Natural Language Processing

Prof. Pushpak Bhattacharyya
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay

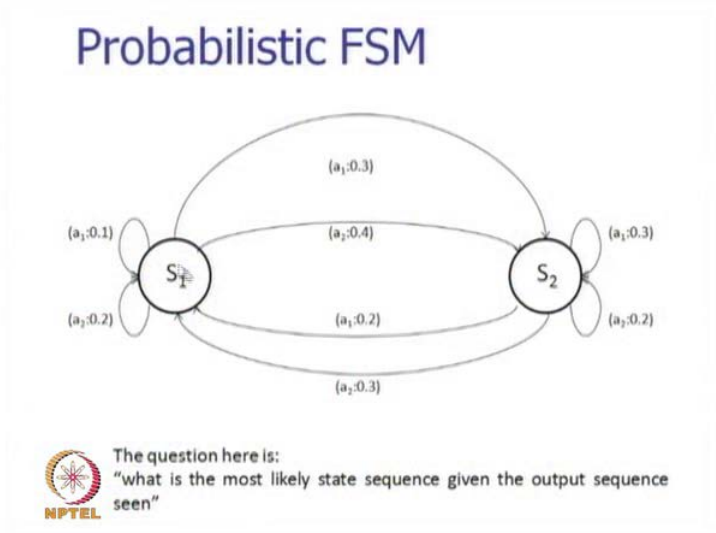
Lecture - 19
HMM, Viterbi, Forward Backward Algorithm

Today, we will discuss the HMM and it is algorithm aspects, we have seen that HMM is a very critical machine, very useful machine for statistical natural language processing. And our goal would be to understand the algorithm aspect to this machine, we remark that there are three important problems that are solve by hidden Markov model. The first is that we would like to compute efficiently, the probability of the observation sequence.

The next problem which is most important is that we would like to compute, the probability of the state sequence, the best possible state sequence given the observation sequence. So, we would like to find out the best possible state sequence, given the observation sequence and the meaning of best possible here is the sequence has the highest probability, given the observation sequence.

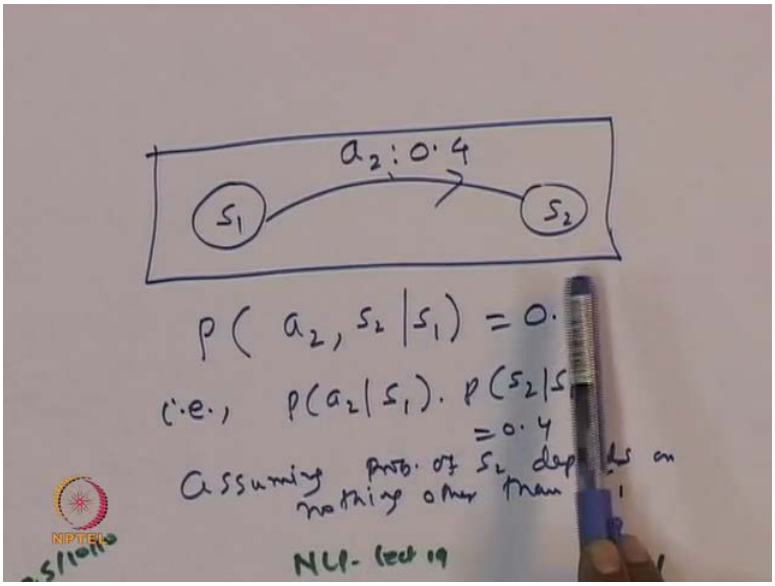
And the third problem is to find out the transition and observation probabilities or parameters of hidden Markov model, given the output sequence or the observation sequences, which are accepted by the machine or generated by the machine. So, we concentrate on the second problem now, to find out the best possible state sequence in terms of highest probability value given the observation sequence.

(Refer Slide Time: 01:55)



So, we first start with this particular diagram, which we called a probabilistic finite state machine. This is nothing but the hidden Markov machine of a order k equal to 1, that is why, we can draw this kind of states with only one state symbol unit S 1 going to S 2 on symbol a 2. If I look at this arc for example, this means that the probability of going to state S 2 with output symbol a 2 is 0.4. So, this you would remember is very definite probability.

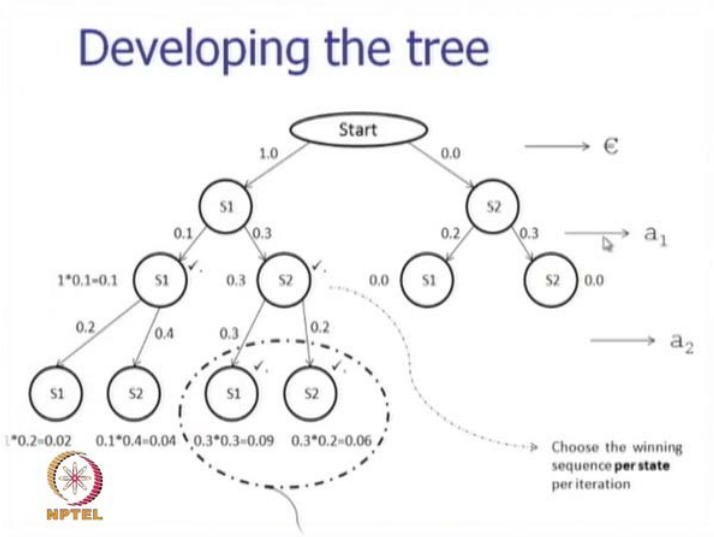
(Refer Slide Time: 02:53)



We look at the paper and I would written it for you, the paper shows the meaning of the probability of a 2 being 0.4 on the arc going from to S 1 to S 2, the meaning of this probability value is p a 2 comma s 2 given s 1 is 0.4. Now, this particular probability can be written as p a 2 given s 1 into p s 2 given s 1, which is equal to 0.4 and the next probability that is p s 2 given s 1 the second probability expression; mix this independence assumptions that the probability of state depends, only one the pervious state and nothing else. So, the meaning of this particular picture is probability of a 2 given s 1 into probability of s 2 given s 1.

So, going back to the slides we have this machine with us and the question that is been asked is, what is the most likely state sequence given the output seen. We will take a particular output sequence, which is a 1, a 2, a 1, a 2 or output sequence will be a 1, a 2, a 1, a 2. Now, we would like to predict the best possible state sequence, corresponding to these observation sequence. So, we are going to find out the best possible state sequence, in terms of s 1 and given the observation sequence a 1, a 2 all right.

(Refer Slide Time: 04:26)



So, here is the tree which clearly depicts what we have to do. So, initially the system is in the start state and from there with probability 1.0, it goes to state S 1 and the probability is 0.0 goes to S 2, which means the machine starts from state S 1. The next symbol is a 1 when a 1 comes in, then from S 2 we go to S 1 with probability 0.2 from S 2 to S 2 with

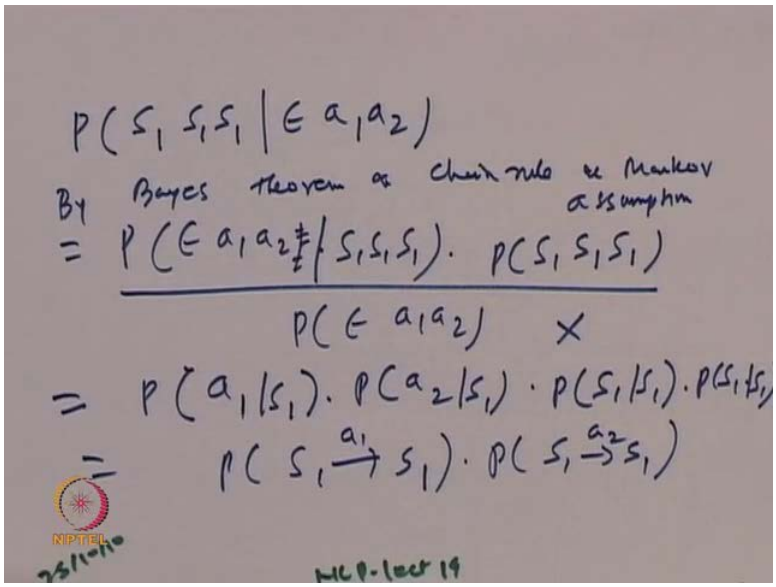
probability 0.3. So, this is as per the finite state machine, which we has shown before with probability value on it.

