning procedure

Alternative to counting: SOFT counting

- Expectation maximization
- *Every* observation contributes to every state

Update rules at each iteration

$$\pi(s_i) = \frac{\sum_{Obs} P(state(t=1) = si \mid Obs)}{\text{Total no. of observation sequences}}$$

$$P(s_j \mid s_i) = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i, state(t+1) = s_j \mid Obs)}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\mu_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs) X_{Obs,t}}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\Theta_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)(X_{Obs,t} - \mu_i)(X_{Obs,t} - \mu_i)^T}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

- Every observation contributes to every state

Update rules at each iteration

$$\pi(s_i) = \frac{\sum_{Obs} P(state(t=1) = s_i \mid Obs)}{\text{Total no. of observation sequences}}$$

$$P(s_j \mid s_i) = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i, state(t+1) = s_j \mid Obs)}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\mu_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs) X_{Obs,t}}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\Theta_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)(X_{Obs,t} - \mu_i)(X_{Obs,t} - \mu_i)^T}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

- Where did these terms come from?

$$P(state(t) = s \mid Obs)$$

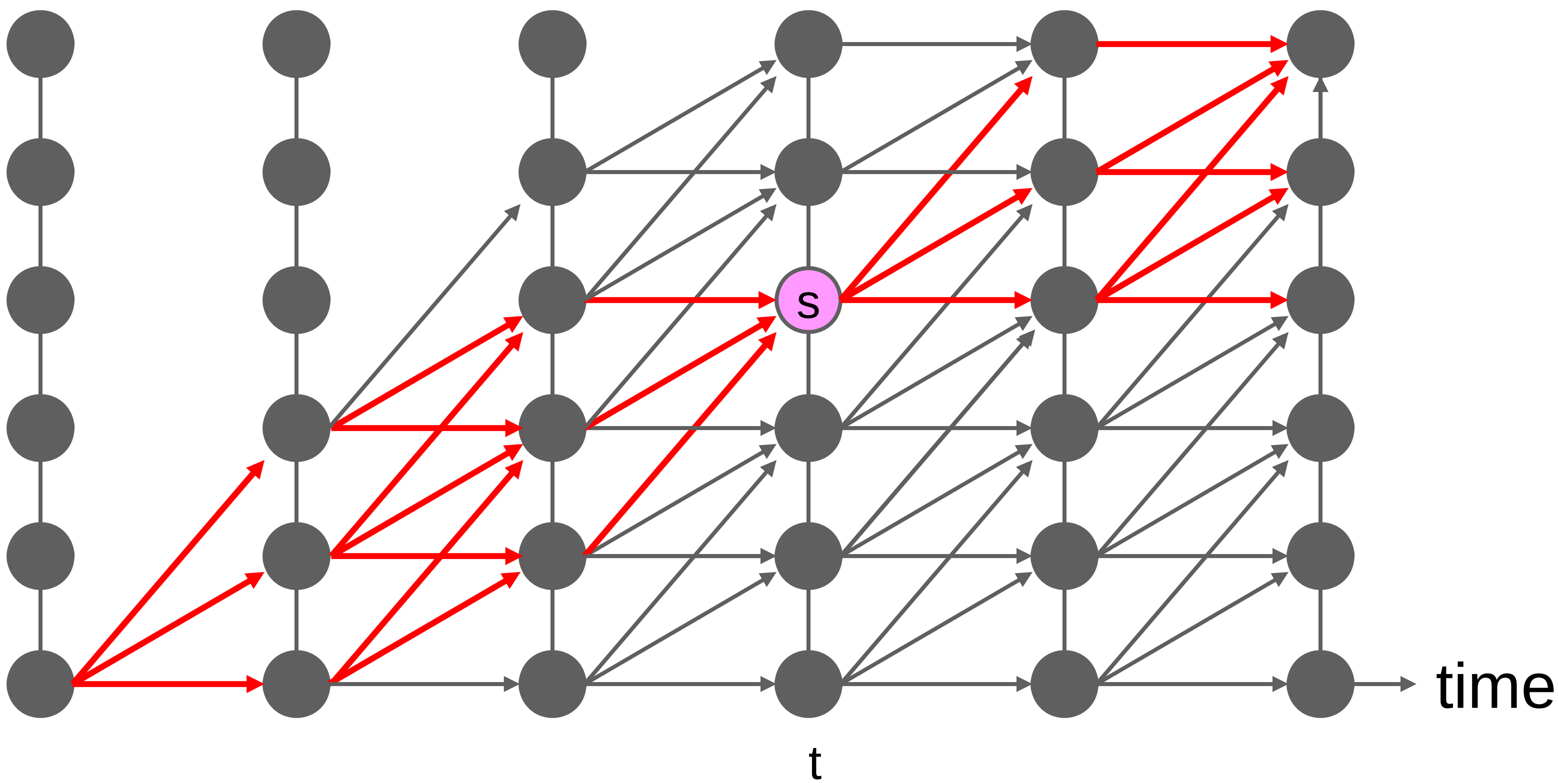
- The probability that the process was at *s* when it generated X_t given the entire observation
  - Dropping the "Obs" subscript for brevity

$$P(state(t) = s \mid X_1, X_2, \ldots, X_T) \propto P(state(t) = s, X_1, X_2, \ldots, X_T)$$

- We will compute $P(state(t) = s_i, x_1, x_2, \ldots, x_T)$ first
  - This is the probability that the process visited *s* at time *t* while producing the entire observation

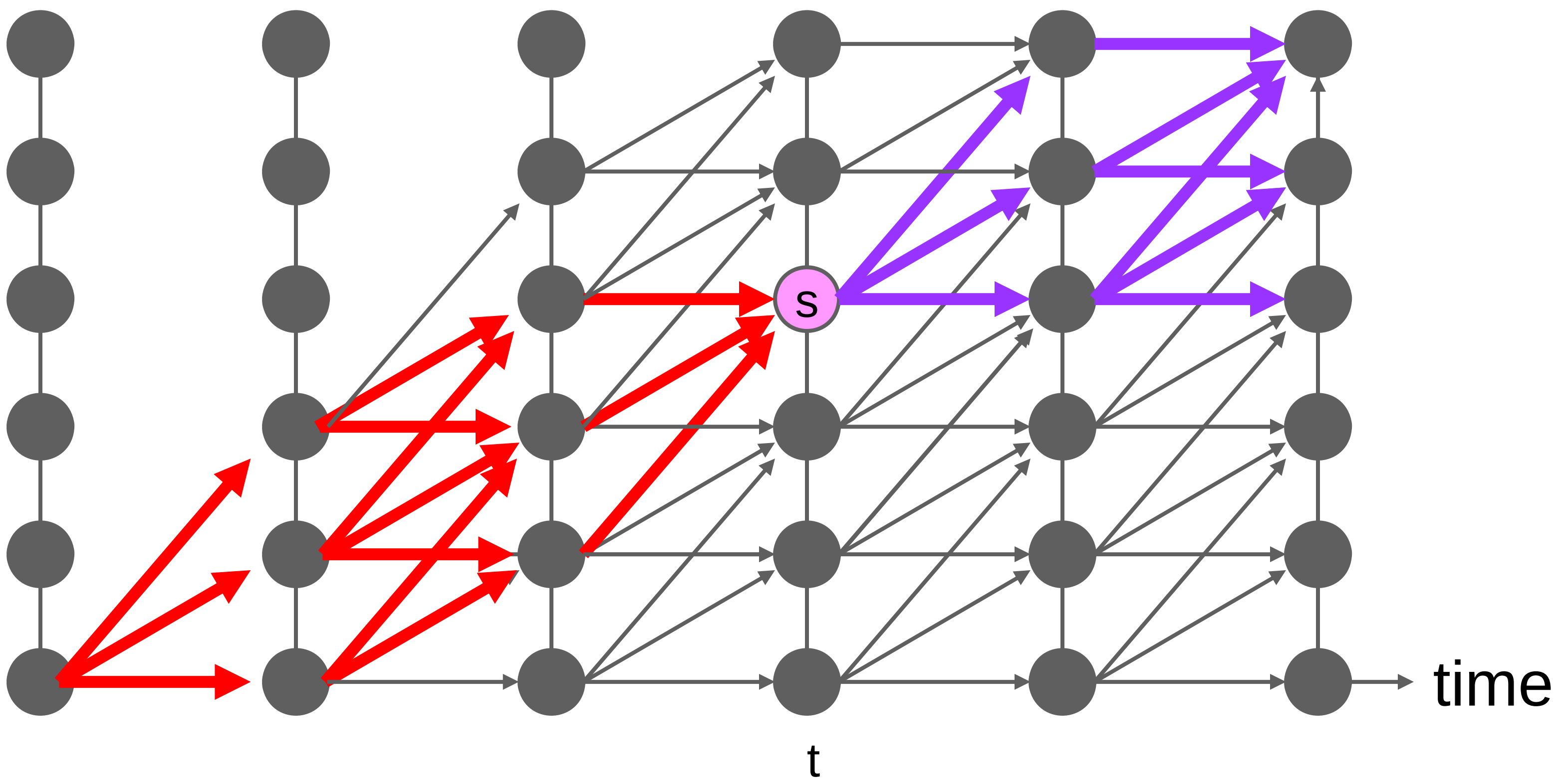
$P(state(t) = s, x_1, x_2, \ldots, x_T)$

