Lecture 23:
Hidden Markov Models

Not in book

Dinucleotide Frequency

- Consider all 2-mers in a sequence
  {AA,AC,AG,AT,CA,CC,CG,CT,GA,GC,GG,GT,TA,TC,TG,TT}
- Given 4 nucleotides:
  each with probability of occurrence is ~ ¼.
  Thus, one would expect that the probability of occurrence of any given dinucleotide is ~ 1/16.
- However, the frequencies of dinucleotides in DNA sequences vary widely.
- In particular, CG is typically underepresented (frequency of CG is typically < 1/16)

Example

- From a 291829 base sequence

AA	0.120214646984	GA	0.056108392614
AC	0.055409350713	GC	0.037792809463
AG	0.068848773935	GG	0.043357731266
AT	0.083425853585	GT	0.046828954041
CA	0.074369148950	TA	0.077206436668
CC	0.044927148868	TC	0.056207766218
CG	0.008179475581	TG	0.063698479926
CT	0.066857875186	TT	0.096567155996

- Expected value 0.0625
- CG is 7 times smaller than expected

Why so few CGs?

- CG is the least frequent dinucleotide because C in CG is easily *methylated*. And, methylated Cs are easily mutated into Ts.
- However, methylation is suppressed around genes and transcription factor regions
- So, CG appears at *relatively* higher frequency in these important areas
- These localized areas of higher CG frequency are called *CG-islands*
- Finding the CG islands within a genome is among the most reliable gene finding approaches

CG Island Analogy

- The CG islands problem can be modeled by a toy problem named *"The Fair Bet Casino"*
- The outcome of the game is determined by coin flips with two possible outcomes: **H**eads or **T**ails
- However, there are two different coins
  - A **F**air coin: **H**eads and **T**ails with same probability ½.
  - The **B**iased coin: **H**eads with prob. ¾, **T**ails with prob. ¼.

The "Fair Bet Casino" (cont'd)

- Thus, we define the probabilities:
  - P(H | Fair) = P(T | Fair) = ½
  - P(H | Bias) = ¾, P(T | Bias) = ¼
  - The house doesn't want to get caught switching between coins, so they do so infrequently
  - Changes between Fair and Biased coins with probability 10%

The Fair Bet Casino Problem

- **Input:** A sequence $x = x_1x_2x_3\ldots x_n$ of *observed* coin tosses made by some combination of the two possible coins (***F*** or ***B***).

- **Output:** A sequence $\pi = \pi_1\, \pi_2\, \pi_3\ldots\, \pi_n$, with each π_i being either *F* or *B* indicating that x_i is the result of tossing the Fair or Biased coin respectively.

Problem…

Fair Bet Casino Problem

Any observed outcome of coin tosses *could* have been generated by *either* coin, or any combination.

But, all coin exchange combinations are not equally likely. What coin exchange combination has the highest probability of generating the observed series of tosses?

↓

Decoding Problem

P(x | fair coin) vs. P(x | biased coin)

- Suppose first, that the dealer never exchanges coins.
- Some definitions:
  - P(x | Fair): prob. of the dealer generating the outcome x using the *Fair* coin.
  - P(x | Biased): prob. of the dealer generating outcome x using the *Biased* coin .

P(x | fair coin) vs. P(x | biased coin)

- $P(x \mid \text{Fair}) = P(x_1 \ldots x_n \mid \text{Fair}) =$

  $$\prod_{i=1,n} p\,(x_i \mid \text{Fair}) = (1/2)^n$$

- $P(x \mid \text{Biased}) = P(x_1 \ldots x_n \mid \text{Biased coin}) =$

  $$\prod_{i=1,n} p\,(x_i \mid \text{Biased}) = (3/4)^k (1/4)^{n-k} = 3^k / 4^n$$

  – Where k is the number of ***H***eads in x.

