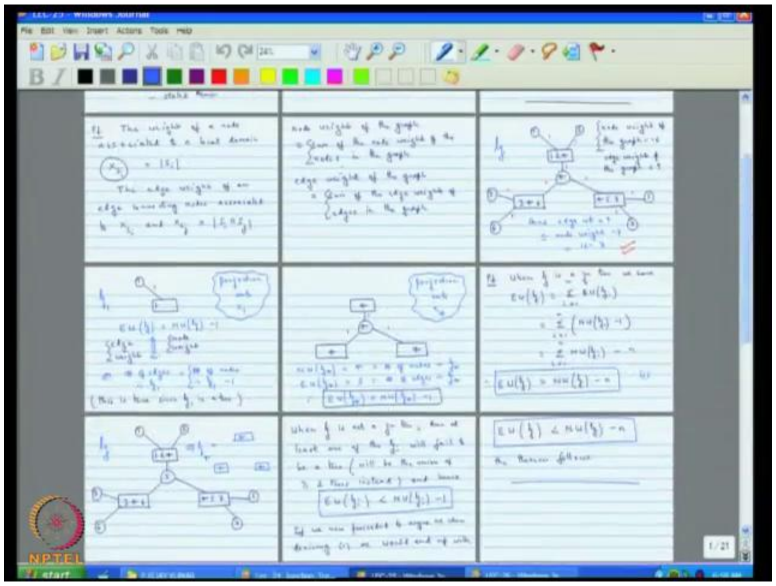
Error Correcting Codes
Prof. Dr. P Vijay Kumar
Department of Electrical Communication Engineering
Indian Institute of Science, Bangalore

Lecture No. # 26
Message Passing on the Jn Tree

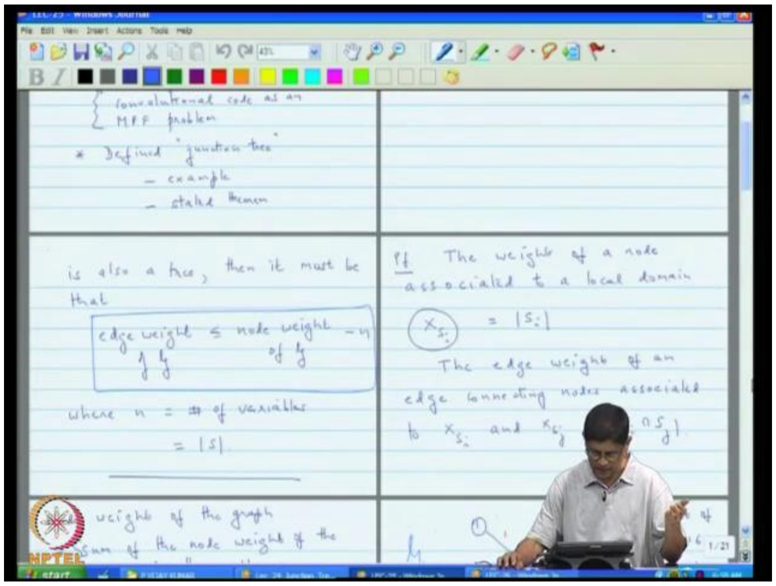
Welcome back, this is our twenty sixth lectures in the series, and as always let me begin with a recap, I am going to show you what we did last time, a quick glance through it.

(Refer Slide Time: 00:26)



In the last lecture we looked at, excuse me the title was examples of junction tree construction, and first what I actually did was I stated a theorem in which I actually stated that if the junction, if the local domains can be organized into a junction tree, then a junction tree will have the properties of a maximal weight spanning tree, so that means that in order to try to construct junction tree, it is logical to start, but trying to construct the maximal spanning tree, if that maximal spanning tree is a junction tree, then you are in then you have an, if that maximal weight maximal spanning tree satisfies the certain inequality as it turns out, then you actually get a junction tree.

(Refer Slide Time: 01:28)



{ convolutional code as an
{ MPF problem

* Defined "junction tree"
  - example
  - stated theorem

is also a tree, then it must be that

$$\boxed{\underset{\text{of } \mathcal{J}}{\text{edge weight}} \le \underset{\text{of } \mathcal{J}}{\text{node weight}} - n}$$

where n = # of variables
$= |S|$

weight of the graph
sum of the node weight of the

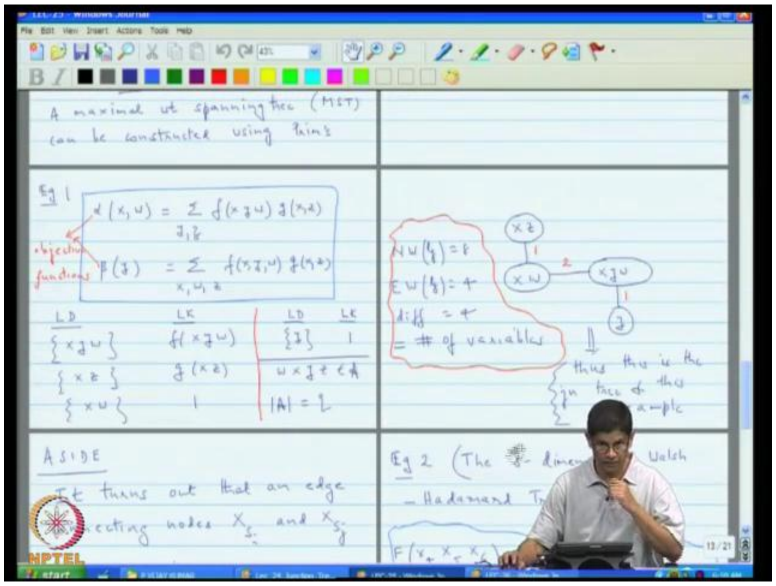
Pf The weight of a node associated to a local domain

$\boxed{x_{S_i}} = |S_i|$

The edge weight of an edge connecting nodes associated to x_{S_i} and x_{S_j} [illegible] $\cap S_j|$

We went through that last time that particular quality is that the edge weight of the graph that you construct, must have the property, then it is less than or equal to the node weight of the graph minus the number of variables. That was the first thing, and then we left at an example (( )) how we are added this construction? I will show to an example of this and then so we prove that then went out to me examples.