- The probability that the HMM was in a particular state *s* when generating the observation sequence is the probability that it followed a state sequence that passed through *s* at time *t*



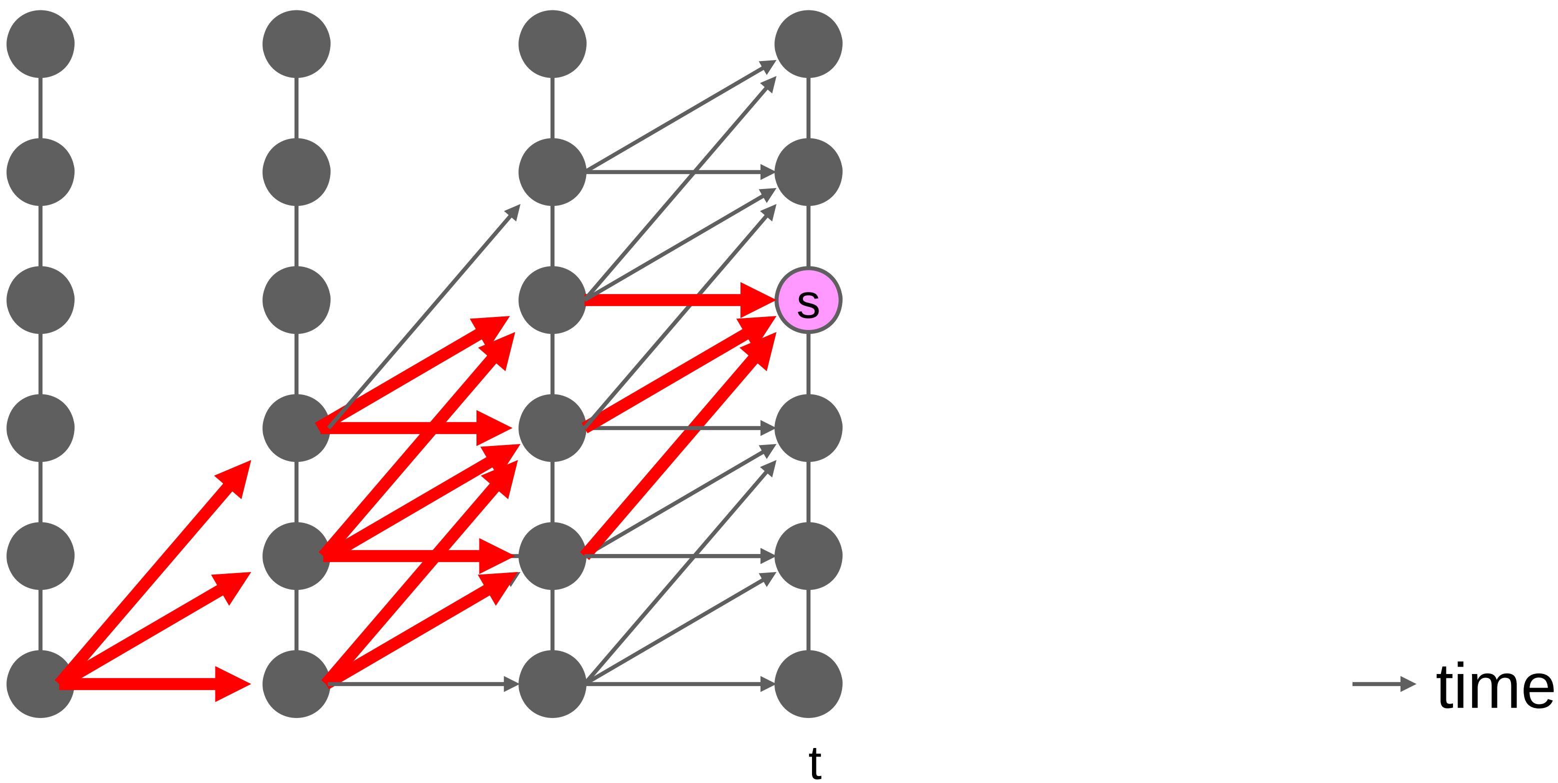
$$P(state(t) = s, x_1, x_2, ..., x_T)$$

- This can be decomposed into two multiplicative sections
  - The section of the lattice leading into state *s* at time t and the section leading out of it



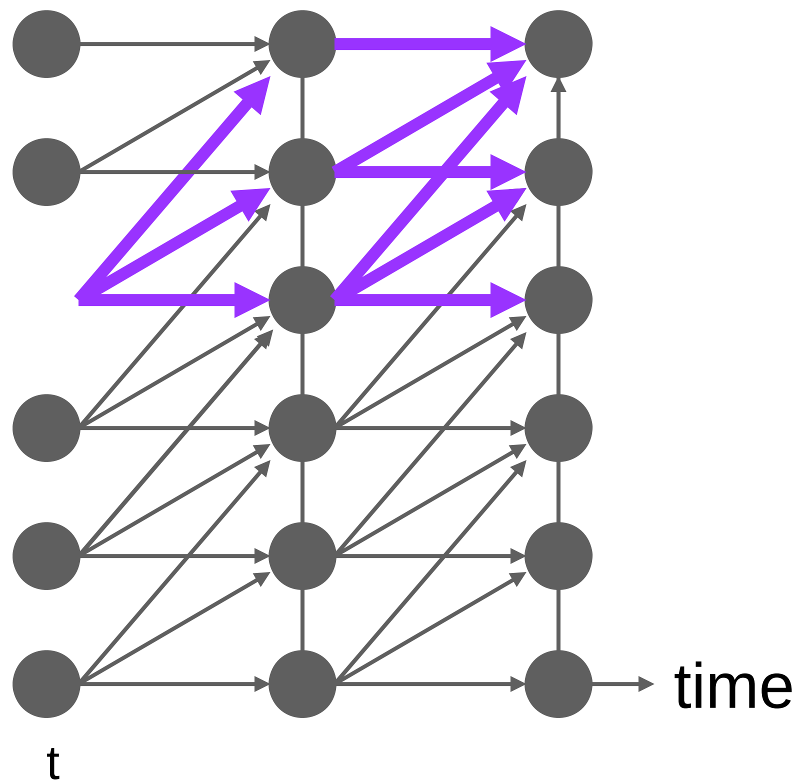
The Forward Paths

- The probability of the red section is the total probability of all state sequences ending at state *s* at time *t*
  - This is simply $\alpha(s,t)$
  - Can be computed using the forward algorithm