P(x | fair coin) vs. P(x | biased coin)

- When is a sequence equally likely to have come from the Fair or Biased coin?

$$P(x \mid \text{Fair}) = P(x \mid \text{Biased})$$

$$1/2^n = 3^k/4^n$$

$$2^n = 3^k$$

$$n = k \log_2 3$$

- when $k = n / \log_2 3$ $(k \sim 0.63\, n)$
- So when the number of heads is greater than 63% the dealer most likely used the biased coin

Log-odds Ratio

- We can define the ***log-odds ratio*** as follows:

$$\log_2(P(x \mid \text{Fair}) \;/\; P(x \mid \text{Biased})) =$$
$$= \Sigma^k_{i=1} \log_2(p(x_i \mid \mathit{Fair}) \;/\; p(x_i \mid \mathit{Biased}))$$
$$= n - k \log_2 3$$

- The log-odds ratio is a means (threshold) for deciding which of two alternative hypotheses is most likely
- "Zero-crossing" measure; if the log-odds ratio > 0 then the numerator is more likely, if it is < 0 then the denominator is more likely, they are equally likely if the log-odds ratio = 0

Computing Log-odds Ratio in Sliding Windows

$x_1 x_2 \boxed{x_3 x_4 x_5 x_6 x_7} x_8 \ldots x_n$

Consider a *sliding window* of the outcome sequence. Find the log-odds for this short window.

Disadvantages:

- the length of CG-island (appropriate window size) is not known in advance
- different window sizes may classify the same position differently

Key Elements of this Problem

- There is an unknown, *hidden*, state for each observation (Was the coin the Fair or Biased?)
- Outcomes are modeled probabilistically:
  - P(H | Fair) = P(T | Fair) = ½
  - P(H | Bias) = ¾, P(T | Bias) = ¼
- Transitions between states are modeled probabilistically:
  - $P(\pi_i = \text{Biased} \mid \pi_{i-1} = \text{Biased}) = a_{BB} = 0.9$
  - $P(\pi_i = \text{Biased} \mid \pi_{i-1} = \text{Fair}) = a_{FB} = 0.1$
  - $P(\pi_i = \text{Fair} \mid \pi_{i-1} = \text{Biased}) = a_{BF} = 0.1$
  - $P(\pi_i = \text{Fair} \mid \pi_{i-1} = \text{Fair}) = a_{FF} = 0.9$

Hidden Markov Model (HMM)

- A generalization of this class of problem
- Can be viewed as an abstract machine with *k hidden* states that emits symbols from an alphabet Σ.
- Each state has its own probability distribution, and the machine switches between states according to this probability distribution.
- While in a certain state, the machine makes 2 decisions:
  - What state should I move to next?
  - What symbol - from the alphabet Σ - should I emit?

Why "Hidden"?

- Observers can see the emitted symbols of an HMM but have *no ability to know which state the HMM is currently in*.
- Thus, the goal is to infer the *most likely hidden states of an HMM* based on the given sequence of emitted symbols.

HHHTHTHHTTTTHTHTHTHHHTHTHTHT
BBBFFFFFFFFFFFFFFFFBBBFFFFFF?

HMM Parameters

Σ: set of emission characters.

Ex.: Σ = {0, 1} for coin tossing
(0 for ***T***ails and 1 ***H***eads)
Σ = {1, 2, 3, 4, 5, 6} for dice tossing

Q: set of hidden states, emitting symbols from Σ.
Q = {F,B} for coin tossing

HMM Parameters (cont'd)

$A = (a_{kl})$: a |Q| x |Q| matrix of probability of changing from state *k* to state *l*. *Transition matrix*

$a_{FF} = 0.9$ $a_{FB} = 0.1$

$a_{BF} = 0.1$ $a_{BB} = 0.9$

$E = (e_k(b))$: a |Q| x |Σ| matrix of probability of emitting symbol *b* while being in state *k*. *Emission matrix*