Now, what will happen is that, this probability will get multiplied by 0 and therefore, the probability of state sequence S 2, S 1 and S 2, S 2 will be equal to 0. So, we need not consider these probabilities and these notes anymore, there is no point advancing these notes, they will always get multiplied by 0 and the probability value will be 0. So, we concentrate on these part of the tree and from S 1 here on a 1 we see that the probability of the state transition from S 1 to S 1 is 0.1 on a 1, S 1 to S 2 is 0.3 on a 1 the probability of sequence S 1, S 1 on the observation sequence epsilon a 1 is 1 into 0.1, which is the product of these probabilities and sequence probability here is 0.3.

So, S 1, S 2 sequence probability given the observation sequence epsilon on a 1 is 0.3 and the sequence probability of S 1, S 1 is 0.1. Next symbol that comes in is a 2 here, we advance the note here S 1 to S 1 and S 2 these note S 2 is also advanced to S 1 and S 2. The probability values on the arts are 0.2, 0.4, 0.3 and 0.2, meaning thereby, S 1 on a 2 goes to S 1 with probability 0.2, S 1 and a 2 goes to S 2 with probability 0.4, S 2 to S 1 is 0.3, S 2 to S 2 is 0.2, these we are seen before in the probabilistic finite state machine.

Now, when we look at the sequences S 1, S 1, S 1 the probability of that comes out to be equal to 0.1, which is the accumulated probability of the subsequence S 1, S 1 into the transition probability which is 0.2 on a 2 and the probability value now becomes 0.1 into 0.2 which is 0.02. So, the probability of that sequences S 1, S 1, S 1 on a 1, a 2 is 0.02. Similarly, the probability of sequences S 1, S 1, S 2 on observation sequence a 1, a 2 is 0.04 that of S 1, S 2, S 1 is 0.09 that of S 1, S 2, S 2 is 0.06. Now, it is quite easy to see that what is have to doing is correct, we will do a slight amount of mathematical calculation to show the our theory is fine, as far as doing this mathematical task is concern.

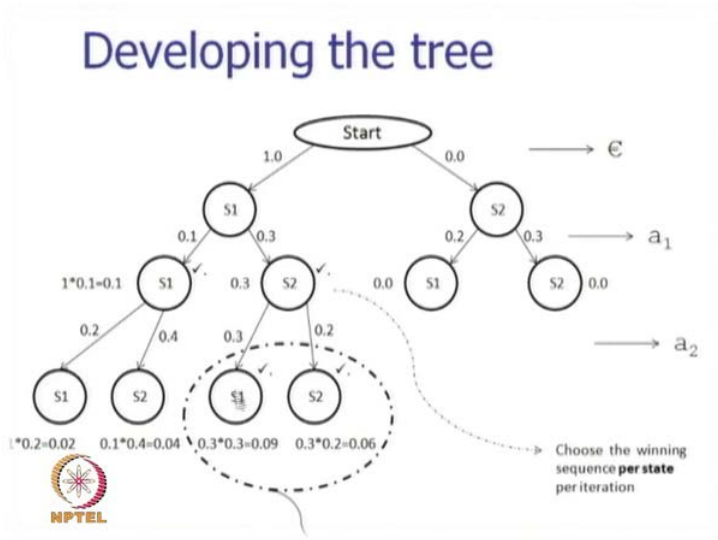
(Refer Slide Time: 08:07)



So, we are saying that we would like to compute the probability of s 1, s 1, s 1 given the sequence epsilon a 1, a 2. Now, by chain rule this is probability by Bayes theorem and chain rule and Markov assumption, we get this as p epsilon a 1, a 2 given s 1, s 1, s 1 into probability of s 1, s 1, s 1 and the denominator is probability epsilon a 1, a 2. So, this denominator is not computed, because this probability values are useful comparison and they are not use, because the denominator is same for all the notes which are compared.

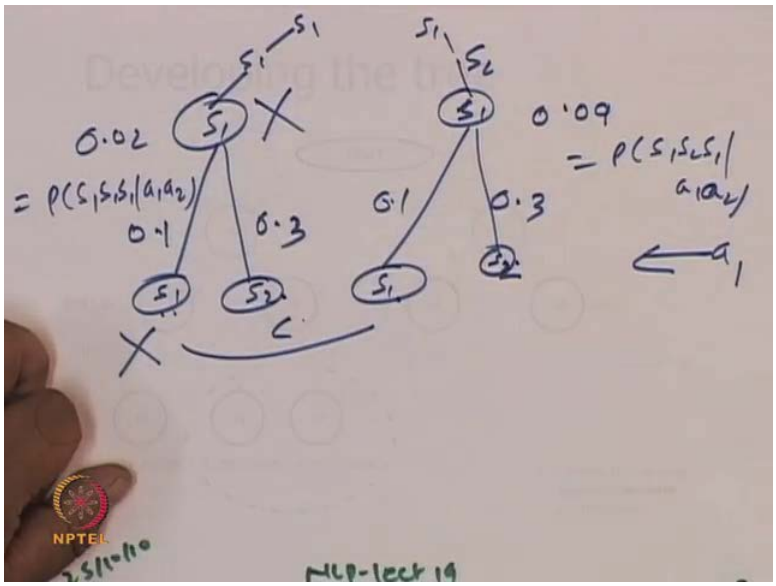
So, this will be equal to probability of a 1 on s 1, into probability of a 2 on s 1, into probability of s 1 on s 1, into probability of s 1 on s 1 and that is equal to probability of s 1 on s 1 on a 1 into probability of s 1 to s 1 on a 2. So, we can take product of these two probabilities and compute the probability of the sequences, this is due the application of Bayes theorem, chain rule and Markov assumption.

(Refer Slide Time: 09:59)



So, looking at slide once again, we see that the sequence probability can be found out, simply by the multiplying the transition probability with the accumulated probability, so far. Now, when we do this and when we have got this four children, ending in S 1, S 2, S 1, S 2 then we compare those sequences which end in the same state. So, sequences ending in S 1 are S 1, S 1, S 1 and S 1, S 2, S 1 the probability of S 1, S 2, S 1 is 0.09 that of S 1, S 1, S 1 is equal to 0.02. Now, we observe that the children of this S 1 will never be able to win over the children of S 1 coming from here, so this again we can illustrate by writing.