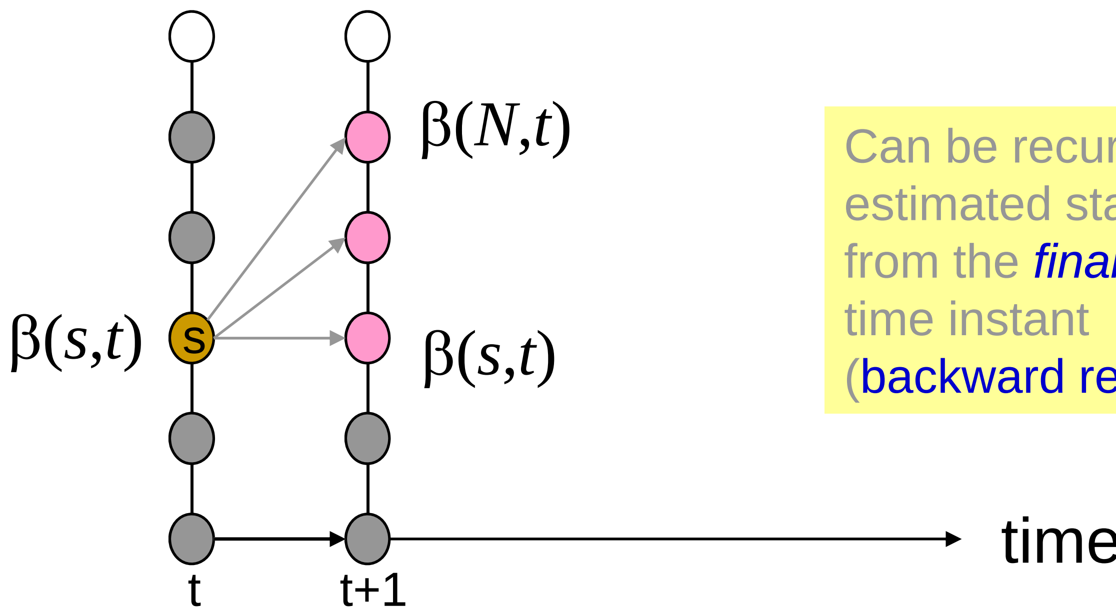
The Backward Paths

- The blue portion represents the probability of all state sequences that began at state *s* at time *t*
  - Like the red portion it can be computed using a *backward recursion*



The Backward Recursion

$$\beta(s,t) = P(x_{t+1}, x_{t+2}, \ldots, x_T \mid state(t) = s)$$



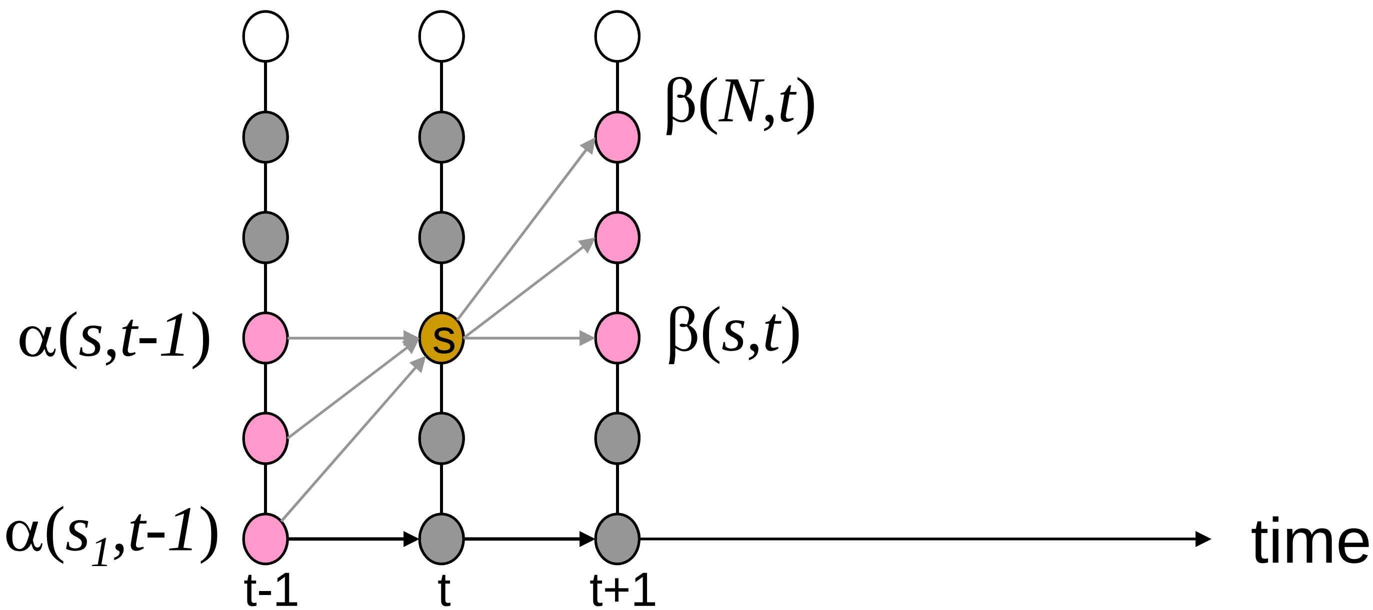
Can be recursively estimated starting from the *final* time time instant (backward recursion)

$$\beta(s,t) = \sum_{s'} \beta(s',t+1) P(s'|s) P(x_{t+1}|s')$$

- β(*s*,*t*) is the total probability of ALL state sequences that depart from *s* at time *t*, and all observations after x_t
  - $\beta(s,T)$ = 1 at the final time instant for all valid final states

The complete probability

$$\alpha(s,t)\beta(s,t) = P(x_{t+1}, x_{t+2}, \ldots, x_T, state(t) = s)$$



Posterior probability of a state

- The probability that the process was in state *s* at time *t*, given that we have observed the data is obtained by simple normalization

$$P(state(t)=s\,|\,Obs)=\frac{P(state(t)=s,x_1,x_2,\ldots,x_T)}{\sum_{s'}P(state(t)=s,x_1,x_2,\ldots,x_T)}=\frac{\alpha(s,t)\beta(s,t)}{\sum_{s'}\alpha(s',t)\beta(s',t)}$$

- This term is often referred to as the gamma term and denoted by $\gamma_{s,t}$

Update rules at each iteration

$$\pi(s_i) = \frac{\sum_{Obs} P(state(t=1) = s_i \mid Obs)}{\text{Total no. of observation sequences}}$$

$$P(s_j \mid s_i) = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i, state(t+1) = s_j \mid Obs)}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\mu_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs) X_{Obs,t}}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\Theta_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)(X_{Obs,t} - \mu_i)(X_{Obs,t} - \mu_i)^T}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

- These have been found

Update rules at each iteration

$$\pi(s_i) = \frac{\sum_{Obs} P(state(t=1) = s_i \mid Obs)}{\text{Total no. of observation sequences}}$$