$e_F(T) = ½$ $e_F(H) = ½$

$e_B(T) = ¼$ $e_B(H) = ¾$

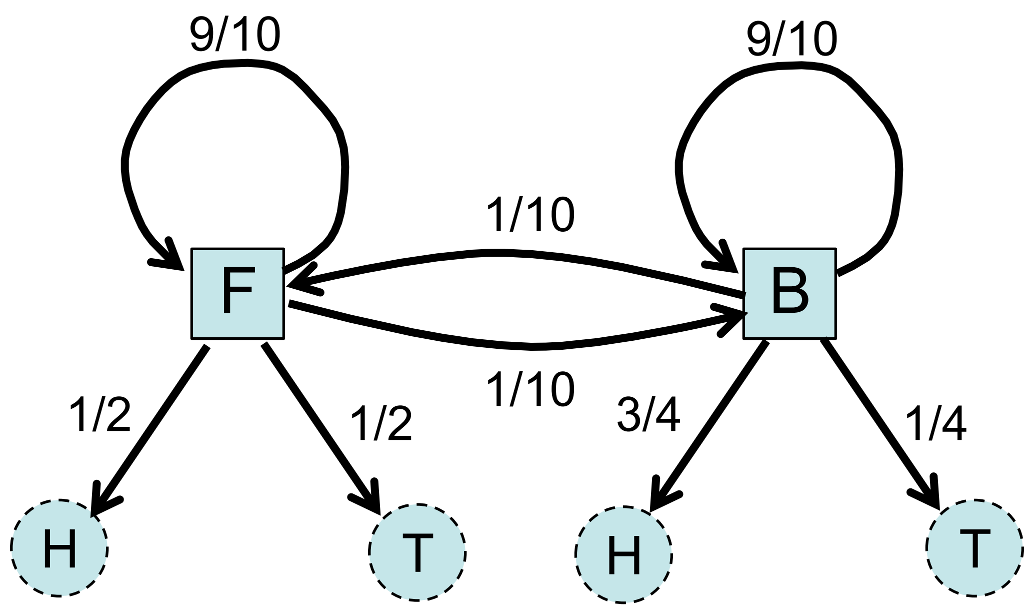
HMM for Fair Bet Casino

- The *Fair Bet Casino* in *HMM* terms:
  $\Sigma = \{0, 1\}$ (0 for ***T***ails and 1 ***H***eads)
  Q = {*F*,*B*} – *F* for Fair & *B* for Biased coin.
- Transition Probabilities *A*, Emission Probabilities *E*

A	Fair	Biased
Fair	0.9	0.1
Biased	0.1	0.9

E	Tails(0)	Heads(1)
Fair	½	½
Biased	¼	¾

HMM for Fair Bet Casino (cont'd)



HMM model for the *Fair Bet Casino* Problem

Hidden Paths

- A *path* $\pi = \pi_1 \ldots \pi_n$ in the HMM is defined as a sequence of hidden states.
- Consider path π = FFFBBBBBFFF and sequence x = 01011101001

x =	0	1	0	1	1	1	0	1	0	0	1
π =	F	F	F	B	B	B	B	B	F	F	F
$P(x_i \mid \pi_i)$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{3}{4}$	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{3}{4}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
$P(\pi_{i-1} \rightarrow \pi_i)$	$\frac{1}{2}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{9}{10}$	$\frac{9}{10}$

$P(x_i \mid \pi_i)$: Probability that x_i was emitted from state π_i

$P(\pi_{i-1} \rightarrow \pi_i)$: Transition probability from state π_{i-1} to state π_i

$P(x \mid \pi)$ Calculation

- $P(x \mid \pi)$: Probability that sequence x was generated by the path π:

$$P(x \mid \pi) = P(\pi_0 \rightarrow \pi_1) \cdot \prod_{i=1}^{n} P(x_i \mid \pi_i) \cdot P(\pi_i \rightarrow \pi_{i+1})$$

$$= a_{\pi_0, \pi_1} \cdot \prod e_{\pi_i}(x_i) \cdot a_{\pi_i, \pi_{i+1}}$$

Decoding Problem

- **Goal:** Find an optimal hidden path of state transitions given a set of observations.
- **Input:** Sequence of observations $x = x_1 \ldots x_n$ generated by an HMM $M(\Sigma, Q, A, E)$
- **Output:** A path that maximizes $P(x \mid \pi)$ over all possible paths π.