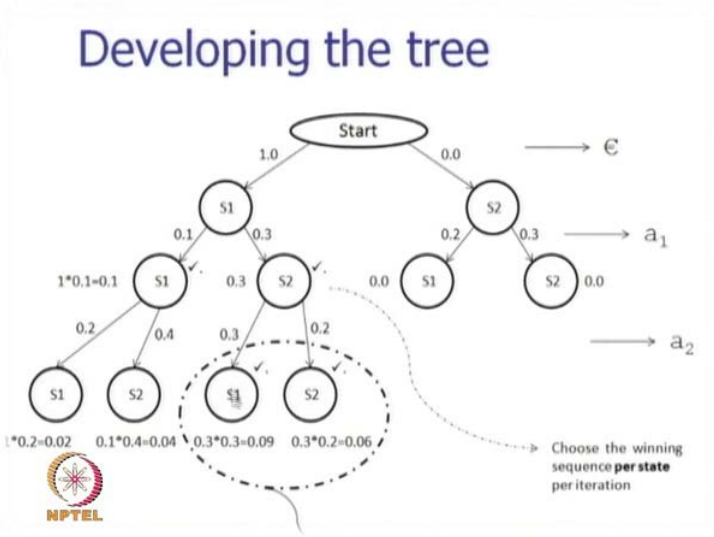
(Refer Slide Time: 11:02)



So, we have an S 1 here and the sequence is s 1, s 1 and another s 1 here, the sequence is s 2, s 1. So, s 1, s 2, s 1 and the accumulated probability here is 0.02, accumulated probability here is 0.09. So, the probability of the sequence s 1, s 1, s 1 is 0.02, probability of the sequence s 1 is to s 2 is 0.09, so these equal to p s 1, s 1, s 1 given a 1, a 2 and this is equal to p s 1, s 2, s 1 given a 1, a 2. Now, when the next symbol comes which is a 1, we will be advancing this state's this will be s 1, s 2, s 1, s 2 and the values here, will be multiplied by the transition probabilities.

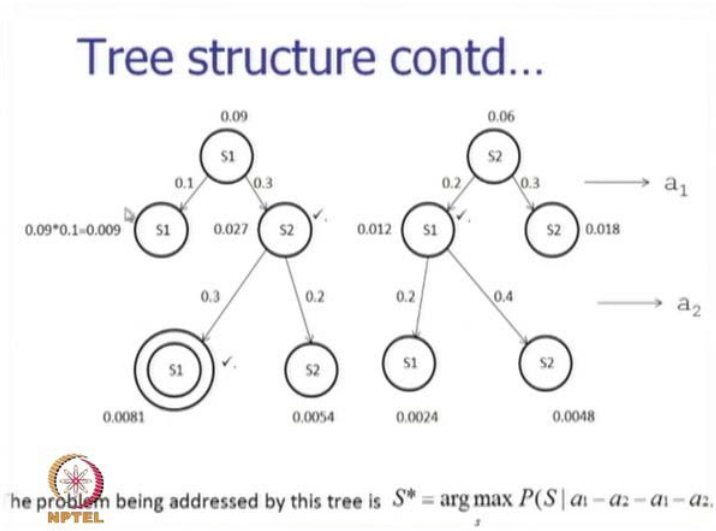
And now, we see that we are always multiplying by the same quantity for a same state transition. And when this happens, this note will always be less than this note, the accumulated probability here will be more than accumulated probability here, similarly the accumulated probability here, will be more than the accumulated probability here. Therefore the children of these notes will never be win over the children of this note therefore, there is no point advancing this note and we cancel this out.

(Refer Slide Time: 12:45)



So, when we cancel this out we look at the slide once again, here this S 1 need not to be advance any more, we have to advance this S 1 this S 2 need not to be advance anymore we have to advance this S 2.

(Refer Slide Time: 13:03)

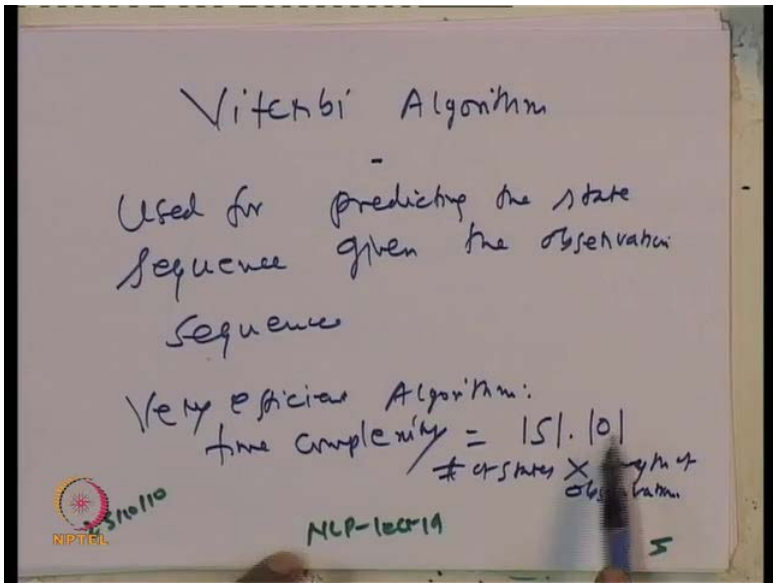


So, going to the next slide we have this S 1, S 2 with their accumulated probabilities 0.09 and 0.06. Now, the next symbol comes in which is a 1, the transition probabilities are has shown here 0.1, 0.3, 0.2, 0.3 accumulated probabilities multiplied by these transition probabilities. The new accumulated probabilities are 0.009, 0.027, 0.012 and 0.018 which is the product of previous accumulated probability and the transition probability alright.

So, now when we look at the sequences ending in the same state. So, here is the sequence ending in S 1, sequence ending in the S 1 here, has higher probability than the sequence ending in the S 1 here. Therefore, we retain this note and this cut this note similarly, this note S 2 probability with 0.27 which is more than 0.018 is retain the other note is discarded and we find that now, we have S 2 and S 1 here, the next symbol is a 2 when a 2 is presented or when a 2 is output.

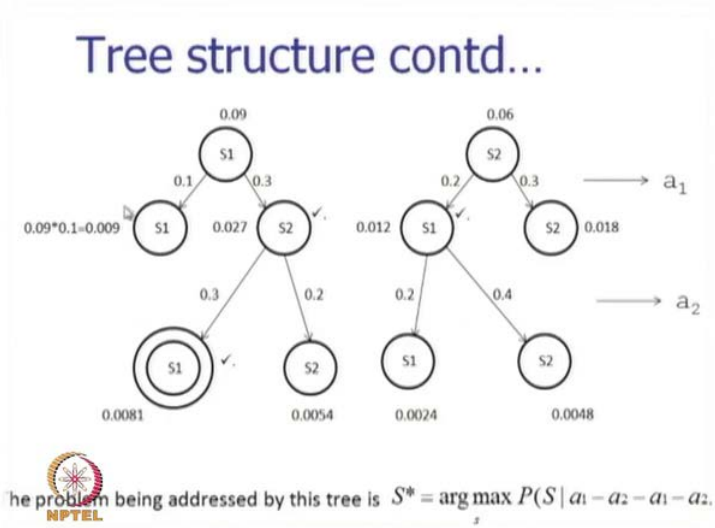
Then, the sequence that results is ending in S 1, S 2, S 1, S 2 with this four notes and this particular note has the highest probability of 0.0081. Therefore, this is the winner note and when work backward, we can recover the state sequence which has the highest probability corresponding to the observation sequence a 1, a 2, a 1, a 2. So, the winner sequence should be S 1, S 1, S 2, S 1, S 2, S 1. So, S 1 is S 2, S 1 is S 2, S 1 is the winner sequence. And this is the best possible state, as far as the output observation sequence is concerned.