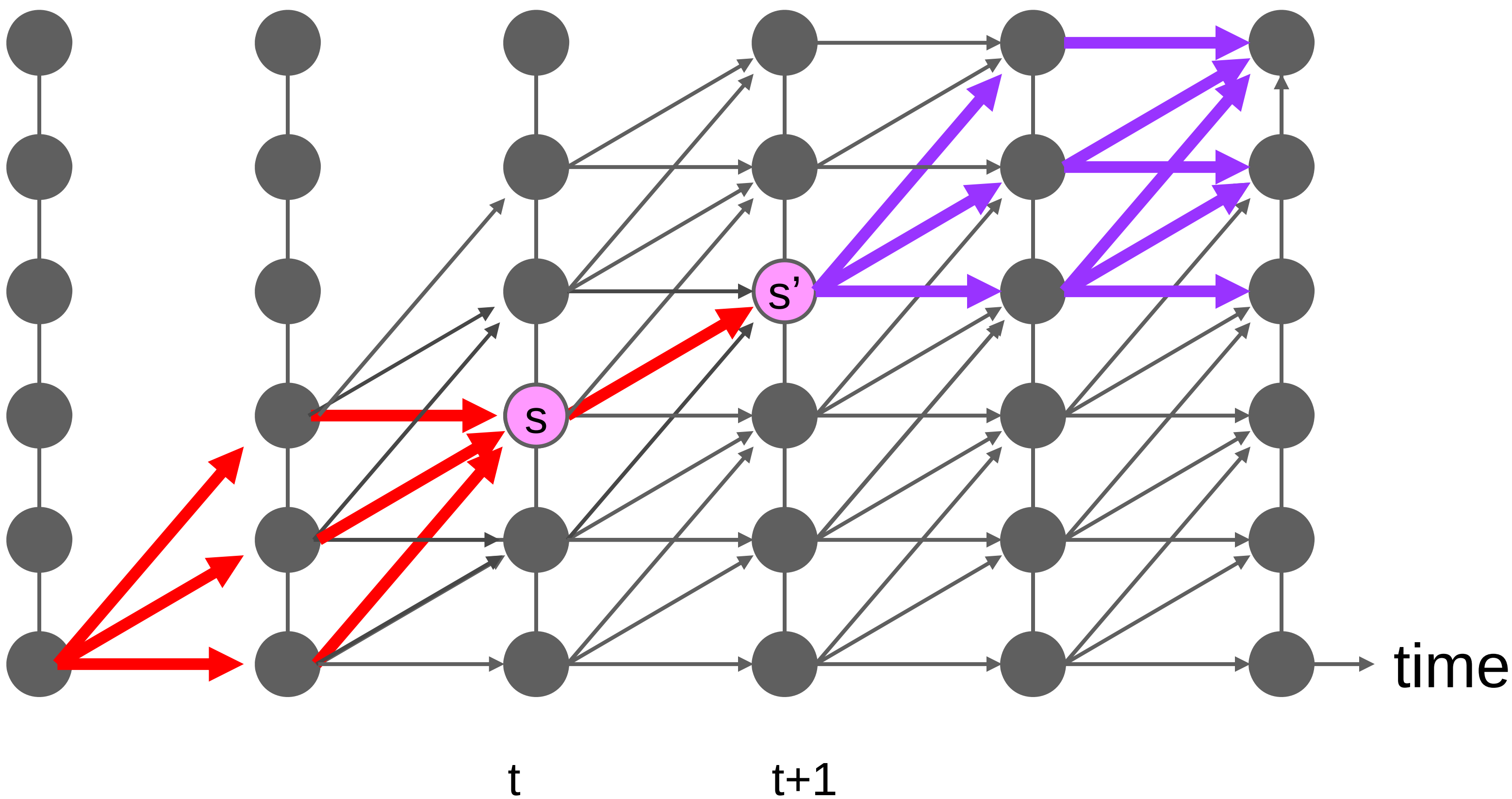
$$P(s_j \mid s_i) = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i, state(t+1) = s_j \mid Obs)}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\mu_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs) X_{Obs,t}}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\Theta_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)(X_{Obs,t} - \mu_i)(X_{Obs,t} - \mu_i)^T}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

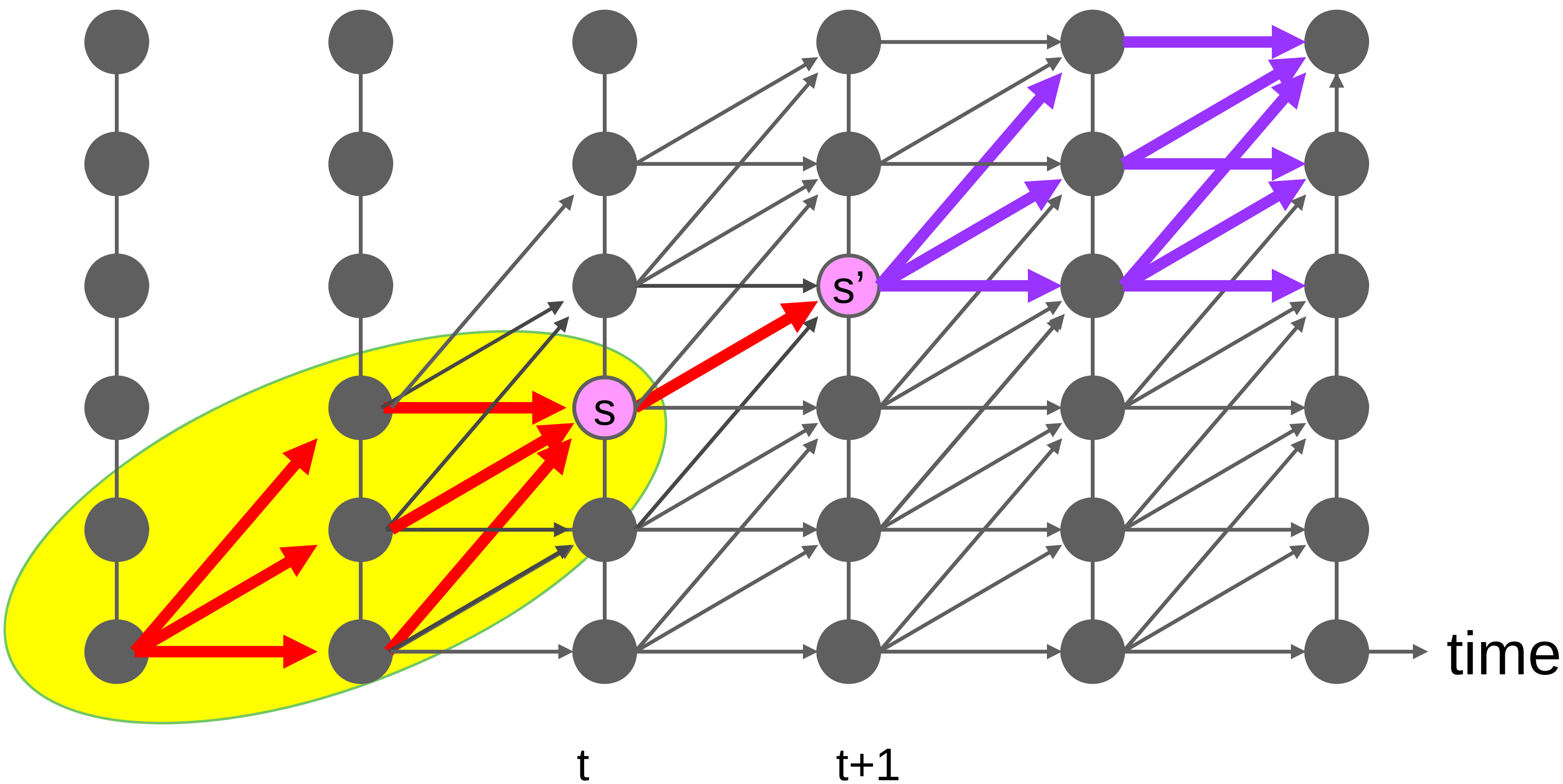
- Where did these terms come from?