(Refer Slide Time: 02:00)



A maximal wt spanning tree (MST) can be constructed using Prim's

Eg 1

$$\boxed{\begin{aligned} \alpha(x,w) &= \sum_{y,z} f(x\,y\,w)\, g(x,z) \\ \beta(y) &= \sum_{x,w,z} f(x,y,w)\, g(x,z) \end{aligned}}$$

objective functions

LD	LK
$\{x\,y\,w\}$	$f(x\,y\,w)$
$\{x\,z\}$	$g(x\,z)$
$\{x\,w\}$	1

LD	LK
$\{y\}$	1

$w\,x\,y\,z \in A$

$|A| = 2$

NW($\mathcal{J}$) = 8
EW($\mathcal{J}$) = 4
diff = 4
= # of variables

$$\begin{tikzcd} & xz \arrow[d, dash, "1"] & \\ & xw \arrow[r, dash, "2"] & xyw \arrow[d, dash, "1"] \\ & & y \end{tikzcd}$$

⇓

thus this is the jn tree of this example

ASIDE

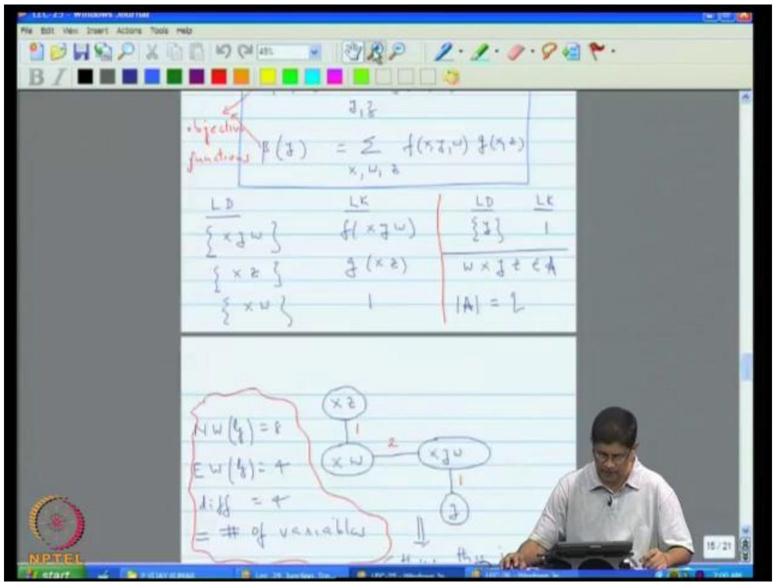
It turns out that an edge connecting nodes x_{S_i} and x_{S_j}

Eg 2 (The [illegible] dimen[illegible] Walsh – Hadamard Tr[illegible]

$F(x_4\, x_5\, x_6$

The First example, we have to do with one of the computations that I introduced early on, they were 2 functions of certain 4 variables alpha and beta that we actually have to compute. I should have one could go about compute a maximum weight spanning tree, and when we check the criterion at the end of ones we finish constructing the maximal weight spanning tree. We found that yes it meets the equality that is requiring to satisfied is satisfied, therefore what we have is a junction tree. And then we looked at a second example involving the eight-dimensional Walsh (( )) transform and there we found the actually you know a junction tree cannot be constructed. The reason being that when you had the maximal weight spanning tree fails to satisfy the equality, and therefore a junction tree simply does not exist at least when constructed using that technique that we have outline.

(Refer Slide Time: 03:01)



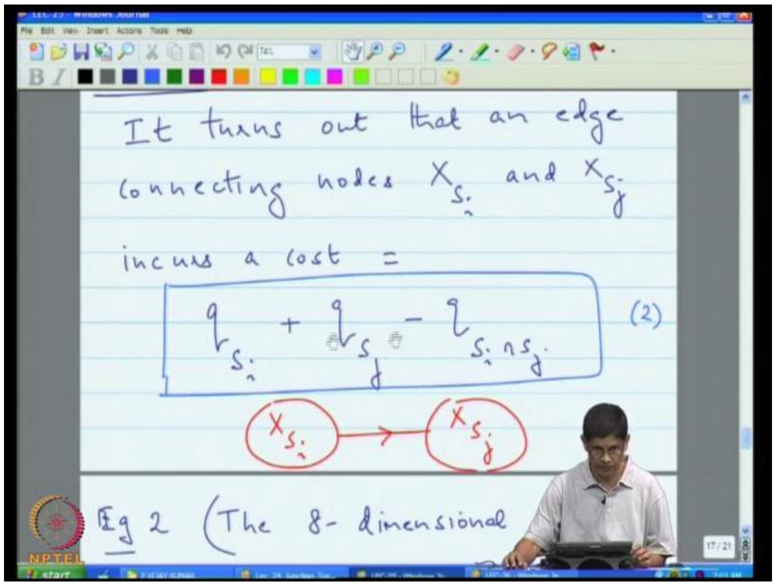
LD	LK
{xyw}	f(xyw)
{xz}	g(xz)
{xw}	1

LD	LK
{y}	1

w x y z ∈ A

|A| = 2

(Refer Slide Time: 03:08)



It turns out that an edge connecting nodes X_{S_i} and X_{S_j} incurs a cost =

$$\boxed{q_{S_i} + q_{S_j} - q_{S_i \cap S_j}} \quad (2)$$

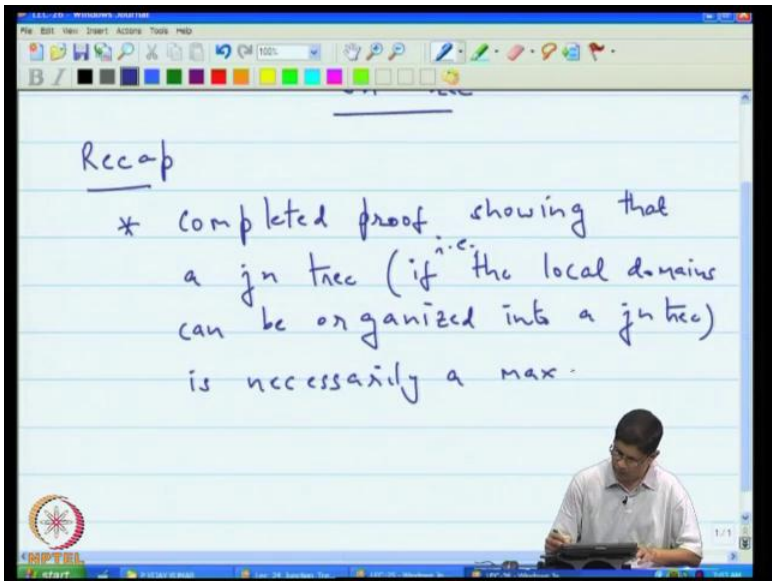
$$\begin{tikzcd} \boxed{X_{S_i}} \arrow[r] & \boxed{X_{S_j}} \end{tikzcd}$$

Eg 2 (The 8- dimensional

And never site common that, I made is that in constructing the all junction tree, that is you are going to be invoked a Greedy algorithm and the particular algorithm is known as Prim's algorithm which tends to which attempts to actually add a node to a existing sub graph, by which has maximal weight which is connected through an edge of maximal weight and sometimes you have more than one choice in our point towards that it turns out and we would not actually prove that in the class. I will just take this without prove it turns out that the edge associated with particular edge.