(Refer Slide Time: 15:16)



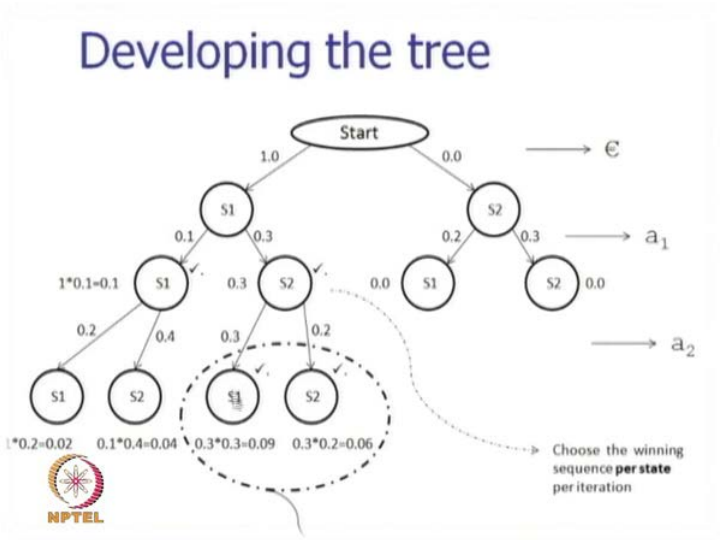
Now, this particular algorithm is known as the viterbi algorithm, this is known as the viterbi algorithm, this is used for predicting the state sequence given the observation sequence. Very efficient algorithm, time complexity equal to number of states into observation length. So, number of states into length of observation is a complexity, so how is the complexity number of states into length of observation.

(Refer Slide Time: 15:56)



So, let us see what is going on in the way the algorithm proceed.

(Refer Slide Time: 16:02)



So, we have seen that initially when the first symbol is obtain, we have two states coming out from S 1 at S 1 and S 2. When we have a next symbol then both S 1, S 2 produce two children each, but only two out this four are retain, other two are discarded. So, if there is k states in the machine then at every level, we retain k states and we discard the others and that is done by, noting all those sequences, which end in the same states and retaining that sequence whose probability is the highest.

So, at every level we retain exactly k states and the numbers of level, is equal to length of the observation sequence. So, numbers of states into length of observation sequence, is the complexity of the algorithm, if we were advancing the all states, at the every level then you can see the complexity would be number of states to the power, length of the observation sequences. So, because of Markov assumption because of the fact that a state is depend only on the previous state, we are able to bring down the complexity from exponential to linear, in the length of the observation sequence. So, these a tremendous amount of saving and that is possible because of dynamic programming. The way the algorithm words is that it makes use of the accumulated probability seen, so far; so these the way algorithm works and the tree based discussion makes it quite loss it.

(Refer Slide Time: 17:53)

Tabular representation of the tree

Latest symbol observed / Ending state	ϵ	a_1	a_2	a_1	a_2
S_1	1.0	(1.0*0.1,0.0*0.2)=(**0.1**,0.0)	(0.02, **0.09**)	(0.009, **0.012**)	(0.0024, **0.0081**)
S_2	0.0	(1.0*0.3,0.0*0.3)=(**0.3**,0.0)	(0.04,**0.0 6**)	(**0.027**,0.018)	(0.0048,0.005 4)

Note: Every cell records the winning probability ending in that state

Final winner

he bold faced values in each cell shows the
equence probability ending in that state. Going backward
rom final winner sequence which ends in state S2 (indicated

Next, we show a tabular representation of the tree which is very useful for computation implementation of the algorithm. Here is the table, which is used as a data structure for the running of the viterbi algorithm. This particular table has a number of columns equal to length of the observation sequences plus 1, which corresponds to epsilon and these particular column with epsilon, indicates what the first state of the system is system goes to S 1 with probability 1.0.

However, if the both the states are possible right at the beginning, then we will distribute this probability between the two rows here, let us understand the next set of columns. Here, the rows indicate the ending states the sequences which ending that particular state for example, this particular row is for all those sequences, which ending state S 1 this is for all those sequences which ending state S 2. Now, when a 1 comes as a symbol, then we have the transition probability of 0.1 and 0.2.

Now, we record at every cell a topple, so these topple is the record of all the accumulated probability of sequences, ending in that particular state. So, the sequences which can ending state S 1 are two in number coming from a previous S 1 or a previous S 2. So, we see here the probability of those sequences are 1.08 into 0.1 and 0.0 into 0.2, so 0.0 comes from this particular cell here, 0.0 is the probability of the sequence ending in S 2 when epsilon is same; that means, we have no symbol is same.

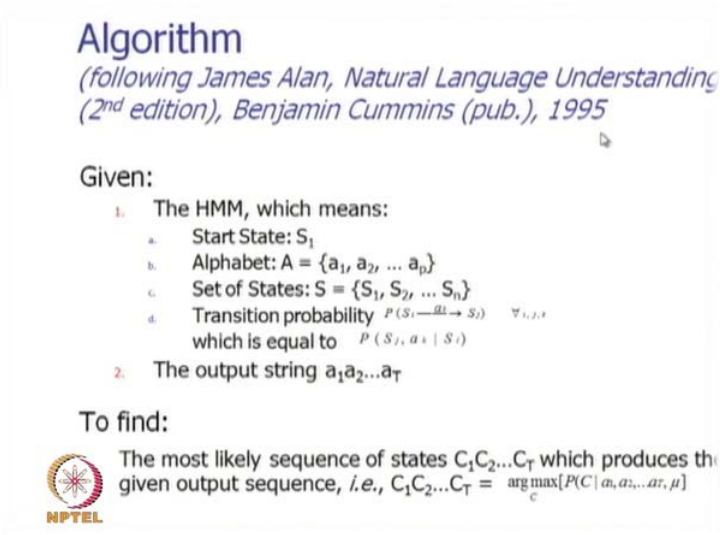
So, when we advance that particular note, we have to multiply the probability of the transition with 0.0 and we get 0.0 here. Similarly, we get 1.08 into 0.1 here and this is bold faced. The meaning of these topple is that from S 1, we get a sequence ending in S 2 with probability value of 0.3 and from S 2 we get a sequence ending in S 2 with probability 0.0, this is the 0.0 alright. The next symbol makes thing much clearer, in a next symbol we find that the topple has 0.02 and 0.09, the topple here has 0.06 and 0.04.

This is 0.09 comes from, the accumulated probability 0.3 multiplied by the transition probability of S 1 to S 2 or rather S 2 to S 1 on the symbol a 2. And this is the winner probability of the sequences ending in S 1, similarly the winner probability here is 0.06 of the sequences ending in S 2. In the next symbol a 1 the topple is 0.009 and 0.012 this is the winner probability here, this is the winner probability finally, 0.0081 is the winner probability here. Now, from this table we can work backwards and recover the states sequence.

So, final state as we see from the bold faced portion here, is S 1 the state sequences ending in S 1 and it came from S 2 because this is the second topple here, it came from S 2. So, we come here this is the bold faced number this is the first component of topples, so this must have come from S 1, so we go to this cell. Now, the bold faced is number is this which is second components, so it must have come from S 2. So, we come here and then the sequence comes out to be equal to S 1 to S 2 to S 1 to S 2 to S 1.