$$P(state(t) = s, state(t+1) = s', x_1, x_2, \ldots, x_T)$$



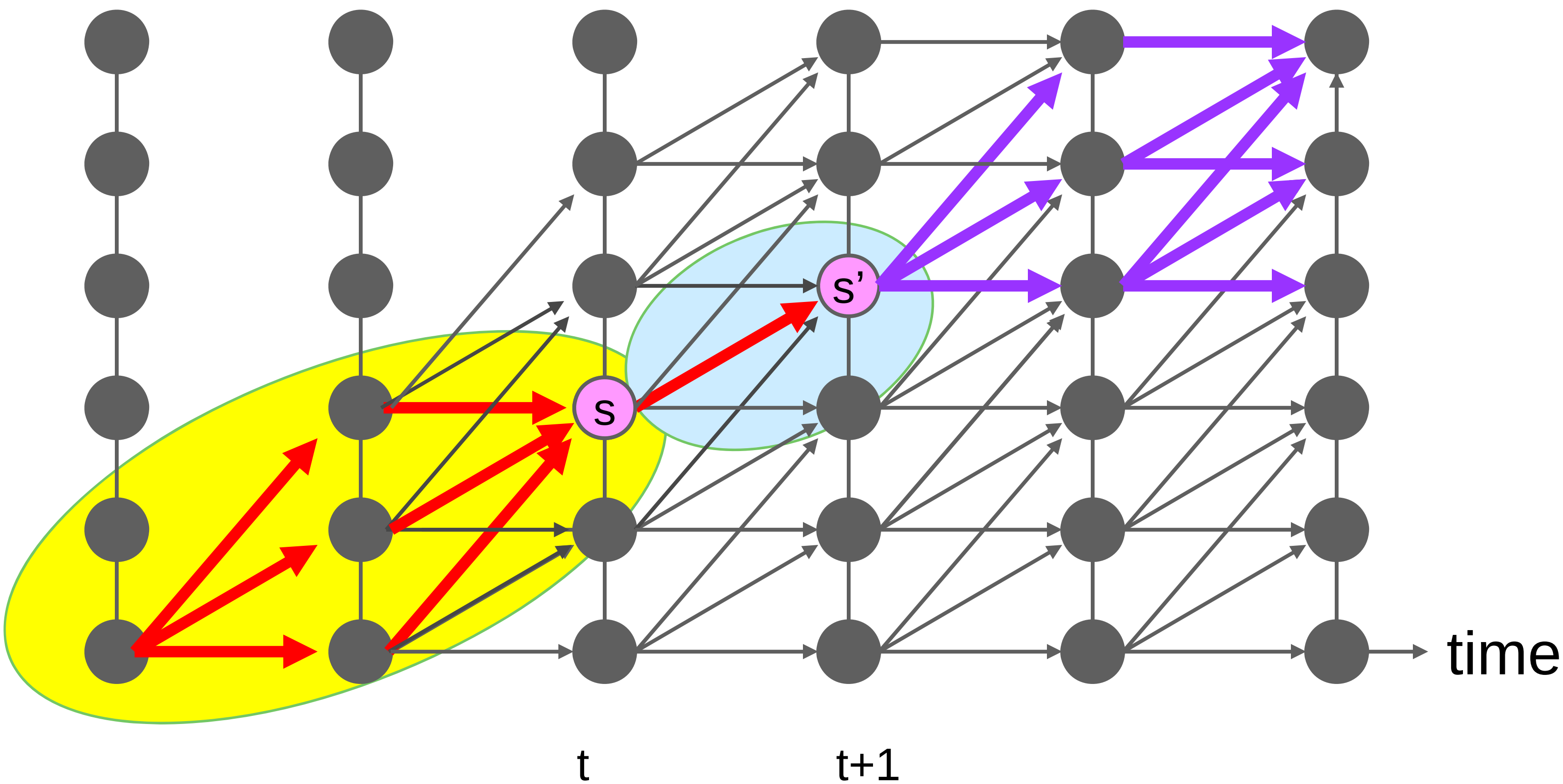
$$P(state(t) = s, state(t+1) = s', x_1, x_2, \ldots, x_T)$$

$$\alpha(s,t)$$



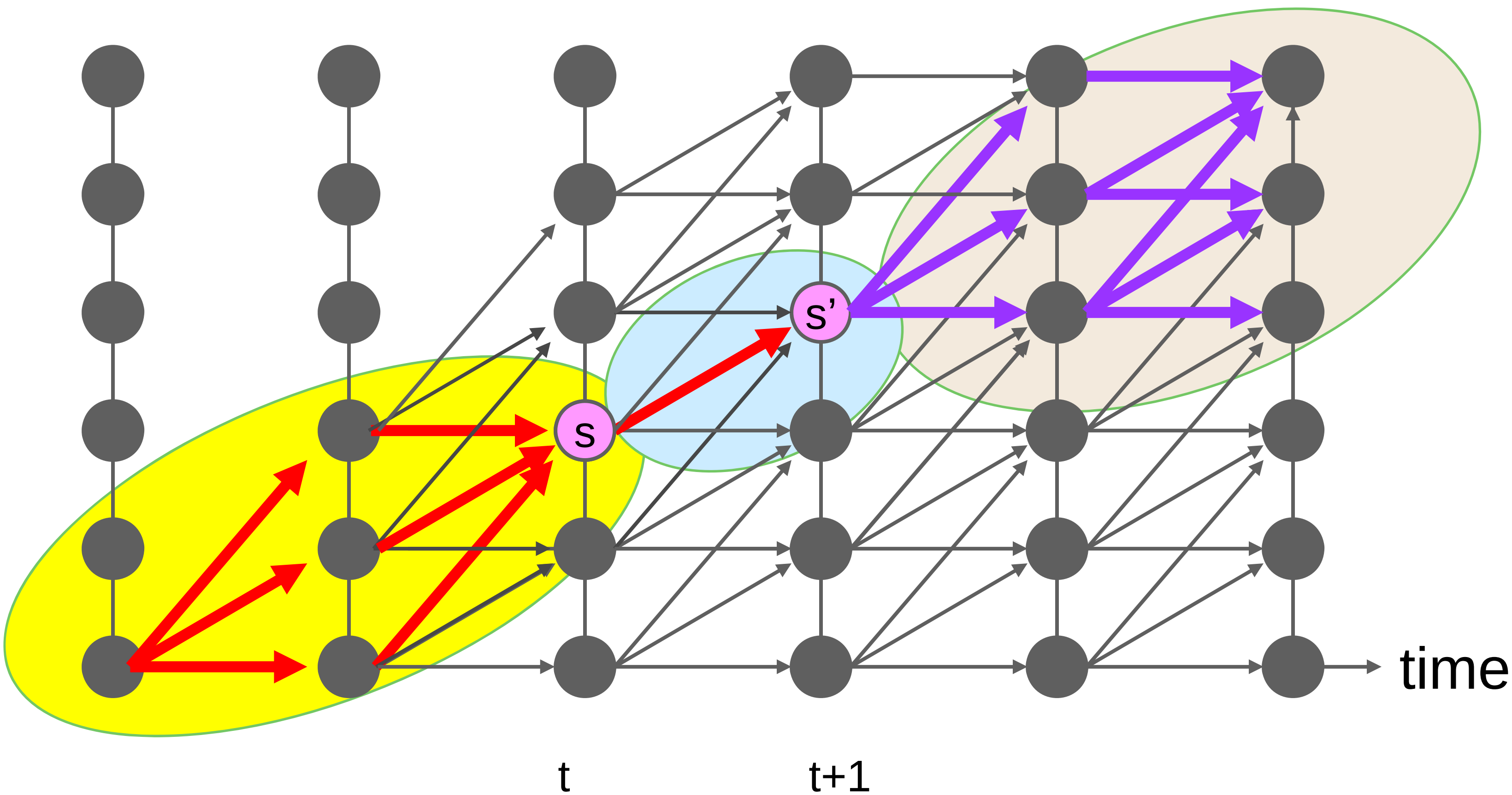
$$P(state(t) = s, state(t+1) = s', x_1, x_2, ..., x_T)$$

$$\alpha(s,t)\, P(s'|s)P(x_{t+1}|s')$$



$$P(state(t) = s, state(t+1) = s', x_1, x_2, \ldots, x_T)$$

$$\alpha(s,t)\, P(s'|s) P(x_{t+1}|s')\, \beta(s', t+1)$$



The a posteriori probability of transition

$$P(state(t) = s, state(t+1) = s' | Obs) = \frac{\alpha(s,t)P(s'|s)P(x_{t+1}|s')\beta(s',t+1)}{\sum_{s_1}\sum_{s_2}\alpha(s_1,t)P(s_2|s_1)P(x_{t+1}|s_2)\beta(s_2,t+1)}$$

- The a posteriori probability of a transition given an observation

Update rules at each iteration

$$\pi(s_i) = \frac{\sum_{Obs} P(state(t=1) = s_i \mid Obs)}{\text{Total no. of observation sequences}}$$

$$P(s_j \mid s_i) = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i, state(t+1) = s_j \mid Obs)}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

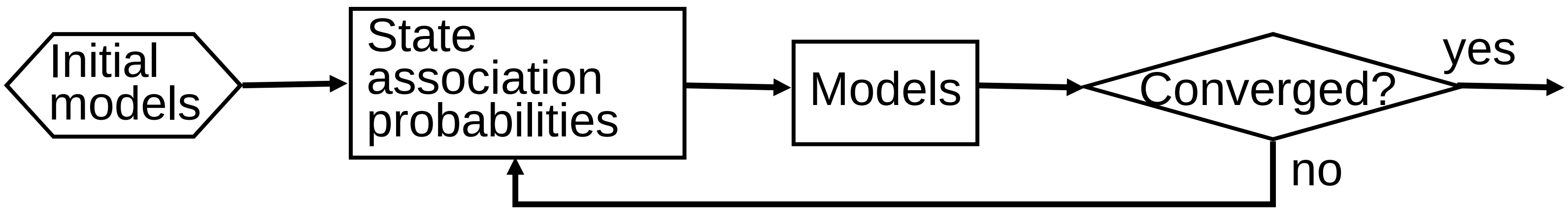
$$\mu_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs) X_{Obs,t}}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

$$\Theta_i = \frac{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)(X_{Obs,t} - \mu_i)(X_{Obs,t} - \mu_i)^T}{\sum_{Obs}\sum_{t} P(state(t) = s_i \mid Obs)}$$