With the cost the computing computational cost is associated with the certain edge is given by, this it is the size of the alphabet to the node on the left size of the alphabet to the hint of the node associated with the node. Am I right? Minus the size of the alphabet is associated with the with the intersection alphabet which is this incase of ties, you can use this criterion that is minimization of this criterion as a way of breaking the ties. First completed proof showing that a junction tree, if the local domains can be organized in to a junction tree, there is enough there is necessarily I will just put i e is necessarily. I think that is about we were and what I would like to do today is continue that and we look at an example, then the title message passing. We will also see once you constructed the junction tree, what next it tense of the next step is to actually pass messages and we will see how that is carried out.

(Refer Slide Time: 04:46)

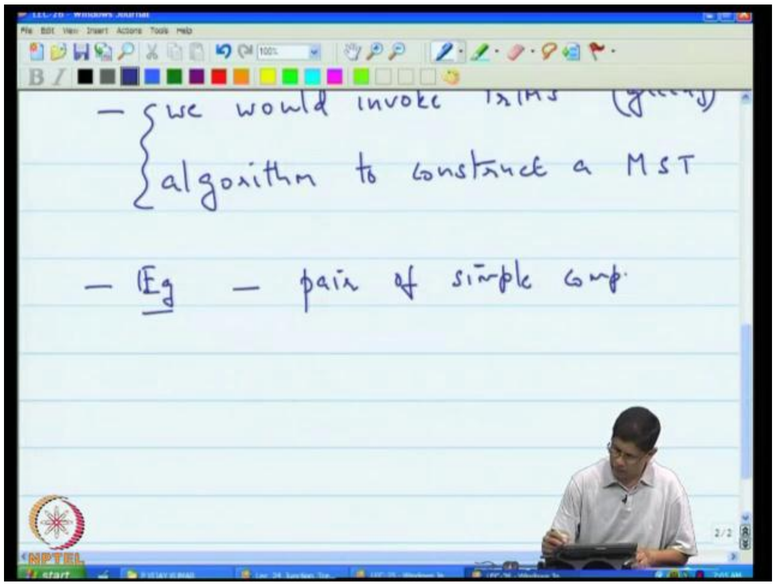


Recap

* completed proof showing that a jn tree (if i.e. the local domains can be organized into a jn tree) is necessarily a max.

Here is recap and at just my set up a little bit here very good. So recap. A maximal weight spanning tree, and then we said that we would invoke Prim's algorithm, I will write Greedy algorithm in brackets to construct a maximal weight spanning tree. So parts, I should abbreviate this. Let us call this M S T, now in the literature M S T often is used in the sense of a minimal spanning tree, but there is not much distinction, because you changes the sign of weights and you can go from maximal weight spanning tree to minimal. There is no wide difference there.

(Refer Slide Time: 06:33)



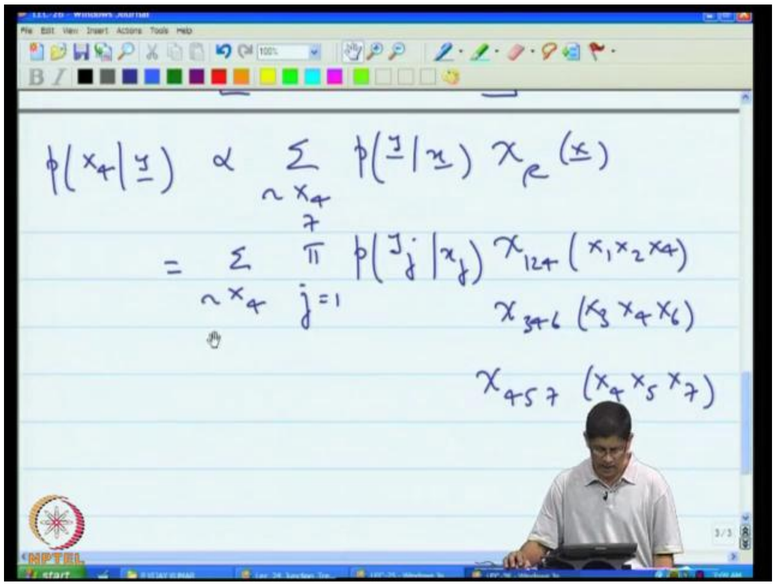
— we would invoke Prim's (greedy) algorithm to construct a MST

— Eg — pair of simple comp.

And then we said that we would invoke Prim's algorithm to construct an M S T as we looked at examples, of simple computation or a pair of simple computation and then we looked at Walsh transform and we saw that for the first example. We were in fact able to construct a junction tree, however in the second the junction tree as according to Prim's algorithm cannot be constructed. Then that brings as to our next example which is that of our 7, 4, 2 code. The third example is our 7, 4, 2 code and this code has the following parity check metrics.

And if for example, let us say you were interested in trying to decode the 4th code symbol in a particular instance.

(Refer Slide Time: 09:31)

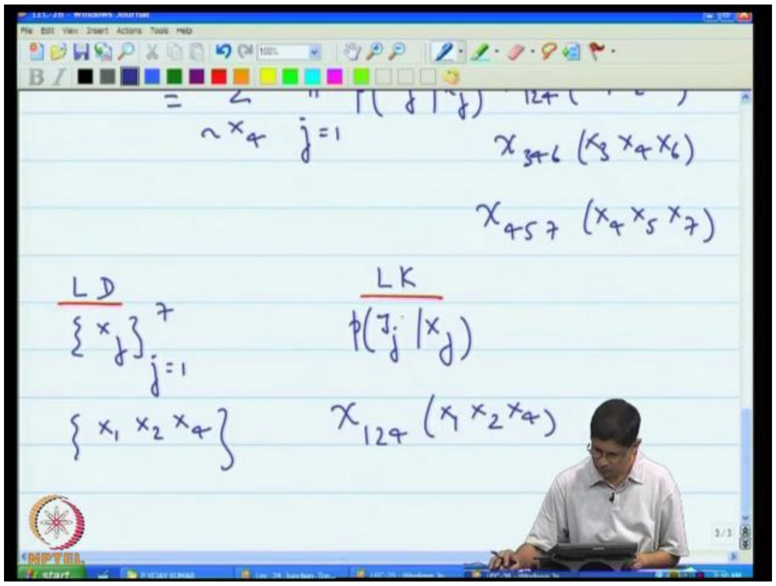


$$p(x_4 \mid \underline{y}) \propto \sum_{\sim x_4} p(\underline{y} \mid \underline{x})\, \chi_{C}(\underline{x})$$

$$= \sum_{\sim x_4} \prod_{j=1}^{7} p(y_j \mid x_j)\, \chi_{124}(x_1 x_2 x_4)\, \chi_{346}(x_3 x_4 x_6)\, \chi_{457}(x_4 x_5 x_7)$$

I have given the entire vector p of x 4 given y, this will shown earlier that this can be computed according to or you can pass on to a computational this type. As before this sign indicates. It is proportional to see they want to complete this and what we actually do here is that you can, I skipped a couple of steps, but basically pass on first of all to the joint distribution then you might then you treat this as marginalization. It is the joint probability of x and y then in fact there into y given x in q of x again you invoke the proportionality constant, again and put the indicator function in the code this function here is 1, if x belongs to the code and 0 otherwise and this is all straight forward.