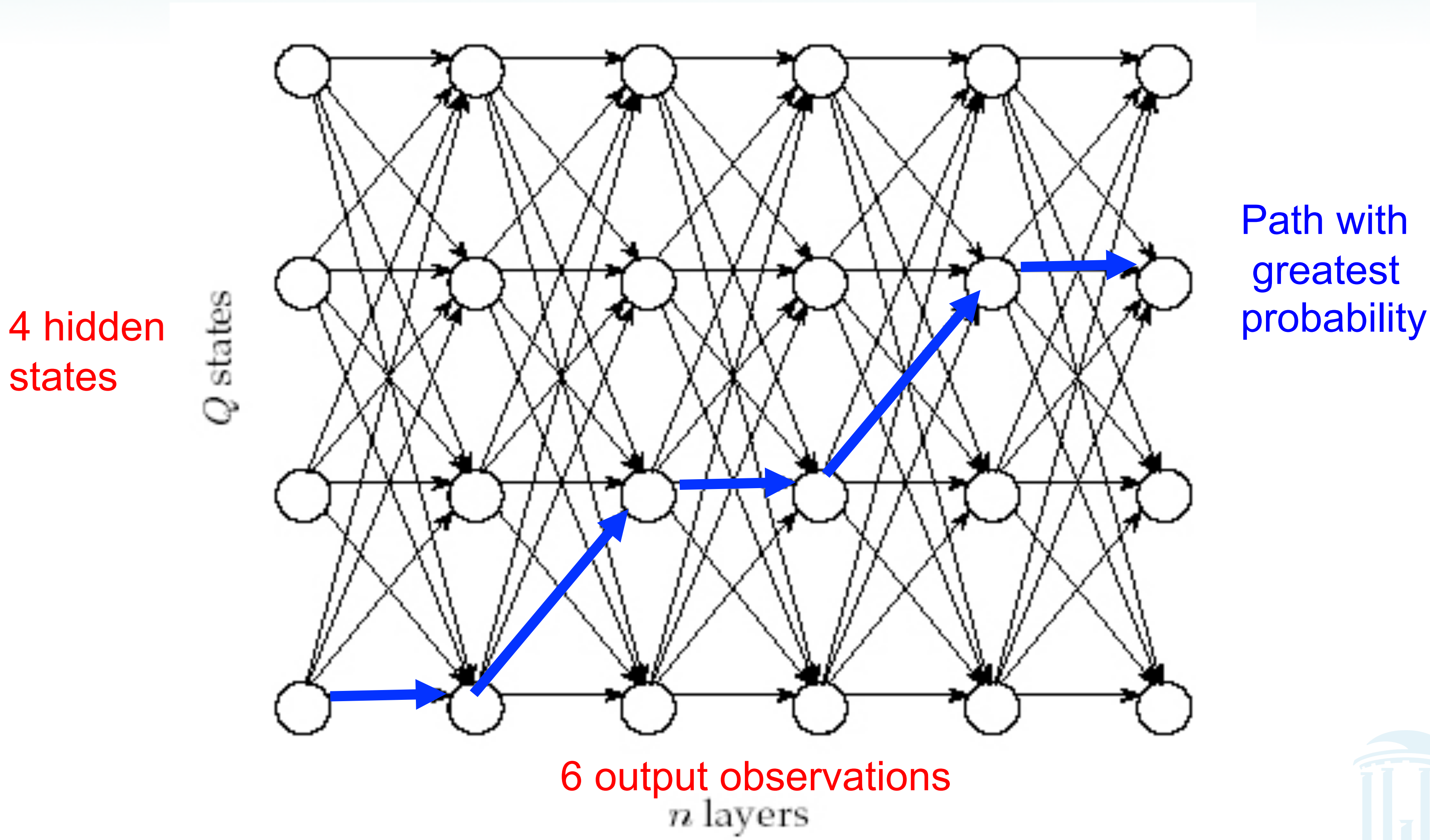
How do we solve this?

- Brute Force
  - Approach:
    - Enumerate every possible path
    - Compute $P(x_{1..n} | \pi_{1..n})$ for each one
    - Keep track of the most probable path
  - How many possible paths are there for *n* observations?
- Is there a better approach?
  - Break the paths in two parts, $P(x_{1..i} | \pi_{1..i})$, $P(x_{i..n} | \pi_{i..n})$
  - $P(x_{1..n} | \pi_{1..n}) = P(x_{1..i} | \pi_{1..i}) \times P(x_{i..n} | \pi_{i..n})$
  - Will less than the highest $P(x_{1..i} | \pi_{1..i})$ ever improve the total probability?
  - Thus to find the maximum $P(x_{1..n} | \pi_{1..n})$ we need find the maximum of each subproblem $P(x_{1..i} | \pi_{1..i})$, for *i* from 1 to n
  - What algorithm design approach des this remind us of?