- These have been found

Training without explicit segmentation: Baum-Welch training

- Every feature vector associated with every state of every HMM with a probability



- Probabilities computed using the forward-backward algorithm
- Soft decisions taken at the level of HMM state
- In practice, the segmentation based Viterbi training is much easier to implement and is much faster
- The difference in performance between the two is small, especially if we have lots of training data

HMM Issues

- How to find the best state sequence: Covered
- How to learn HMM parameters: Covered
- How to compute the probability of an observation sequence: Covered

Magic numbers

- How many states:
  - No nice automatic technique to learn this
  - You choose
    - For speech, HMM topology is usually left to right (no backward transitions)
    - For other cyclic processes, topology must reflect nature of process
    - No. of states – 3 per phoneme in speech
    - For other processes, depends on estimated no. of distinct states in process

Applications of HMMs

- Classification:
  - Learn HMMs for the various classes of time series from training data
  - Compute probability of test time series using the HMMs for each class
  - Use in a Bayesian classifier

  - Speech recognition, vision, gene sequencing, character recognition, text mining, topic detection…

Applications of HMMs

- Segmentation:
  - Given HMMs for various events, find event boundaries
    - Simply find the best state sequence and the locations where state identities change

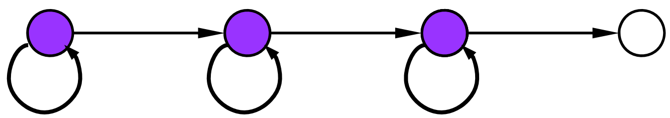
- Automatic speech segmentation, text segmentation by topic, geneome segmentation, …

Implementation Issues

- For long data sequences arithmetic underflow is a problem
  - Scores are products of numbers that are all less than 1
- The Viterbi algorithm provides a workaround – work only with *log* probabilities
  - Multiplication changes to addition – computationally faster too
  - Underflow almost completely eliminated
- For the forward algorithm complex normalization schemes must be implemented to prevent underflow
  - At some computational expense
  - Often not worth it – go with Viterbi

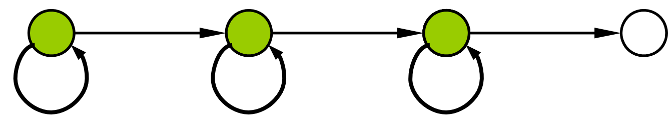
Classification with HMMs

HMM for *Yes*



P(*Yes*) P(*X*|*Yes*)

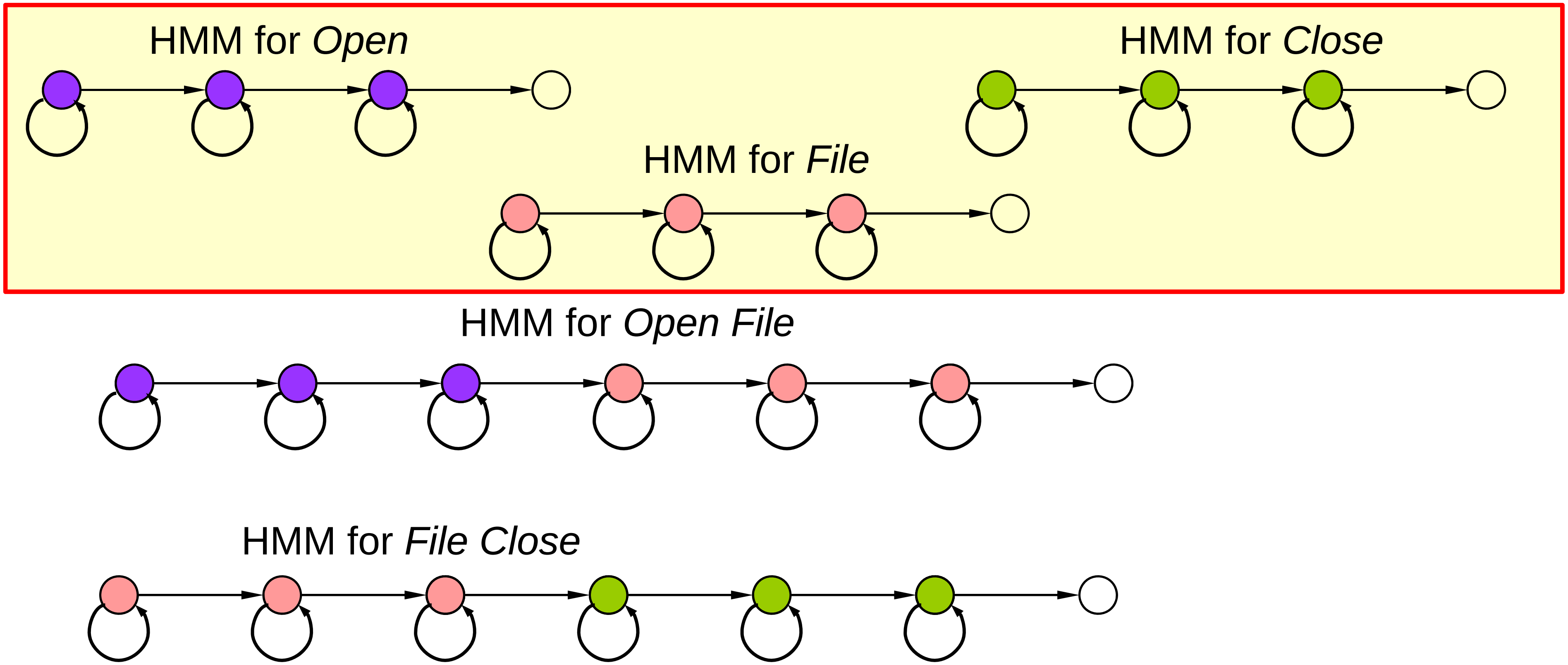
HMM for *No*



P(*No*) P(*X*|*No*)

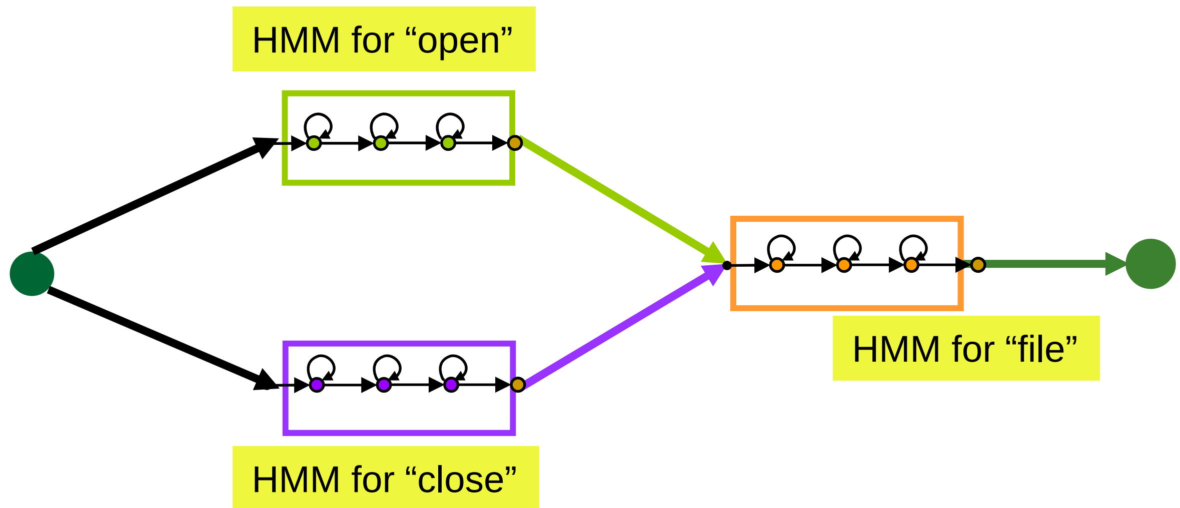
- Speech recognition of isolated words:
- Training:
  - Collect training instances for each word
  - Learn an HMM for each word
- Recognition of an observation X
  - For each word compute P(X|word)
    - Using forward algorithm
    - Alternately, compute P(X,best.state.sequence |word)
      - Computed using the Viterbi segmentation algorithm
  - Compute P(word) P(X|word)
    - P(word) = a priori probability of word
  - Select the word for which P(word) P(X|word) is highest