So, this is the way the computation proceed with the observation sequence and after the computation is over, we can recover from the data structure the whole state sequences. The bold faced value in each cell, shows the sequence probability ending in that states going backward from final winner sequence, which end states in S 2 indicate going to the second topple we recover the sequence. So, this particular tabular representation which is the data structure corresponding to the working of the viterbi algorithm, is very convenient to store the computation and also the finally, recover the path.

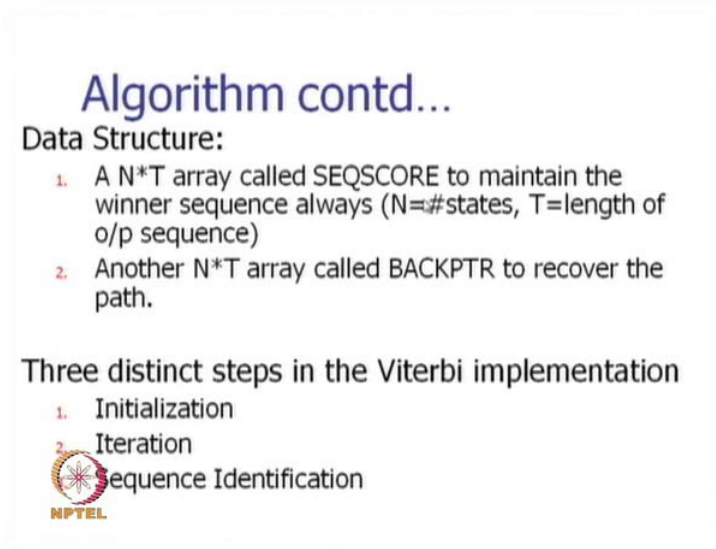
(Refer Slide Time: 23:18)



So, this gives rise to the algorithm for viterbi and it is the version that appears in James Alan, natural language understanding second edition, Benjamin Cummins publisher 1995. So, what is given to the algorithm is HMM, which means we are given the start state S 1 the alphabet a 1, a 2, a p which is the observation symbols, the set of states s 1, s 2, up to s n, the transition probabilities for various state pair and output symbol combination. And this is nothing but P S i going to S j on symbol a k, which is equal to P S j comma a k given S i.

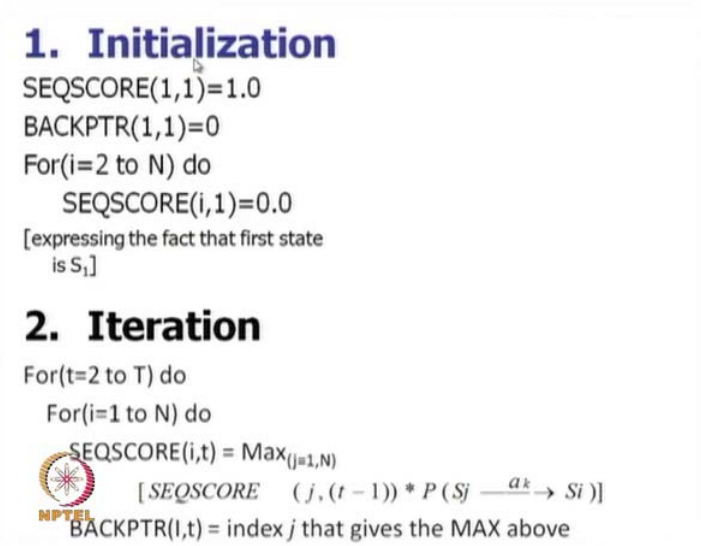
So, the probability of all these combinations are given to us form the transition and other observation probability tables and the output string we have been told is that a 1, a 2 up to a t you have to compute the probability of the best state sequence, which corresponds to this observation sequence. To find the most likely sequence of states C 1, C 2, C t which produces the given output sequence that is C 1, C 2, C t is equal to arg max over c of P C given a 1, a 2 up to a t with mu as the model.

(Refer Slide Time: 24:53)



For this algorithm, we have a data structure which is a N by T array called the sequence score to maintain the winner sequence always, N is the number of states. So, N rows, T is the length of the output sequence, so the number of columns would be equal to T plus 1, another N into T array called back pointer to recover the path. So, the three distinct steps in the viterbi implementation are initialization, iteration and sequence identification.

(Refer Slide Time: 25:27)



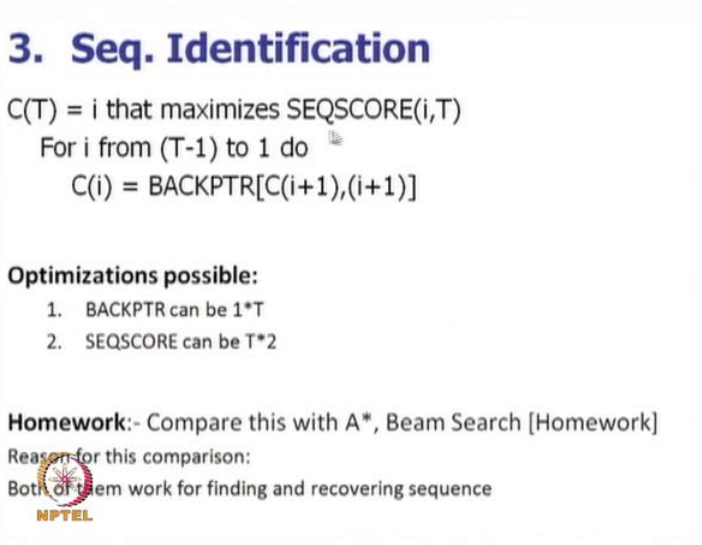
When we initialize, we make sequence score 1 1 equal to 1.0 that is to show that S 1 is the starting sequence, back PTR 1.1 is 0 to show that this is the first state, no state

perceive this and for 2 to N do sequence score i 1 equal to 0.0. So, we make the probability value 0 in all states rather than S 1.

Now, come iteration step for t equal to 2 to capital T do for i equal to 1 to N do. So, this means we go of over the observation sequence, symbol by symbol T equal to 2 to T, the capital T is the length of the observation sequence. And for every symbol on the observation sequence, we do and iteration over the number of states. So, this is to record the state in which the sequence is ending. So, sequence store i comma t is max of j equal to 1 plus N, sequence score j comma t minus 1 into probability of S j to S i for the symbol a k.

So, this essentially is multiplying the accumulated sequence probability which is sequence score j comma t minus 1, by the transition probability from S j to S i on symbol a k. So, this particular max make sure that we advance only k states at a particular level. So, this the main most important part of the viterbi algorithm, we do not advance any state whose probability value is less than the winner sequences probability value, ending in the particular state. Back pointer I comma t is equal to index a that gives the maximum the above, it is a way of keeping the pointer to be able to recover the states sequence.

(Refer Slide Time: 27:53)

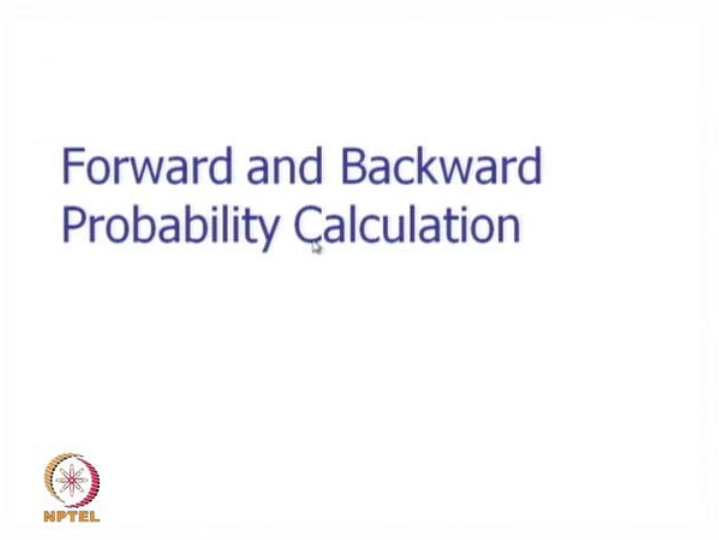


Sequence identification finally, C T equal to i that maximizes sequence score i comma T. for i from T minus 1 to 1 do C i equal to back PTR C i plus 1 comma i plus 1 and this produces the, state sequence which has been found to the highest probability state

sequence. Now, the back pointer can be an single dimensional array and the sequence score can be of order T square.