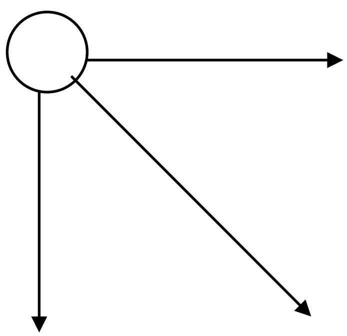
Building Manhattan for Decoding

- Andrew Viterbi developed a "Manhattan-like grid" (Dynamic programming) model to solve the *Decoding Problem*.
- Every choice of $\pi = \pi_1 \ldots \pi_n$ corresponds to a path in the graph.
- The only valid direction in the graph is *eastward*.
- This graph has $|Q|^2(n-1)$ edges.

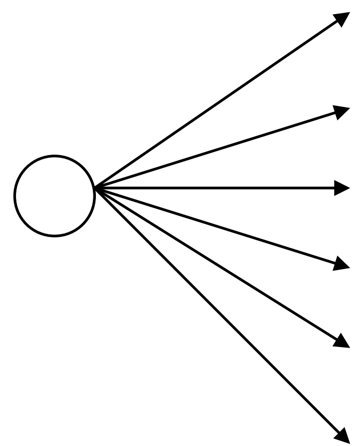
Edit Graph for Decoding Problem



Decoding Problem vs. Alignment Problem



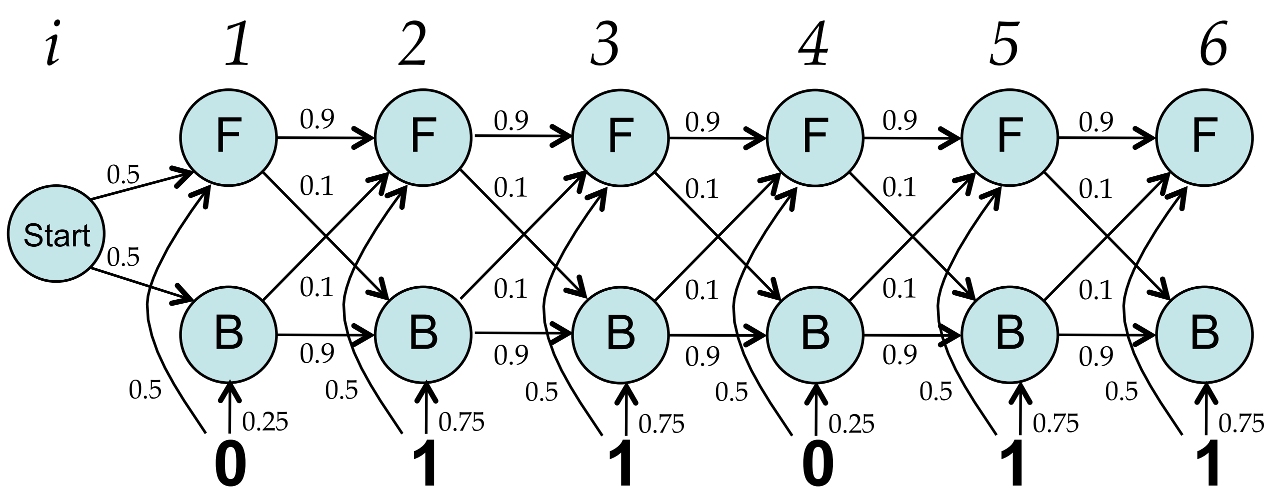
Valid directions in the *alignment problem.*



