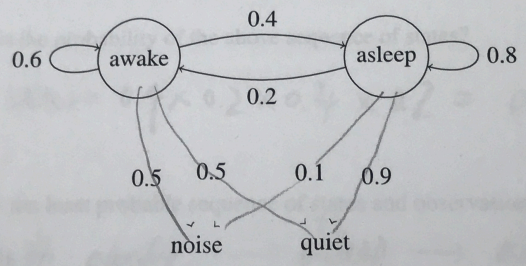
Markov chains. A Markov chain is a state machine which consists of the following:

1. A set of states $Q = \{q_1, \ldots, q_n\}$.
2. A transition probability matrix A, where each a_{ij} represents the probability of transitioning from state q_i to state q_j, such that for each i, $\sum_{j=1}^{n} a_{ij} = 1$.
$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$
3. A set of possible observations $V = \{v_1, \ldots, v_m\}$.
4. An emission probability matrix B, where each b_{ij} represents the probability of state q_i emitting the observation v_j, such that for each i, $\sum_{j=1}^{m} b_{ij} = 1$.
$$B = \begin{bmatrix} b_{11} & \cdots & b_{1m} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nm} \end{bmatrix}$$
5. A special start state q_0 which is not associated with observations, along with transition probabilities $a_{01} \cdots a_{0n}$ from the start state to the other states. The start state may be identical to one of the other states.

A Markov chain starts at the start state q_0, and at each time point $t_1, t_2 \ldots$ performs a transition and emits an observation. The *Markov property* states that the probability of being in a particular state at time t_i depends only on the previous state (that is, the state at t_{i-1}), and the probability of an observation at time t_i depends only on the current state (that is, the state at t_i).

Problem 1. Consider a Markov chain that represents the probability that a child left alone in her room will be awake or asleep. There are two states {awake, asleep}, and two possible observations coming from the room {noise, quiet}. The transition and emission probabilities are noted in the following diagram: transitions are shown with solid arrows, and emissions with dashed arrows. (Note that the diagram is identical to the one discussed in class, but the probabilities are different!)



The child starts by being awake, and remains in the room for 4 time points, $t_1 \ldots t_4$ (4 iterations of the Markov chain).

a. (1 point) What is the most probable sequence of states for $t_1 \ldots t_4$?

Start with awake and asleep → asleep → asleep → asleep
(t_1 t_2 t_3 t_4)

b. (1 point) What is the probability of the above sequence of states?

The above sequence has probability 0.2048

$0.4 \times 0.8 \times 0.8 \times 0.8 = 0.2048$

c. (1 point) What is the most probable sequence of states and observations?

Based on given probability, the sequence is below

Start with awake → asleep → asleep → asleep → asleep
quiet quiet quiet quiet

d. (1 point) What is the probability of the above sequence of states and observations?

$P(\text{observed states}) = \underbrace{0.4 \times 0.9}_{t_1} \underbrace{\times 0.8 \times 0.9}_{t_2} \underbrace{\times 0.8 \times 0.9}_{t_3} \underbrace{\times 0.8 \times 0.9}_{t_4} =$

e. (1 point) What is the least probable sequence of states?

Start with awake state then asleep → awake → asleep → awake

f. (1 point) What is the probability of the above sequence of states?

$P(\text{least prob state}) = 0.4 \times 0.2 \times 0.4 \times 0.2 = 0.0064$

g. (1 point) What is the least probable sequence of states and observations?

Start with awake, → asleep → asleep → asleep → asle
(t_1)
noise noise noise noise

h. (1 point) What is the probability of the above sequence of states and observations?

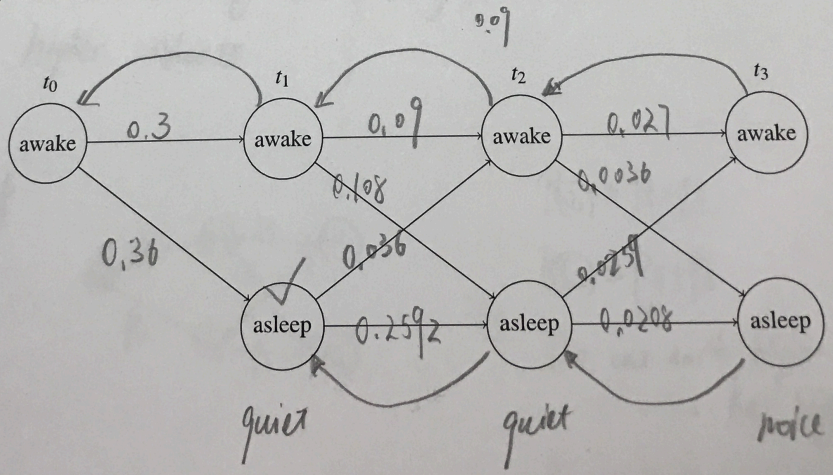
$0.4 \times 0.1 \times 0.8 \times 0.1 \times 0.8 \times 0.1 \times 0.8 \times 0.1 = 0.00002048$