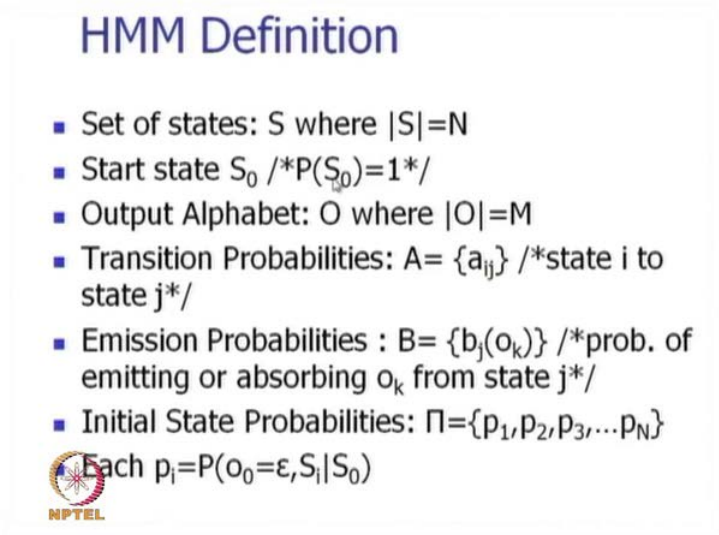
One can compare this viterbi research, with A star and beam search and the reason for this comparison is that, both this algorithms are finding and recovering sequence. So, it will be the interesting idea to compare the nuances of the algorithms.

(Refer Slide Time: 28:46)



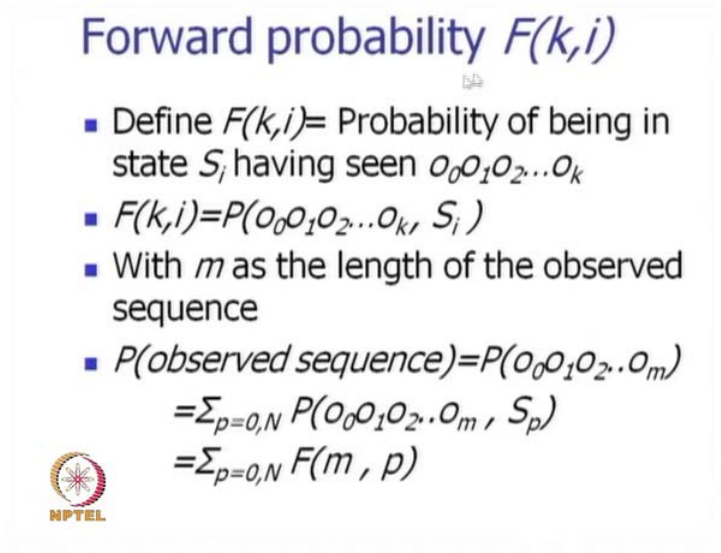
Now, we come to an important topic called the forward and backward probability calculation, we will make use of some mathematical expressions for this case.

(Refer Slide Time: 28:57)



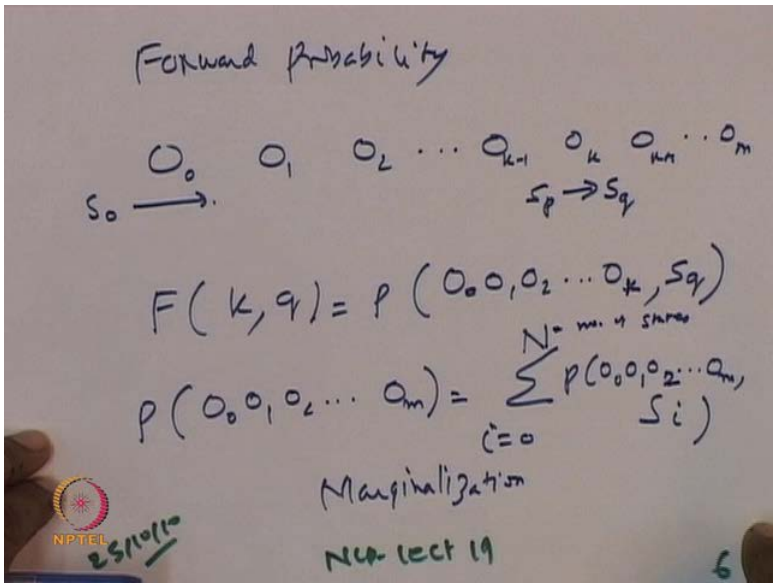
We go back to the definition of hidden Markov model because this will by require for forward and backward probability calculation. The state of states is S where, the number of state is equal to N the start state is S 0 and the probability of S 0 is equal to 1, output alphabet is O, where the number of output symbol is M. The transition probabilities is a matrix A with values small a i j state i to state j transition probability is given through the number a i j, emission probabilities B is b j o k probability of emitting or absorbing o k from the state j. Initial state probabilities pi is equal to p 1, p 2, p 3, p n and each p i is equal to probability of output symbol being epsilon and the state being S i given the state S 0. So, it is the essentially capturing the probability of the system being in a particular state initially.

(Refer Slide Time: 30:18)



We now define the forward probability F k comma i, the forward probability F k comma i is probability of being in a state S i having seen O 1, O 2, O 2 up to O k. So, this is better expressed by working through the mathematical calculations. So, we are working out the forward probability.

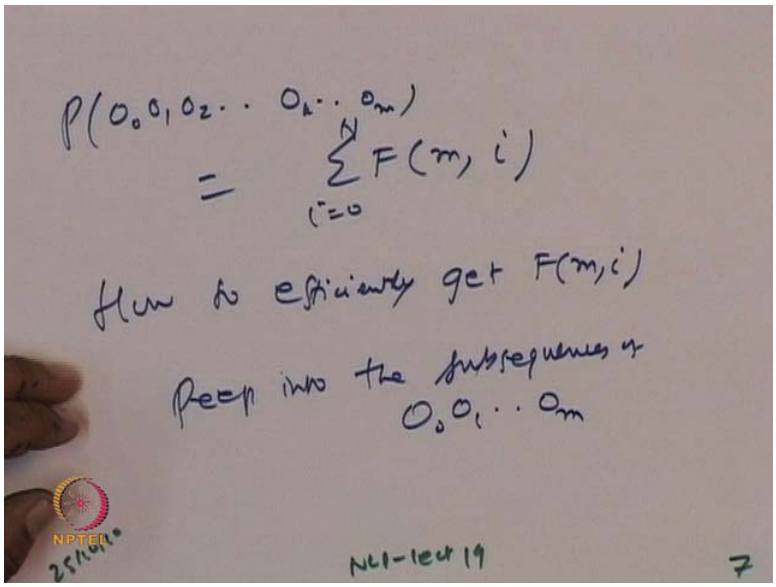
(Refer Slide Time: 30:46)



So, the diagram is useful here we have the symbols O 0, O 1, O 2, O k minus 1, O k, o k plus 1, O m, the system starts from S 0 and goes to some state here. Let us assume, the probability of the state is s or the state is S q where the symbol O k is same and before that the state was S p, S p to S q 1 symbol O k. So, the forward probability F k comma q is the probability of seen O naught, O 1, O 2 up to O k and being in state S q.