Valid directions in the *decoding problem.*

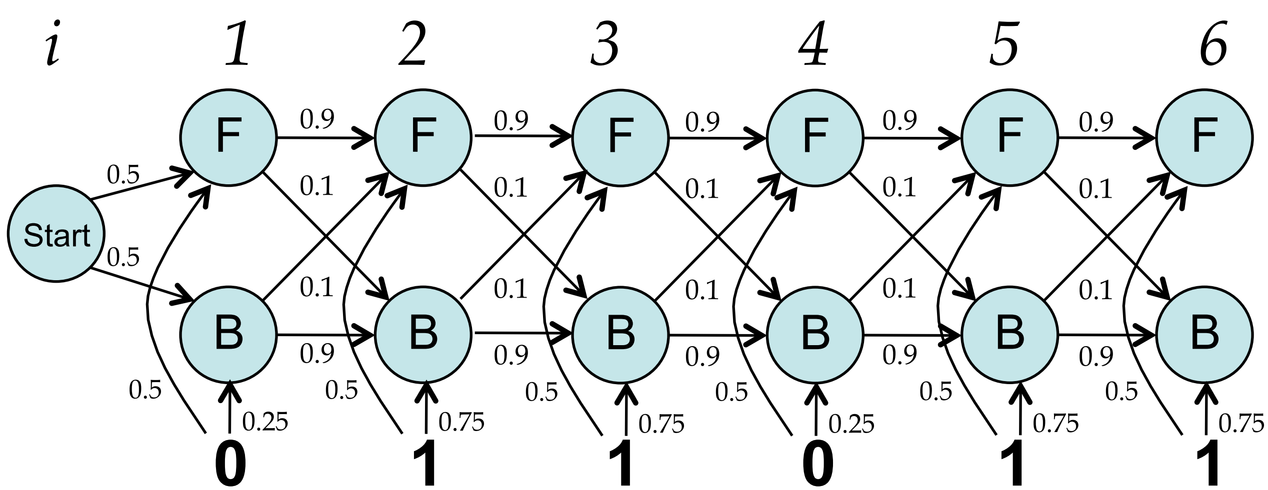
Viterbi Decoding of Fair-Bet Casino

- Each vertex represents a possible state at a given position in the output sequence
- The observed sequence conditions the likelihood of each state
- Dynamic programming reduces search space to: $|Q|$+transition_edges×(n-1) = 2+4×5 from naïve 2^6



Decoding Problem

- The *Decoding Problem* is equivalent to finding a longest path in the *directed acyclic graph (DAG),* where "longest" is defined as the product of the probabilities along the path.



Viterbi Decoding Algorithm

- Since the *longest path* is the *product* of edges' weights, if we use the log of the weights we can make it a sum again!
- The value of the product can become extremely small, which leads to underflow.
- Logs avoid underflow (precision loss due to adding numbers of vastly different magnitudes)

$$s_{k,i+1} = \log e_l(x_{i+1}) + \max_{k \in Q} \{s_{k,i} + \log(a_{kl})\}$$

Viterbi Decoding Problem (cont'd)

- Every path in the graph has the probability $P(x \mid \pi)$.
- The Viterbi decoding algorithm finds the path that maximizes $P(x \mid \pi)$ among all possible paths.
- The Viterbi decoding algorithm runs in $\boldsymbol{O(n|Q|^2)}$ time (length of sequence times number of states squared).
- The Viterbi decoding algorithm can be efficiently implemented as a *dynamic program*

Dynamic Program's Recursion

$s_{l,i+1} = \max_{k \in Q} \{s_{k,i} \cdot \text{weight of edge between } (k,i) \text{ and } (l,i+1)\}$