The Viterbi algorithm. A Hidden Markov Model (HMM) is a Markov chain where we cannot observe the states directly, but we can observe the emissions. The Viterbi algorithm is used for decoding the sequence of states, that is finding the most likely sequence of states that could give rise to a sequence of observations. Given a set of states Q and a sequence of time points $1 \dots T$, the algorithm builds two matrices of size $Q \times (1 \dots T)$: a probability matrix representing the probability of each state at each time point, and a backpointer matrix which points from each state at each time point to the most likely previous state. At the final time point T, the algorithm selects the state with the highest probability, and returns the path of backpointers from that state, representing the most likely sequence of states to give rise to the observations. The following is pseudocode for the algorithm: the notation $a(q',q)$ represents the transition probability between states q' and q, and $b(q,o_t)$ represents the emission probability by state q of the observation noted at time t.

 # Initialization step at $t = 1$
for q **in** Q :
    probability$(q,1) = a(q_0,q) * b(q,o_1)$
    backpointer$(q,1) = q_0$
 # Recursion step for the remaining time points
for t **from** 2 **to** T :
    for q **in** Q :
        probability$(q,t) = \max_{q' \in Q}$ probability$(q',t-1) * a(q',q) * b(q,o_t)$
        backpointer$(q,t) = \arg\max_{q' \in Q}$ probability$(q',t-1) * a(q',q)$
 # Termination step
most_probable_state$(T) = \arg\max_{q' \in Q}$ probability(q',T)
return the backtrace path by following the backpointers from the most probable state

Problem 2. Consider the same Markov chain from problem 1, this time as a hidden Markov model. The child starts by being awake, and remains in the room for 3 time points, $t_1 \dots t_3$ (3 iterations of the Markov chain). The observations are: quiet, quiet, noise.

a. (6 points) Using the Viterbi algorithm, identify the most likely sequence of states that would give rise to these observations. Show your work.



Most likely
Awake → Awake → Awake → Awake
P = 0.027
0.2592

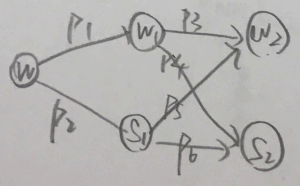
b. (5 points) After the first iteration (at t_1), the sum of the probabilities is less than 1. Why? Where is the rest of the probability mass?

The cases like awake + noise
asleep + noise

c. (6 points) Suppose we are not interested in decoding the sequence of states (that is, whether the child was awake or asleep at each point), but only in the overall most likely state at the end (that is, whether the child is awake or asleep at the end). Obviously we can remove the backpointer lines from the Viterbi algorithm; however, this would still give us the probability of only the most likely path to each end state. What additional change can we make to the algorithm so that instead of giving us the probability of the most likely path to each state at each time, it will give the overall probability of being at each state at each time? Explain why.

Compute the sum of probabilities that come into state at different time stamps, the one with the bigger Probability sum has the higher chance.

eg.



$P(W_2) = P_3 + P_5$

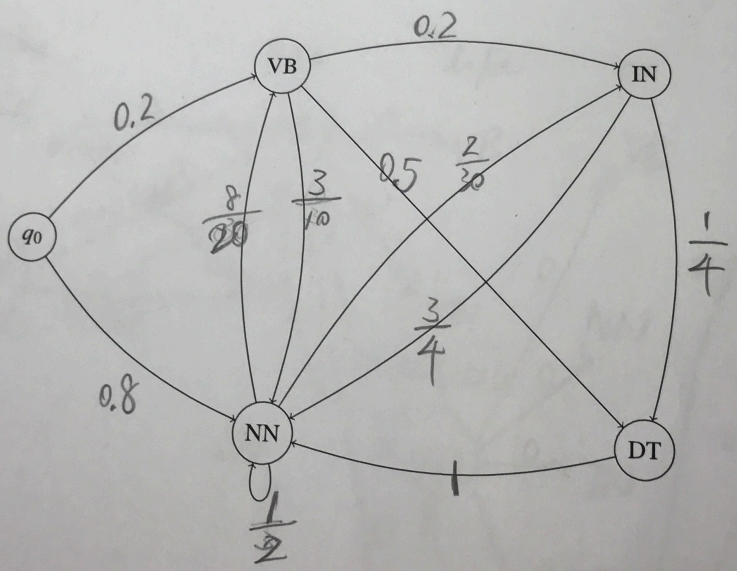
$P(S_2) = P_4 + P_6$

The one with higher probabilities chances has bigger chances

Problem 3. In part-of-speech tagging, we learn a hidden Markov model from a tagged corpus: each state is a part-of-speech tag, transition probabilities are the conditional probabilities of tags given the previous tag, observations are words, and emission probabilities are the conditional probabilities of words given tags. The start state is the beginning of a sentence, which is not a part-of-speech tag. In this problem we will look at some data that will help us tag the sentence *Time flies like an arrow*. We will use the following sentences as a corpus of training data (the notation word/TAG means word tagged with a specific part-of-speech tag).