Creating composite models



- HMMs with absorbing states can be combined into composites
  - E.g. train models for open, close and file
  - Concatenate them to create models for "open file" and "file close"
    - Can recognize "open file" and "file close"

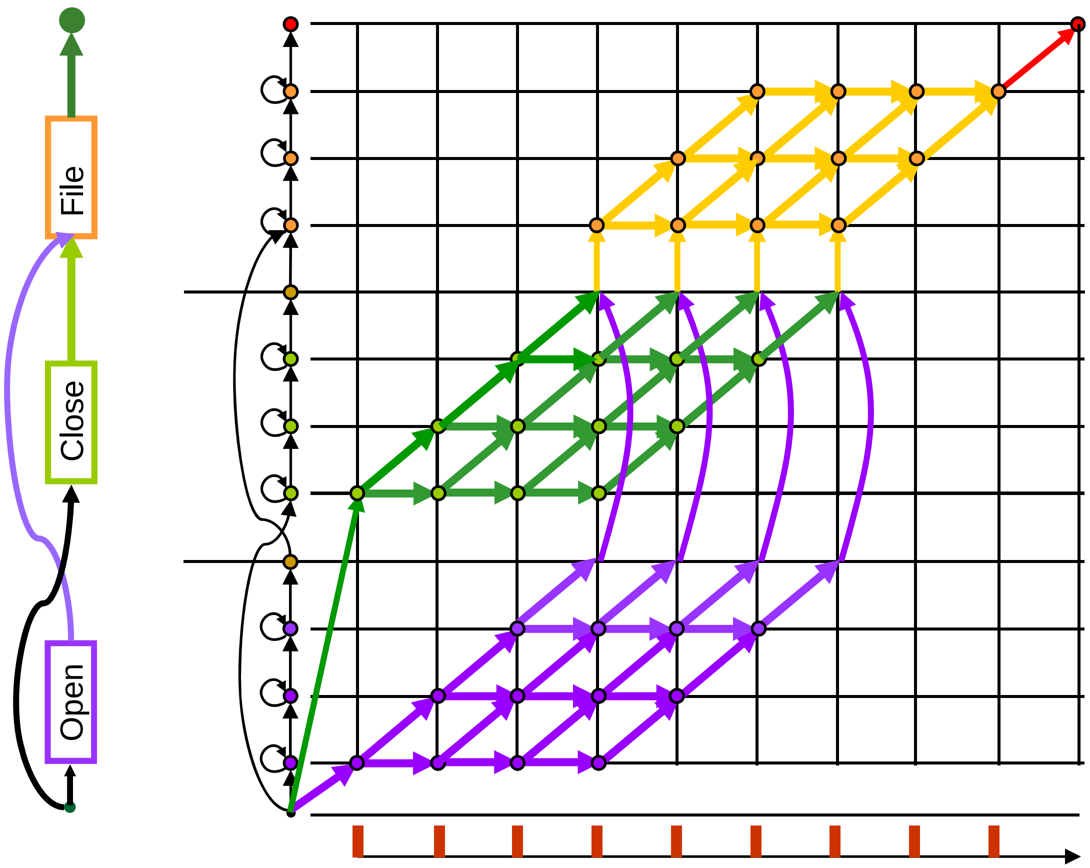
Model graphs



- Models can also be composed into graphs
  - Not just linearly
- Viterbi state alignment will tell us which portions of the graphs were visited for an observation X

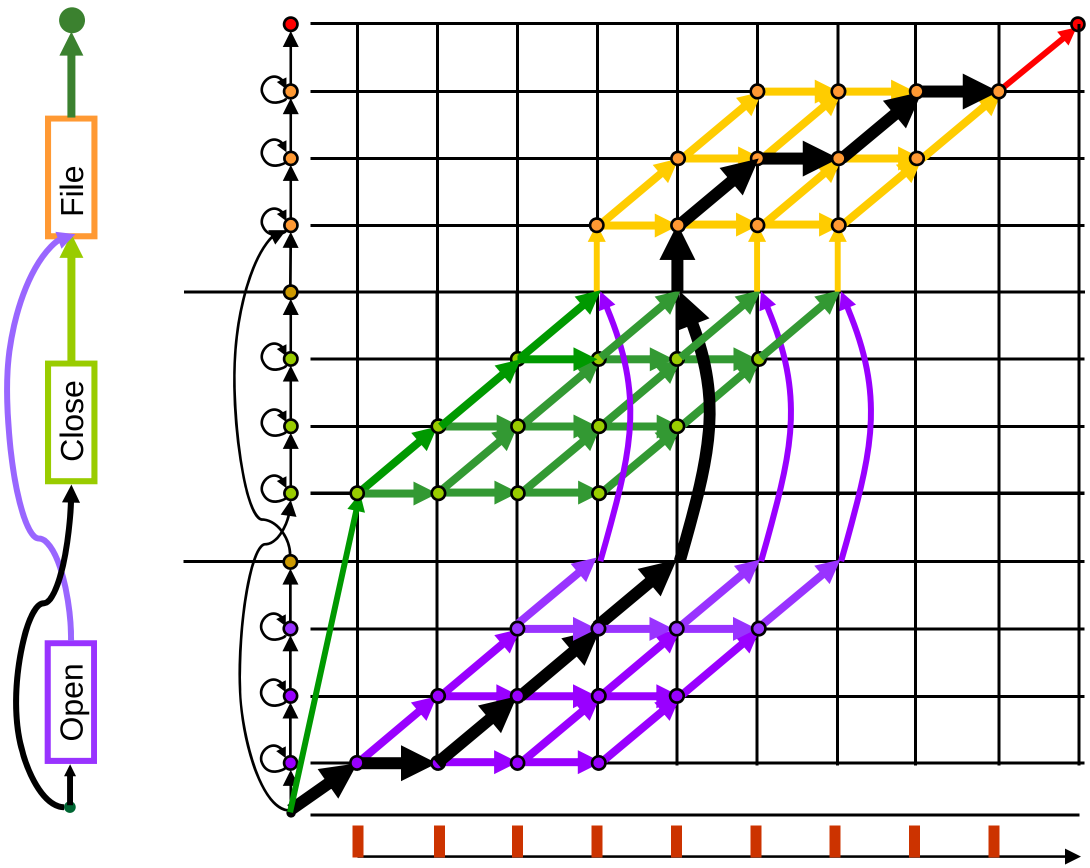
Recognizing from graph

- Trellis for “Open File” vs. “Close File”
  - The VITERBI best path tells you what was spoken

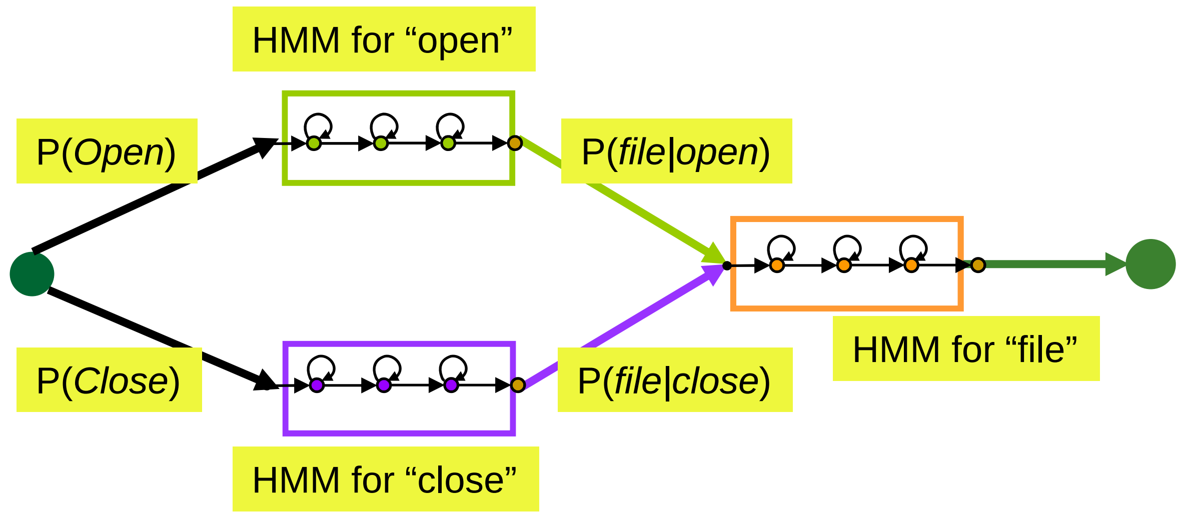


Recognizing from graph

- Trellis for “Open File” vs. “Close File”
  - The VITERBI best path tells you what was spoken