$= \max_{k \in Q} \{ s_{k,i} \cdot a_{kl} \cdot e_l (x_{i+1}) \}$

$= e_l (x_{i+1}) \cdot \max_{k \in Q} \{s_{k,i} \cdot a_{kl}\}$

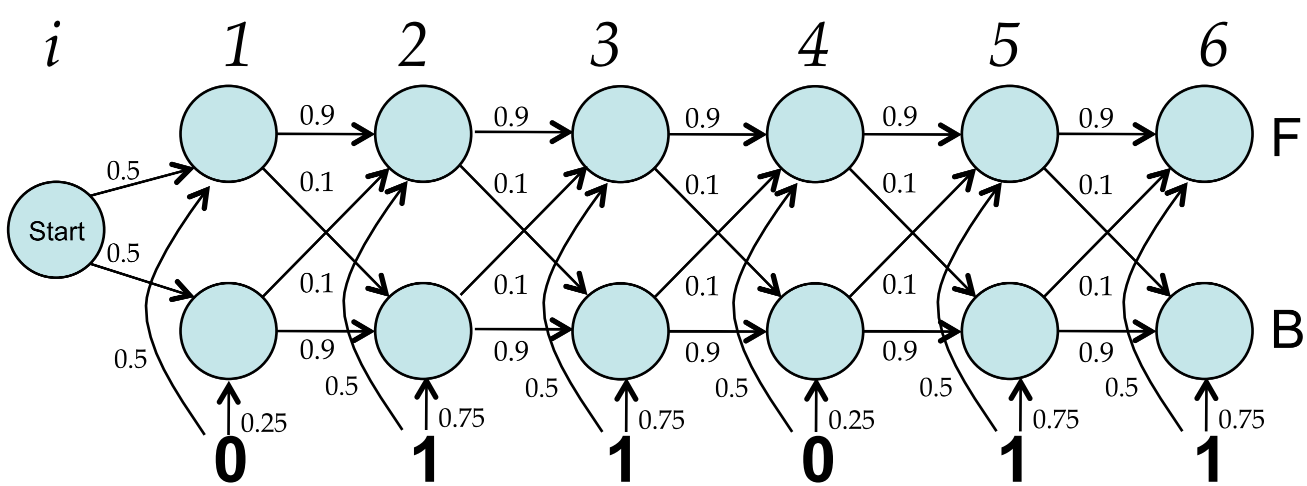
Decoding Problem (cont'd)

- Initialization:
  - $a_{start,k} = 1/|Q|$
  - $s_{k,0} = 0$ for $k \neq begin$.
- Let π^* be the optimal path. Then,

$$P(x|\pi^*) = \max_{k \in Q} \{s_{k,n} \cdot a_{k,end}\}$$

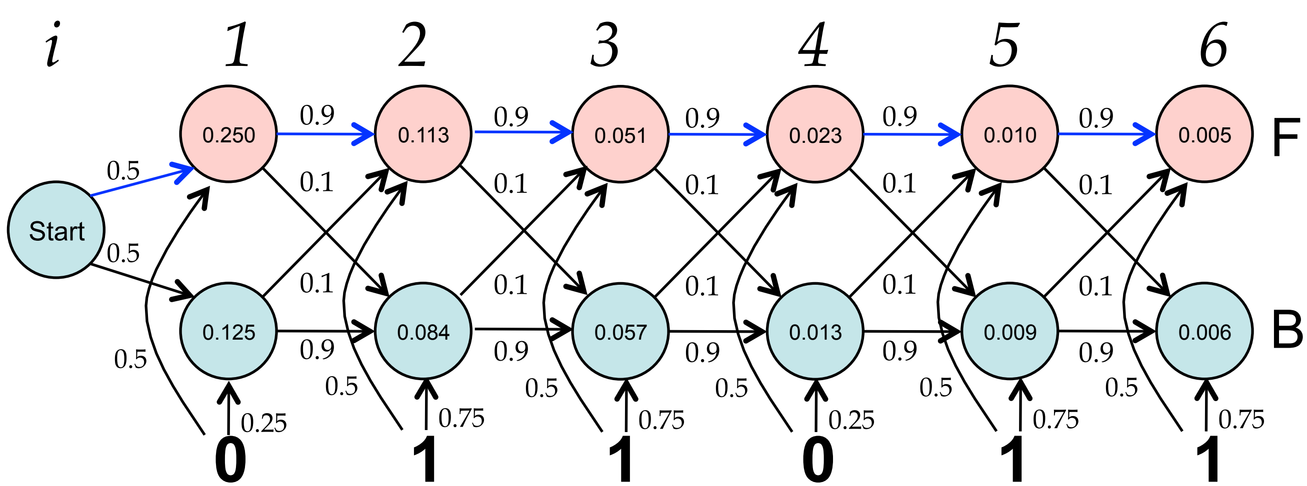
Viterbi Example

- Solves all subproblems implied by emitted subsequence
- How likely is the best path?
- What is it?



Viterbi Example

- Solves all subproblems implied by emitted subsequence
- How likely is the best path? 0.006
- What is it? BBBBBB



How likely is most likely?

- The "most likely path" may not be a lot more likely than a 2nd or 3rd most likely paths (more so in more realistic cases than this one).
- Actual probability of the "most likely path" is not that high.
- Are there better questions we could ask?