1. eat/VB breakfast/NN at/IN morning/NN time/NN
2. take/VB time/NN with/IN arrow/NN projects/NN
3. horse/NN riders/NN like/VB the/DT airport/NN
4. paper/NN flies/VB on/IN hydrogen/NN gas/NN
5. bees/NN sting/VB like/IN some/DT flies/NN
6. beans/NN soil/VB an/DT iron/NN grill/NN
7. flies/NN smell/VB an/DT arrow/NN drink/NN
8. people/NN like/VB an/DT army/NN arrow/NN
9. dinner/NN time/NN flies/VB all/DT day/NN
10. horse/NN flies/NN time/VB morning/NN rays/NN

a. (5 points) Based on the corpus, fill in the transition probabilities in the state chart below.



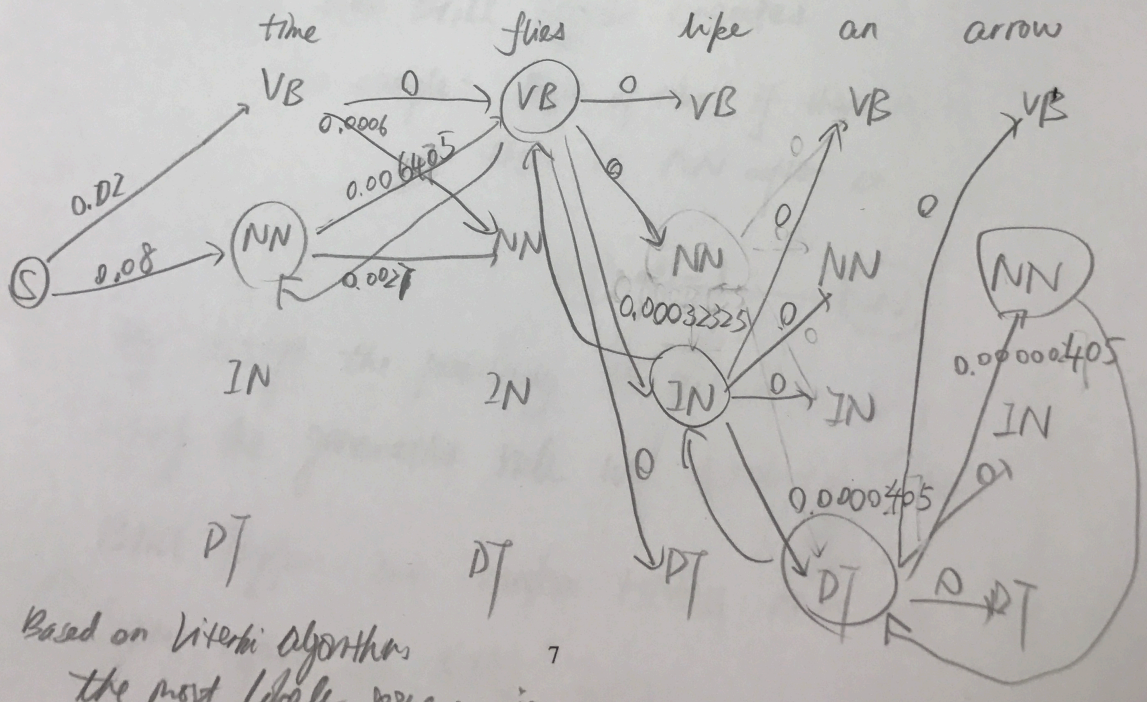
tag

word

b. (5 points) Fill in the emission probabilities for the 4 states; only write down the probabilities for the words *time flies like an arrow*.

	time	flies	like	an	arrow
VB	1/10	2/10	2/10	0/10	0/10
NN	3/30	3/30	0/30	0/30	3/30
IN	0/4	0/4	1/4	0/4	0/4
DT	0/6	0/6	0/6	3/6	0/6

c. (7 points) Now use the Viterbi algorithm with the above model to tag the sentence *Time flies like an arrow*. What is the most likely tag sequence, based on the training data? Show your work.



Based on Viterbi algorithm
the most likely sequence is
NN VB IN DT NN
Time flies like an arrow

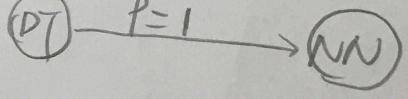
Problem 4. In the Brill tagger, the initial tagger gives the most likely tag for each word, regardless of context. Suppose we wanted to use a hidden Markov model to give the same effect: that is, a hidden Markov model that will result in the most likely tag for each word, regardless of context. How would we set the transition probabilities? How would we set the emission probabilities? Why? (8 points)

To convert HMM to BRILL Tagger.

we set the emission probability as P(word | category) as in Problem 3. eg. P(time | noun) = 3/30

set the transition probability between different tags as either 0 or 1 depending on the principle rule Brill Tagger creates

For example: Most of time, if there is a DT there is NN after it.



By assign the probability to either 0 or 1 using the generative rule we developed from Brill Tagger. we make HMM have the same effect as BRILL Tagger