(Refer Slide Time: 11:55)



$\sim x_4 \quad j=1$

$\chi_{346}(x_3 x_4 x_6)$

$\chi_{457}(x_4 x_5 x_7)$

$\underline{LD}$ $\{x_j\}_{j=1}^{7}$ $\quad$ $\underline{LK}$ $p(y_j|x_j)$

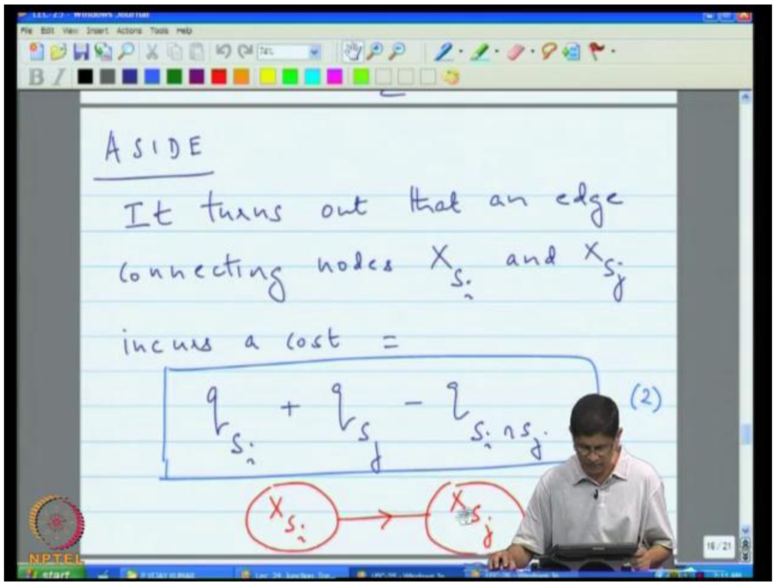
$\{x_1\, x_2\, x_4\}$ $\quad$ $\chi_{124}(x_1 x_2 x_4)$

Because this factor is like this and this factors like this and you have the sum and as I pointed out earlier this wiggle in front of x 4 means, this is what we recall not sum that we sum overall the variable x c other than x 4 and from this that local domains become parent. I will just list them again the local domains are each of each of the x j, j equals to 1 to 7 associated to p of j given x j then we have x 1, x 2, x 4, here these are the local kernels.

We have our, with this we covered local kernels that are there that one local kernel associated to each variable x, a sitting here again you disregard. You do not pay attention to the y sub j because they corresponds to the particular realization here 1 2 4, 3 4 6, 4 5 6 and after them all down then in far as organizing these into our local domain. I think I am going to just perhaps, we should drips, we want to illustrate we can take advantage of this to illustrate Greedy algorithm, we can start with any arbitrary local domain.

Let us say that, I start with the local domain 1 2 4. I want to employ Prim's Greedy algorithm which has try to maximize your edge weight, but then I have the side criterion which has that you try to all minimize this quantity.

(Refer Slide Time: 14:16)



ASIDE

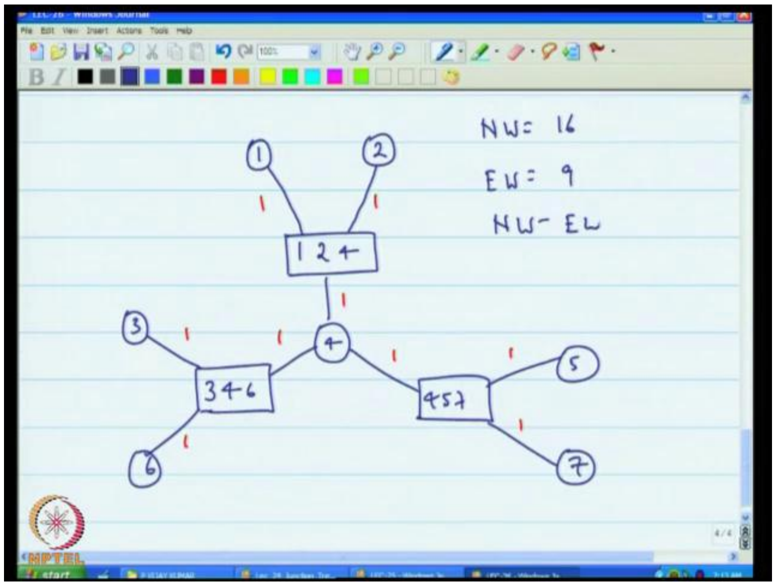
It turns out that an edge connecting nodes X_{S_i} and X_{S_j} incurs a cost =

$$\boxed{q_{S_i} + q_{S_j} - q_{S_i \cap S_j}} \quad (2)$$

$$X_{S_i} \rightarrow X_{S_j}$$

This quantity over here, which is q of s i that is the alphabet to the left less the alphabet to the right minus the alphabet of their intersection, I want to keep eye on that as well, here my choices are these and of course in the interest of maximizing edge weight. I can connect this to either 3 4 5 7 or 1 2 or 4 but from this point of view see the intersection is going to be 1, the intersection is 1, so this thing is for the same size you trying to minimize this.