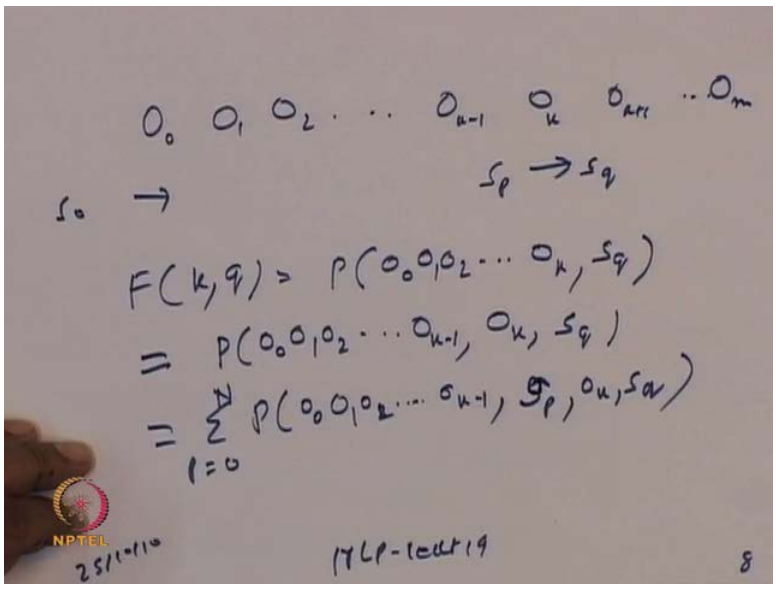
So, this is joint probability which expresses the fact that we have seen, the sequence O naught, O 1, O 2 up to O k and we are in the state S q. So, it is quite easy now to express probability of the whole all four sequence O naught, O 1, O 2 up to O m this is nothing but probability of O naught, O 1, O 2 up to O m comma S i, i equal to 0 to N. So, this is nothing but marginalization. So, this whole sequences probability O naught, O 1, O 2 up to O m is appended with S state S i and we have to take all possible values of S i, i go goes from 0 to N where N is the number of states.

(Refer Slide Time: 33:01)



So, this means that the final output probability or the probability of the sequence p O naught, O 1, O 2 up to O m is nothing but sigma F m i, i equal to 0 to N. So, by using forward probability we can may goes up, we can find out the probability of output sequence O 0 to O m. Now, how to efficiently get F m i this requires us to peep into the subsequences of O naught, O 1 to O m.

(Refer Slide Time: 33:57)

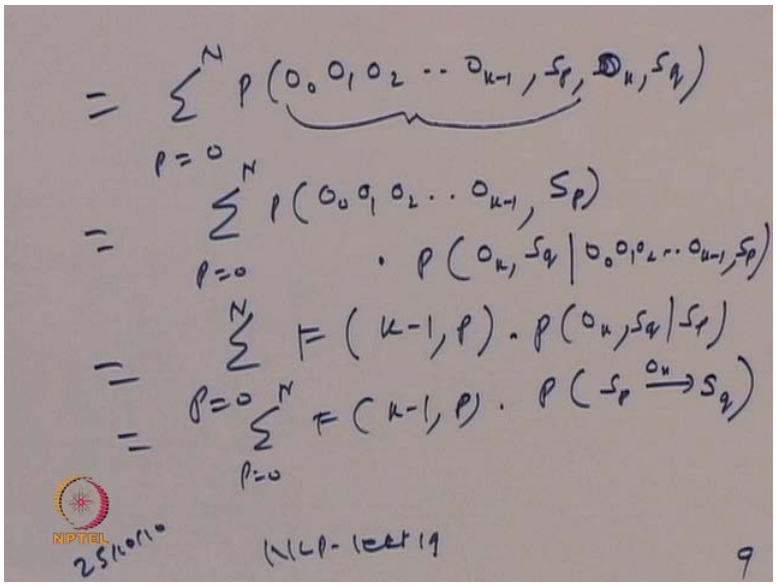


So, if you invoked this diagram once again O naught, O 1, O 2, O k minus 1, O k, O k plus 1, O m. And we have the state as S p and state at here S q, the state here is 0 and

goes to n state S 1, S 2 state etcetera. So, now, F k q is nothing but the probability of O 0, O 1, O 2 up to O k and S q, it is probability O 0, O 1, O 2, up to O k and S q. So, this I can write as, this is the sequence on which I am operating and I am trying to see, if I can make use of the subsequence's property to compute this probability.

So, this would be equal to P O naught, O 1, O 2 up to O k minus 1, O k comma S q. So, I am just dividing this sequence O 0, O 1, O 2 up to O k in a two parts, O 0 to O k minus 1 and O k. Now, I do marginalization and I introduce another variable O 0, O 1, O 2 up to O k minus 1 comma S p, O k, S q where p is equal to 0 to N. So, this I will have to do I am just introducing marginalization, here is a new state which is introduced this is S p. S p is nothing but the state previous to S q and having seen the sequence O naught, O 1, O 2 up to O k minus 1, this system is in state S p after that S p is O k and goes to state S q alright. So, this is the expression.

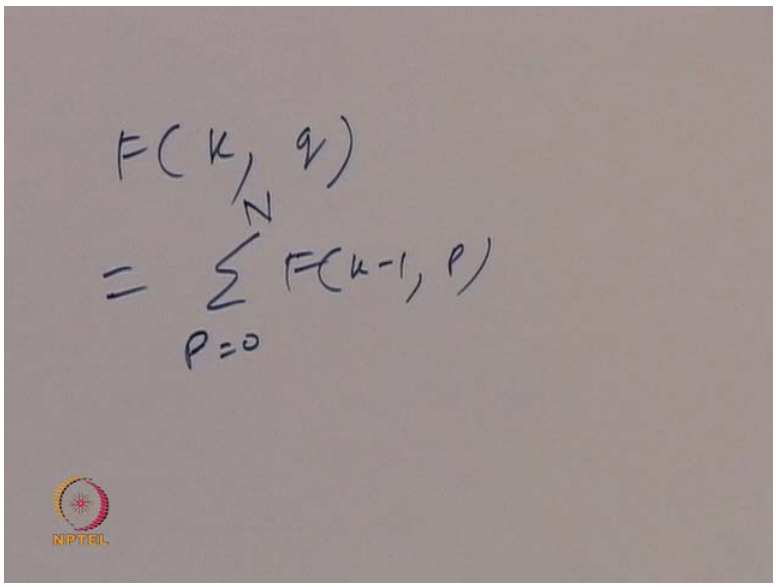
(Refer Slide Time: 36:22)



Now, what I can do is that I can take up this particular term O naught, O 1, O 2 up to O k minus 1 comma S p comma O k comma S q p goes from 0 to N this I write as sigma p equal to 0 to N probability of O naught, O 1, O 2 up to O k minus 1 and the state S p into probability of O k comma S q given o naught, O 1, O 2 up to O k minus 1 and S p. So, this is how chain rule operates I have taken out this particular part and I take this part as the conditioning, entity in the next probability, which is a multiplied.