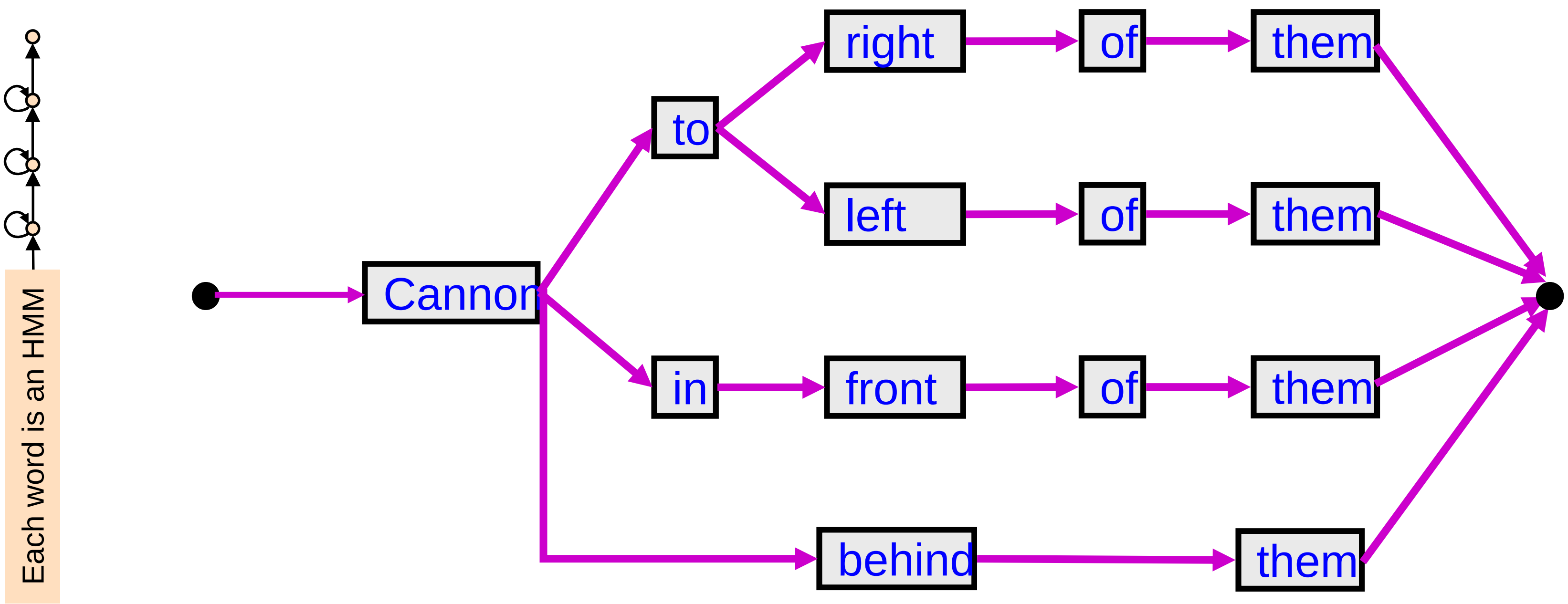
“Language” probabilities can be incorporated



- Transitions between HMMs can be assigned a probability
  - Drawn from properties of the language
  - Here we have shown “Bigram” probabilities

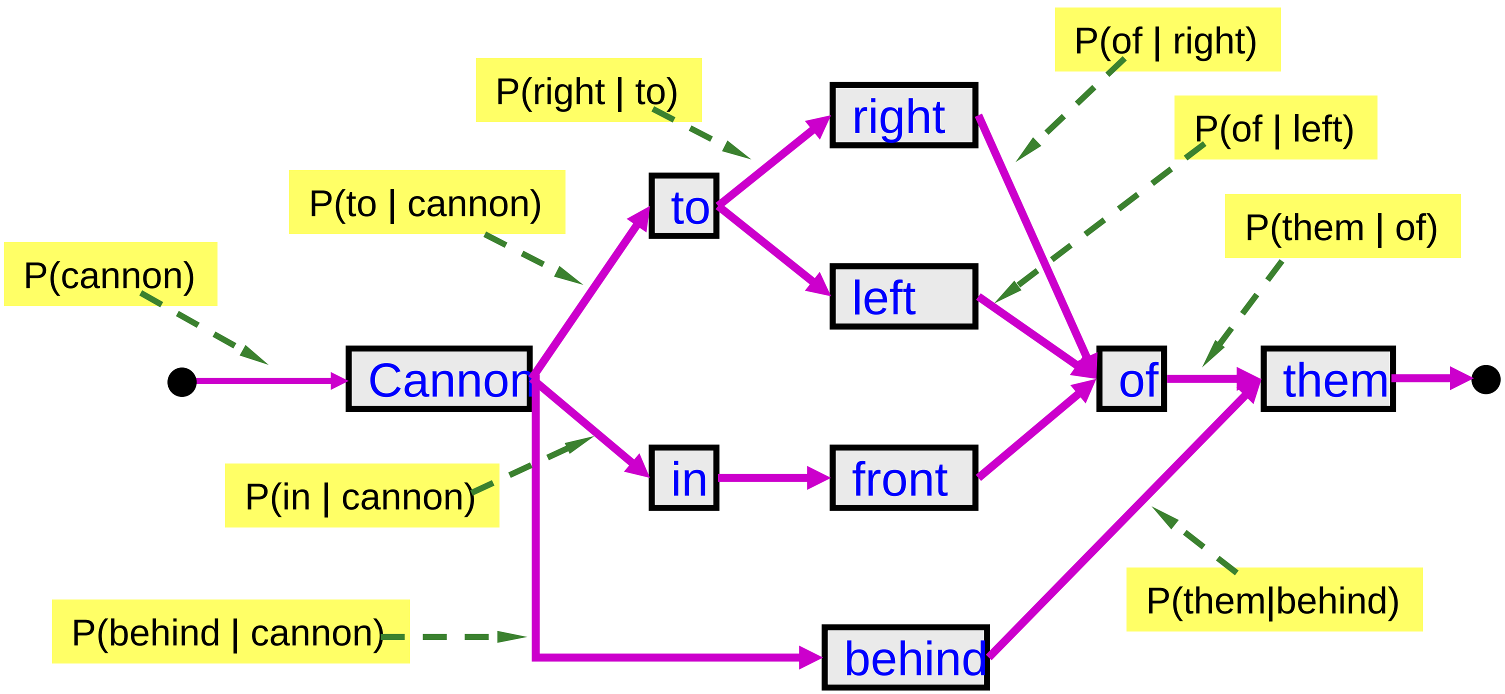
This is used in speech recognition

- Recognizing one of four lines from “charge of the light brigade”
  - Cannon to right of them
  - Cannon to left of them
  - Cannon in front of them
  - Cannon behind them
- Each “word” is an HMM



Graphs can be reduced sometimes

- Recognizing one of four lines from “charge of the light brigade”
  - Graph reduction does not impede recognition of what was spoken



Speech recognition: An aside

- In speech recognition systems models are trained for *phonemes*
  - Actually “triphones” – phonemes in context
- Word HMMs are composed from phoneme HMMs
- Language HMMs are composed from word HMMs
- The graph is “reduced” using automated techniques
  - John McDonough talks about WFSTs on Thursday