(Refer Slide Time: 15:00)



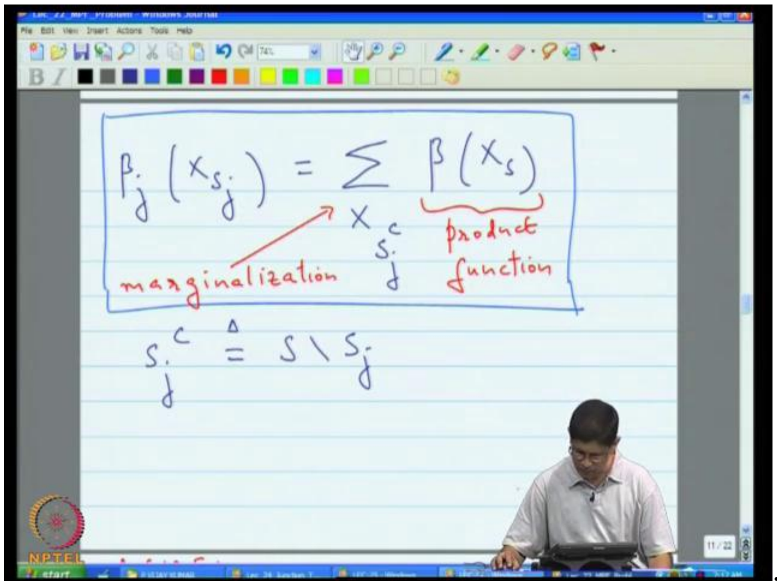
And it becomes clear that you should connect to either or 1 or 2. We connect to 1 over here and then continue in this way you see that you could connect 2 towards 4, and this now at this point is the 4 has intersection 1 with either of these you could introduce either of these in the cost is the same there nothing, must you choose there you put down 4 6 here connect this and her you can tag on 3 and 6 and 4 5 7 5 and 7 which is exactly.

This time we will employ Prim's algorithm, you can check that all here edge weight are 1 because the intersection is always been, so 1 the node weight of the graph is 7 plus 9, which is 16 the edge weight is 9, the node weight minus the edge weight is equal to 7 which is equal to the number of variables and we saw this earlier, this is in fact the junction tree. Now what I am going to do is, I am going to this is this part of here was a repetition from an earlier lecture. I am going, I too just put that down just going to jog your memory anyway see was my get rid of that this is a junction tree.

And I will also not be need in these edge weights, this is our last example of and I just want to draw this graph again on the next space will be cut and pasted. I just said this is our last example our next goal is I just want to move it to the center let see that. Let us see, if I can do that is good what we have d1 just recap very quickly and give a broad over view. We actually stated that the certain problem called the N P F problem where you are given the objectives to marginalize a certain product function and then we said that the first step that will actually do is to actually identify the local domain and the local kernels, and we will try to see if we can organize the local domains in to the structure of a graph which is known as a junction tree.

Now, sometimes as possible me times it is not. Now, let us is possible like in this particular example, then the next step is actually computing the objective function, so in all of these they could be multiple objective function each associated with one of vertices, if you go back to our original formulation of met. I me back on the G D L, let see if I can bring that back, we go this is the setting of the merge the N P F problem and you see here that.

(Refer Slide Time: 18:55)



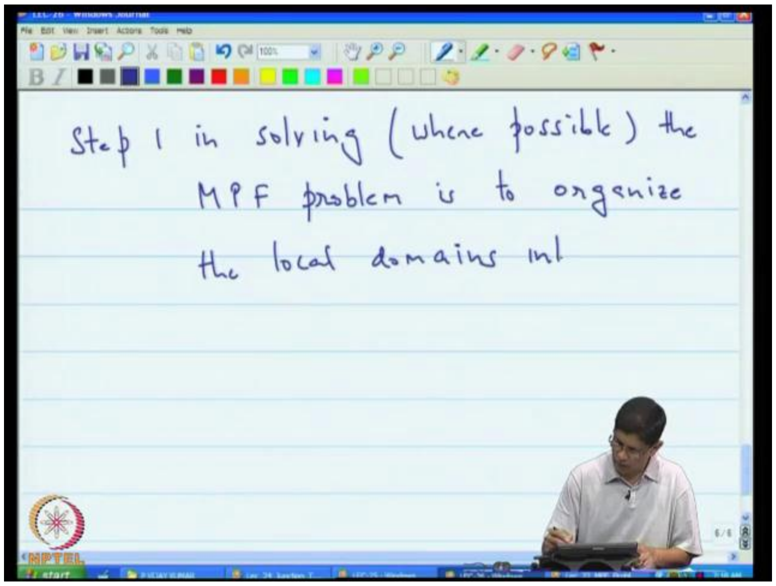
$$\boxed{\beta_j(x_{S_j}) = \sum_{x_{S_j^c}} \underbrace{\beta(x_S)}_{\text{product function}}}$$

marginalization

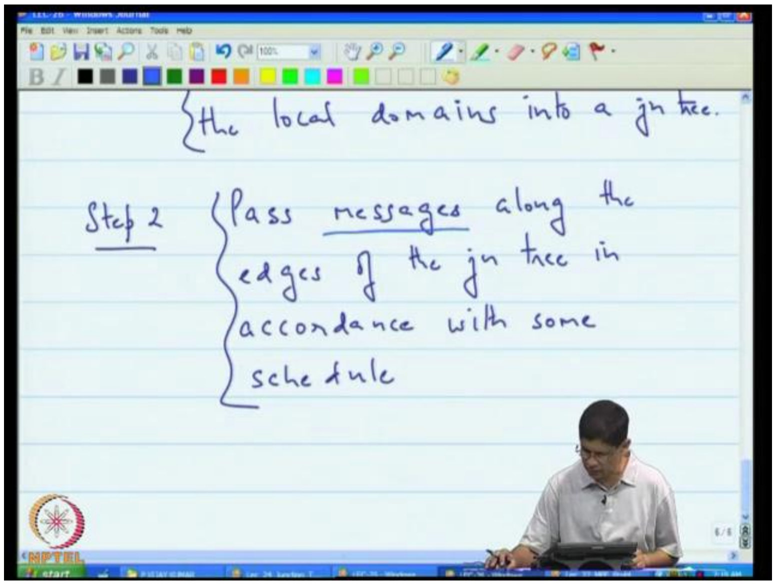
$$S_j^c \triangleq S \setminus S_j$$

Here is an example, where here that you actually have is the global kernel, which is the product of all the local kernels and you are trying to marginalize it, and you could have one objective function associated with one local domain you were as many objective functions as you have local domain. So, this is what you are trying to compute getting back here. It is your next step in trying to compute this objective function the next step is message passing.