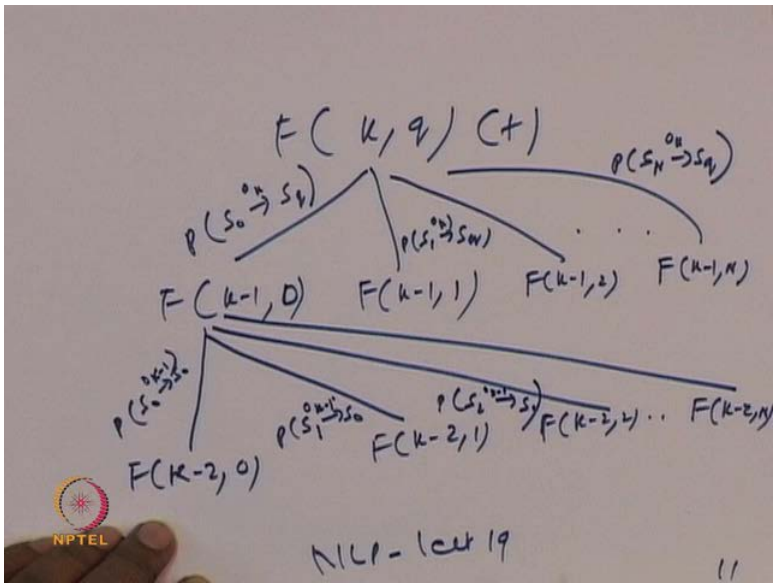
Now, p equal to 0 to N this probability is familiar to us, is nothing but F k minus 1 p and this probability, we can neglect O naught to o k minus 1 because we said that the next state and the next symbol, depends only on the current state. So, next state and the next symbol depends only on the current state. So, this equal to p O k comma S q given S p and the term comes out to be equal to p equal to 0 to N F k minus 1 p into probability of S p to S q or O k alight. So, this is the expression that we have got and it is a recursive expression for the forward probability, giving us a recursive algorithm to compute the forward probability.

(Refer Slide Time: 38:32)



So, what we have got is F k comma q is nothing but sigma F k minus 1 comma p, p is equal to 0 to N the recursive expression. Now, from this we can draw a diagram and see how this quantity is computed.

(Refer Slide Time: 38:59)

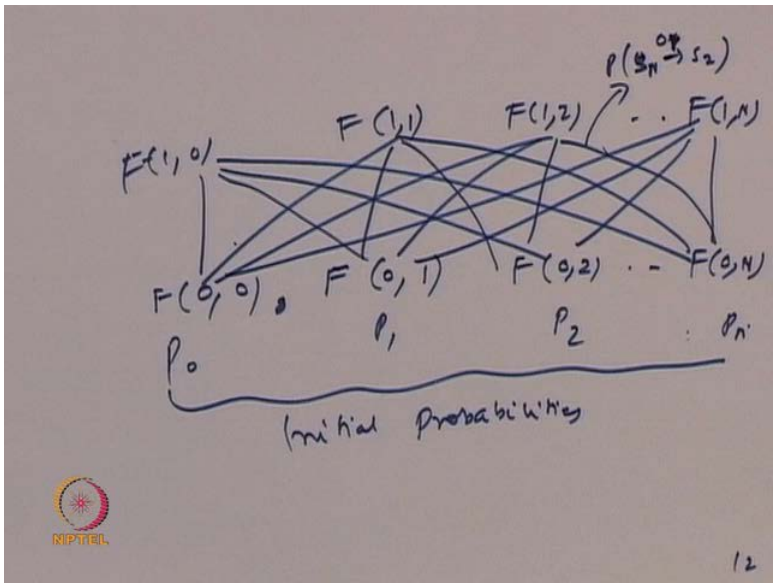


So, if k comma q is computed by F k minus 1 comma 1, F k minus comma 0, F k minus 1 comma 1, F k minus 1 comma 2 and so on F k minus 1 comma N and on this, we have the probability S 0 to S q on O k probability of S 1 to S q on O k and probability of S n to S q on O k. So, how this will be computed is as follow, we get this F values multiplied them, by the transition quantity here and sum up all these products to we have finally, obtain F k comma q.

So, F k comma q is obtain from F k minus 1, 0 F k minus 1, 1 F k minus 1, 2 up to F k minus 1, N. So, there are these N computation of F and each is multiplied by this transition and finally, everything is summed up, so we can see that at every stage we have N computations. Now, F k minus 1 0 can again be computed as follows, F k minus 2, 0 F k minus 2, 1 F k minus 2, 2 up to F k minus 2, N what are the probabilities on these arts.

The probability here would be p S 0 to S 0 on k minus 1 observation, this will be p S 1 to S 0 with order k minus 1, with observation k minus 1 this will be p S 2 to S 0 with symbol k minus 1 and so on.

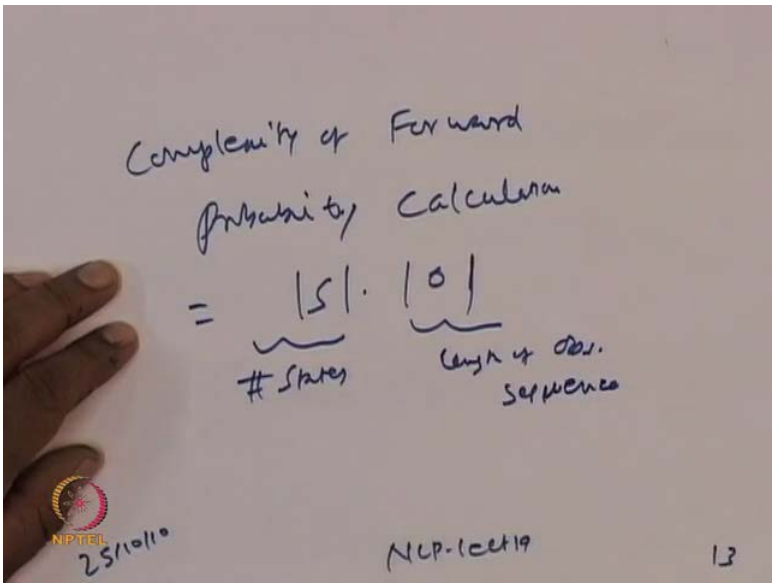
(Refer Slide Time: 41:21)



So, as we develop this tree we will finally, have a situation where we were computing F 0, 0 F 0, 1 F 0, 2 up to F 0 N and these are used to compute F 1 0 then F 1 1, F 1 2 up to F 1 N, alright. And these quantities, which are the boundary conditions really these are nothing but p 0, p 1, p 2, p n and these are nothing but initial probabilities. So, the boundary condition is the initial probability, these are the terminations points for the recursion.

So, from F 0, 0 F 0, 1 F 0, 2 F 0, N which in turn at the initial probabilities, we can compute F 1, 0 F 1, 1 F 1, 2 up to F 1, N. We can see the probability on any art here, the probability here for example, it would be this p S N 2 to S 2 on symbol O 1, so this is the probability. So, this probability is known these probabilities values are also known, we multiplied them and at each note we perform a some of these lower probability values or probability values, so we have these probabilities values. So, this is quite easy to see that forward probability can be computed in linear time.

(Refer Slide Time: 43:32)



So, the complexity of forward probability calculation is nothing but number of states into length of observation sequence. So, number of states into length of observation sequence, we can similarly compute the backward probability which we will see in the next class.