0.0058 BBBBBB	0.0001 BBBFFB	0.0000 FFFBFF	0.0000 FBBFBF
0.0046 FFFFFF	0.0001 FFFFBF	0.0000 FFBFBB	0.0000 BFBBFF
0.0013 FBBBBB	0.0001 FFBFFF	0.0000 FBFFBB	0.0000 BFFBBF
0.0012 FFFFBB	0.0001 FBFFFF	0.0000 FBBFFB	0.0000 BBFBFF
0.0009 FFBBBB	0.0001 FFBBBF	0.0000 FFBFFB	0.0000 FFBFBF
0.0008 FFFFFB	0.0001 BFFFBB	0.0000 FBFFFB	0.0000 FBFFBF
0.0006 FFFBBB	0.0001 FBBBFF	0.0000 FBFBBB	0.0000 BFFBFF
0.0006 BBBFFF	0.0001 BBFFFB	0.0000 FBBBFB	0.0000 BFBFBB
0.0004 BBBBBF	0.0000 BFBBBB	0.0000 BBBFBF	0.0000 FBFBBF
0.0004 BBFFFF	0.0000 BBBBFB	0.0000 FFBBFB	0.0000 BFBFFB
0.0003 BBBBFF	0.0000 BBFBBB	0.0000 BBFFBF	0.0000 FBFBFF
0.0003 BFFFFF	0.0000 BFFFFB	0.0000 BFFFBF	0.0000 BFBBFB
0.0001 BBBFBB	0.0000 FFFBBF	0.0000 BFBFFF	0.0000 BBFBFB
0.0001 FBBFFF	0.0000 FFBBFF	0.0000 FFFBFB	0.0000 BFFBFB
0.0001 FBBBBF	0.0000 FBBFBB	0.0000 BFBBBF	0.0000 FBFBFB
0.0001 BBFFBB	0.0000 BFFBBB	0.0000 BBFBBF	0.0000 BFBFBF

More Focused Question

- Are there common aspects of the most likely solutions?
- Which coin was I most likely using on the 4th roll

0.0058 BBBBBB	0.0001 BBBFFB	0.0000 FFFBFF	0.0000 FBBFBF
0.0046 FFFFFF	0.0001 FFFFBF	0.0000 FFBFBB	0.0000 BFBBFF
0.0013 FBBBBB	0.0001 FFBFFF	0.0000 FBFFBB	0.0000 BFFBBF
0.0012 FFFFBB	0.0001 FBFFFF	0.0000 FBBFFB	0.0000 BBFBFF
0.0009 FFBBBB	0.0001 FFBBBF	0.0000 FFBFFB	0.0000 FFBFBF
0.0008 FFFFFB	0.0001 BFFFBB	0.0000 FBFFFB	0.0000 FBFFBF
0.0006 FFFBBB	0.0001 FBBBFF	0.0000 FBFBBB	0.0000 BFFBFF
0.0006 BBBFFF	0.0001 BBFFFB	0.0000 FBBBFB	0.0000 BFBFBB
0.0004 BBBBBF	0.0000 BFBBBB	0.0000 BBBFBF	0.0000 FBFBBF
0.0004 BBFFFF	0.0000 BBBBFB	0.0000 FFBBFB	0.0000 BFBFFB
0.0003 BBBBFF	0.0000 BBFBBB	0.0000 BBFFBF	0.0000 FBFBFF
0.0003 BFFFFF	0.0000 BFFFFB	0.0000 BFFFBF	0.0000 BFBBFB
0.0001 BBBFBB	0.0000 FFFBBF	0.0000 BFBFFF	0.0000 BBFBFB
0.0001 FBBFFF	0.0000 FFBBFF	0.0000 FFFBFB	0.0000 BFFBFB
0.0001 FBBBBF	0.0000 FBBFBB	0.0000 BFBBBF	0.0000 FBFBFB
0.0001 BBFFBB	0.0000 BFFBBB	0.0000 BBFBBF	0.0000 BFBFBF

Next Time

- How do we get at these other questions about an HMM?
- What if my observations are corrupted (i.e. there is noise in my observed sequence)?
- What if I don't know the parameters of my HMM model (emission, transition, and observation noise probabilities)?
- Leave the Casino to find a biological application