(Refer Slide Time: 19:35)



Step 1 in solving (where possible) the MPF problem is to organize the local domains int

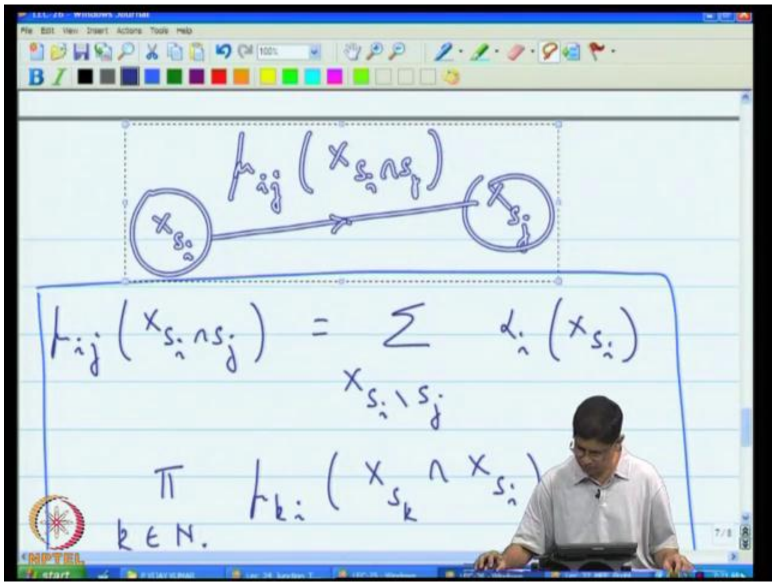


{the local domains into a jn tree.

Step 2 {Pass messages along the edges of the jn tree in accordance with some schedule

So, step one in living and I should prepare possible the M P F problem is to organize the local domains in to a junction tree. Step two pass messages along the edges of the junction tree in accordance with some schedule. We will come to this issue, have schedule a little later, but let us focus on the word message here the question is well, what is the message that you have planning to transmit between the nodes.

(Refer Slide Time: 21:35)



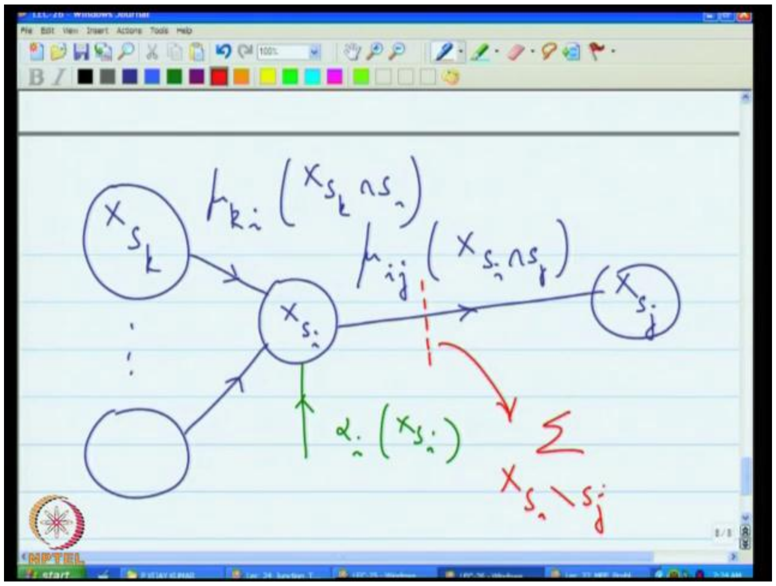
$$\begin{tikzcd} x_{S_i} \arrow[r, "\mu_{ij}(x_{S_i \cap S_j})"] & x_{S_j} \end{tikzcd}$$

$$\mu_{ij}(x_{S_i \cap S_j}) = \sum_{x_{S_i \setminus S_j}} \alpha_i(x_{S_i}) \prod_{k \in N_i} \mu_{ki}(x_{S_k} \cap x_{S_i})$$

In general, let say that you have a node x s i, and you have a second node x s j then the message between them that you will pass will be first of all will designate that as mu i j x of s i intersect s j, that means that the message pass between the two will be only a function of all the variables, that are common to both of these local domains that exactly is the expression.

The exact expression is that mu i j of x s i intersect s j, is equal to the sum over x of s i minus s j of alpha i x s i times the product of j in, I am switching is that meat that a k here time the product k in n i k not equal to j of mu k i of x s k intersect x s i, this then is a message that you will pass what is that mean showed. So, let me expand this figure a little bit on the next page.

(Refer Slide Time: 23:56)


x_{s_k}
$\mu_{ki}(x_{s_k \cap s_i})$
$\mu_{ij}(x_{s_i \cap s_j})$
x_{s_i}
x_{s_j}
$\alpha_i(x_{s_i})$
$\sum_{x_{s_i \setminus s_j}}$

What you have in the figure is that, there are other nodes that feed in to this node you may have a generic x s k and you may had other nodes and each of them in turn is going to pass on a message to this node and that message will be mu k i of x of s k intersect s i, all that were sink and then there is another message that is all coming into this node which is the particular local kernel of this node. From a graphical point of view, it is very simple were we are saying that the message that is passed along this edge is simply the product of all the incoming messages to this node times.

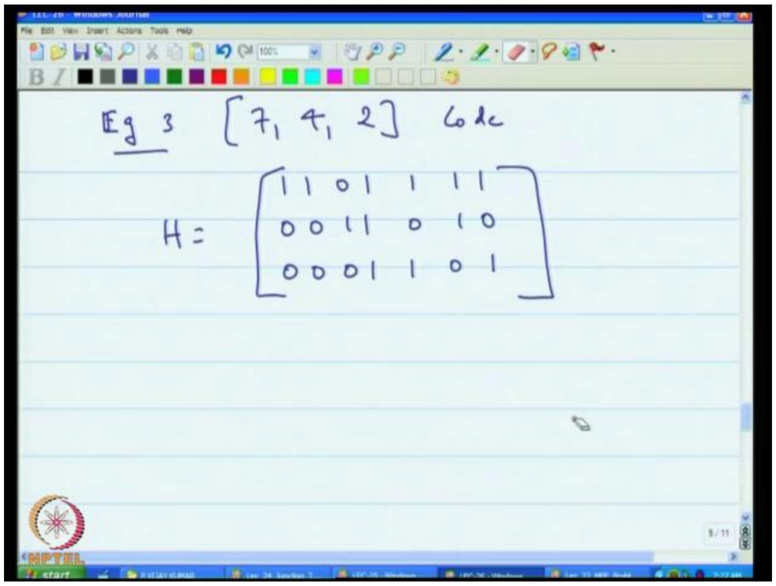
The value of the local kernel and one thing that is to keep in mind that, there is no memory in the system, in the sense that the message never depend upon what happen what messages were passed in the present. It could happen in a network that you will pass be continues be passing messages between nodes, and it is initially you might simply be passing the wrong message.

Simply, because the incoming message user not correct, but eventually what will happen is that there is a spreading of knowledge throughout the network eventually each node is going to acquire whatever knowledge meats to acquire, and then it will have the right message to pass on here and that will then continue to propagate on the network. Message in an actual is simply the product of all the incoming messages multiplied by the local kernel. And there is an additional q step and that q step is a step in which involves marginalization with respect to what is respect to what well you see that in this junction tree and this side we have variables corresponding s j and here s i there are some common variables, but there are me missing variables there is there are me variables in s i which are missing in x s j what you can do can actually marginalize get rid of there through marginalization.

What that means was that over here. You will carry out a summation over x s i minus s j and that is exactly what this expression is actually telling you this n i here n i is nothing but the neighboring nodes of node i is the set of neighbors of the node x s i this is of course in the in this graph in the junction tree. Let us see if we can give an example for I am going to go back to this the decoding of this code, let us the select the page and copy it over.

I want to give this in certain context perhaps let us to the following, I may insert a new page here in between and give you the setting. Example of message passing, we are going to work with our 7, 4, 2 code which has the parity check metrics that, I introduced earlier perhaps I will put that down here.

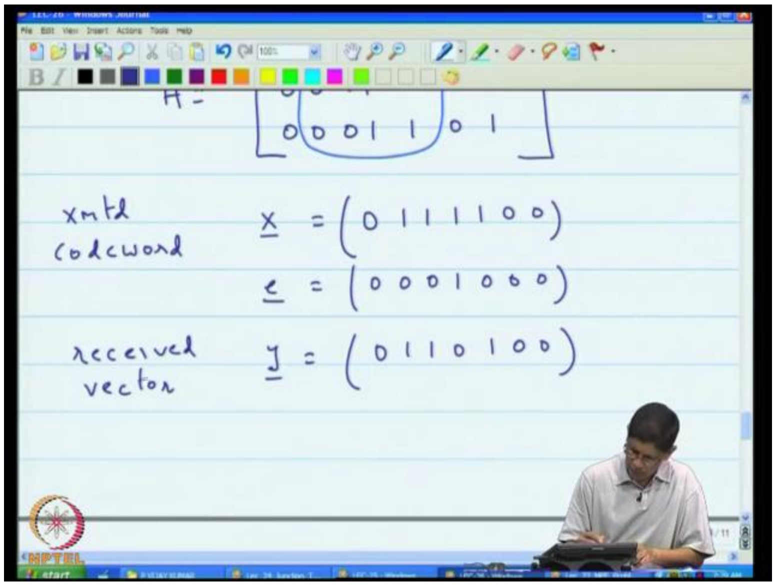
(Refer Slide Time: 29:11)



Eg 3 $[7, 4, 2]$ Code

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

(Refer Slide Time: 29:26)



$H = \begin{bmatrix} \cdots \\ 0\ 0\ 0\ 1\ 1\ 0\ 1 \end{bmatrix}$

xmtd codeword $\underline{x} = (0\ 1\ 1\ 1\ 1\ 0\ 0)$

$\underline{e} = (0\ 0\ 0\ 1\ 0\ 0\ 0)$

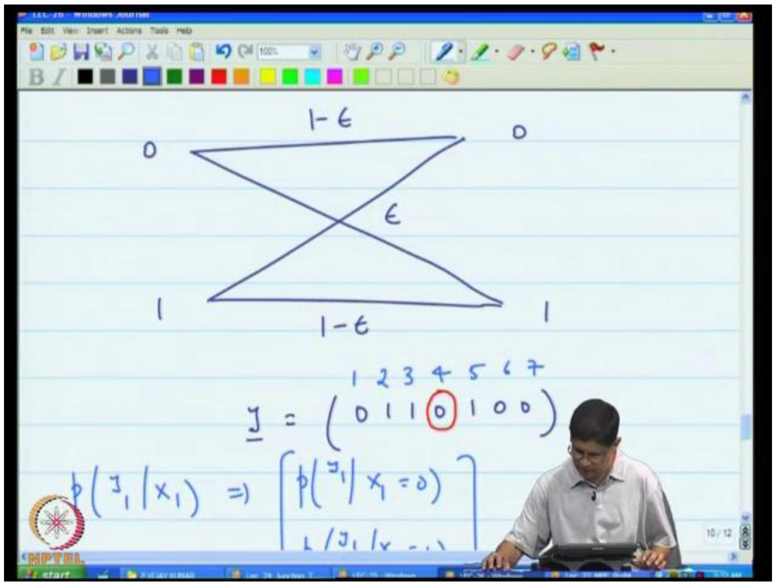
received vector $\underline{y} = (0\ 1\ 1\ 0\ 1\ 0\ 0)$

We want to decode this code and let us consider the case when say in an example, that the transmitted code word x is given by well I mean to make sure each times x is 0, but you can check that the x true for this example, what I am doing here is I am taking the middle 4 symbols equal to 0 and another that will work. Because if I had, there is me thing a little there was narrow in the metrics, in which sorry about that we should have been 0s and I need to correct the beginnings, let me go back and correct that was the type there. I did catch these should have been

0 is the same code that we heard earlier no change, if i long here in this code. I know that this fact here the code word, because if I had the entries of let us say that these columns 2 3 4 5. if I add these column then I will actually get all 0 column from that I know that this is a code word and let us seen that there is a single error.

Let us say that is there is an error vector which is 0, 1, 1, 1. I want to introduce a single error, let 0, 0, 0, 1, 0, 0, 0. I have deliberately introduced it in the fourth symbol and now my receive vector will then be 0, 1, 1, 0, 1, 0, 0. So this is the received vector. My aim is to actually correct this error here, this error that was actually introduced which means that this particular symbol is actually in error. Now, I will all assume that all I guess any one more page here. I will all assume that the channel as banners metric channel.

(Refer Slide Time: 32:15)



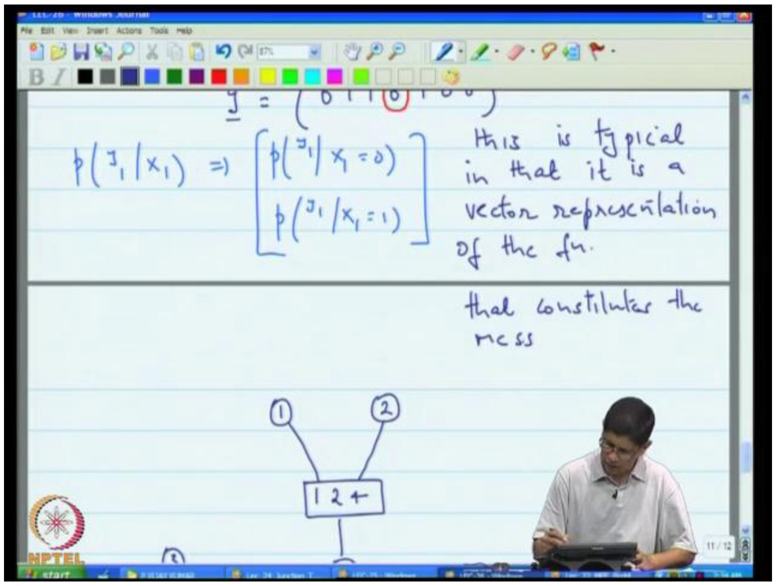
$1-\epsilon$
0
0
ϵ
1
1
$1-\epsilon$
1 2 3 4 5 6 7
$\underline{y} = (0\ 1\ 1\ \textcircled{0}\ 1\ 0\ 0)$
$p(y_1 | x_1) \Rightarrow \begin{bmatrix} p(y_1 | x_1 = 0) \\ p(y_1 | x_1 = 1) \end{bmatrix}$

So, here are the cross over probabilities and what I want to do now is try to pre compute some quantities that will actually be using. Let me copy this over and what I want to do is, I want to compute these symbols, now correspond to 1 through 7.

Let me have put down 1 2 3 4 5 6 and 7 and if I mean say in p of y 1 given x 1 and intrusive this quantity for, p of y 1 given x 1 is equal to 0, and then also p of y 1 given x 1 equal to 1, that brings me to another common that I need to make that is in here the message is as we have given a formula for the messages that are passed and you can see that these massages are functions of certain variable, what does it mean to pass as a message a function of a certain variable the variable can take on different values what does it mean to say that mean to pass on a function well the easiest way to do it and which is the way we will follow is to actually say that either function of a certain set of variables will evaluate this function for all possible values of these variables and will pass on that as a vector will write them as a vector in message passing, you could think of the message is being functions or you could think of them as being vectors which tell out the values of all possible values like the function, in our case in here as it turns out our messages are often functions of the single variables.

Let for example, when you are passing this conditional probability as a message then this is the kind of message that will actually end of passing, we will write this is a vector, we will just make a small command here.

(Refer Slide Time: 35:17)

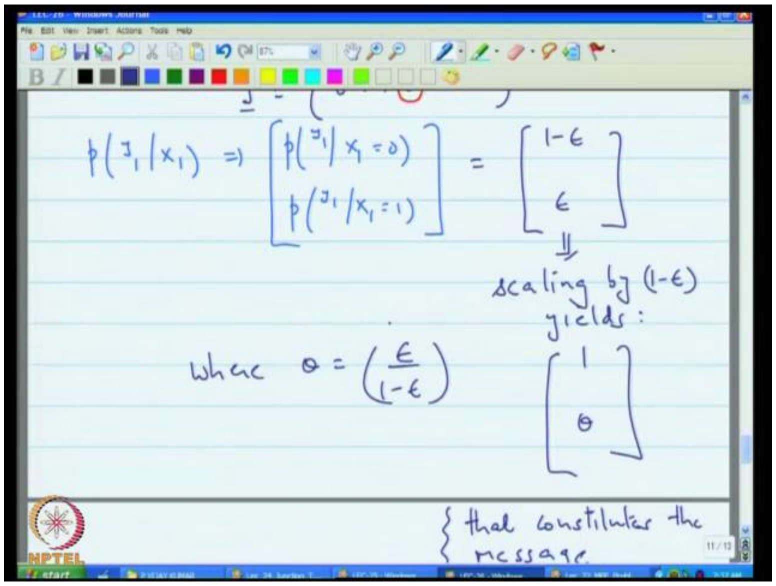


$$p(y_1/x_1) \Rightarrow \begin{bmatrix} p(y_1/x_1=0) \\ p(y_1/x_1=1) \end{bmatrix}$$

This is typical in that it is a vector representation of the fn. that constitutes the mess

This is typical in that it is it is a vector representation of the message of the function that constitutes the message. In this particular instance you can see that since y 1 is 0.

(Refer Slide Time: 36:27)



$$p(y_1/x_1) \Rightarrow \begin{bmatrix} p(y_1/x_1=0) \\ p(y_1/x_1=1) \end{bmatrix} = \begin{bmatrix} 1-\epsilon \\ \epsilon \end{bmatrix}$$

$\Downarrow$

scaling by $(1-\epsilon)$ yields:

$$\begin{bmatrix} 1 \\ \theta \end{bmatrix}$$

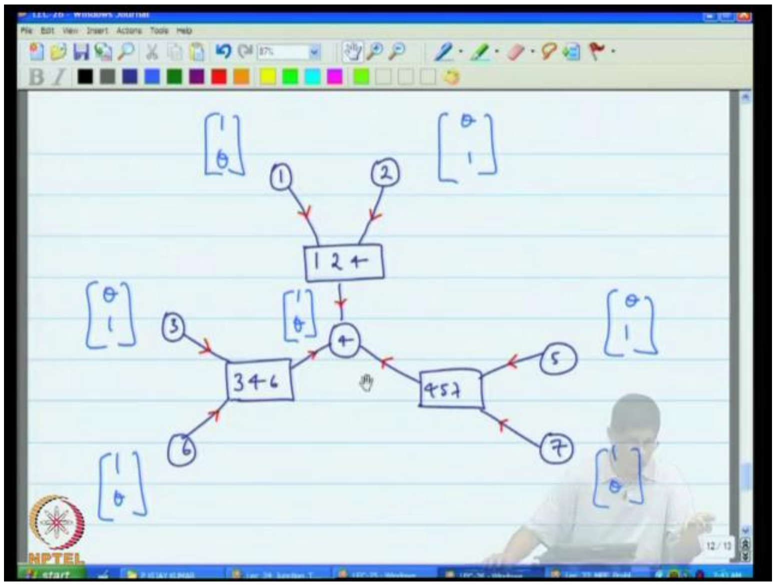
where $\theta = \left(\frac{\epsilon}{1-\epsilon}\right)$

$\}$ that constitutes the message

you can actually evaluate this and this is a binary symmetric channel the probability of receiving a 0, given that you transmit a 0 is 1 minus epsilon and this is accordingly epsilon, this is typical of a message that we were actually pass. However, when we pass messages or it is perfectly, because we are trying to pass these messages to make decisions about the variable and your decision about variable is not going to be altered, if you actually scale all your messages by a universal constant in this case, what we will do for the sack of convenience will actually take that universal constant to be 1 minus epsilon and with that we can here and scaling by 1 minus epsilon yields the following vector 1 and theta where theta is epsilon upon 1 minus epsilon, this is purely for the purpose of simplifying our competition.

Introduction, I am not ready to identify the messages that are passed here. So, just as this is the local kernel associated with that node has variable as x 1 right, I think in identifying the local kernels it helps to actually have in front of as that they received vector. Let us copy that over again, and I will put it here keeping this in mind where ever we have a 0, the particular local kernel value will be 1 theta where ever it is a 1 what is going to happen is that there precisions of the epsilons and 1 minus epsilon gets interchange.

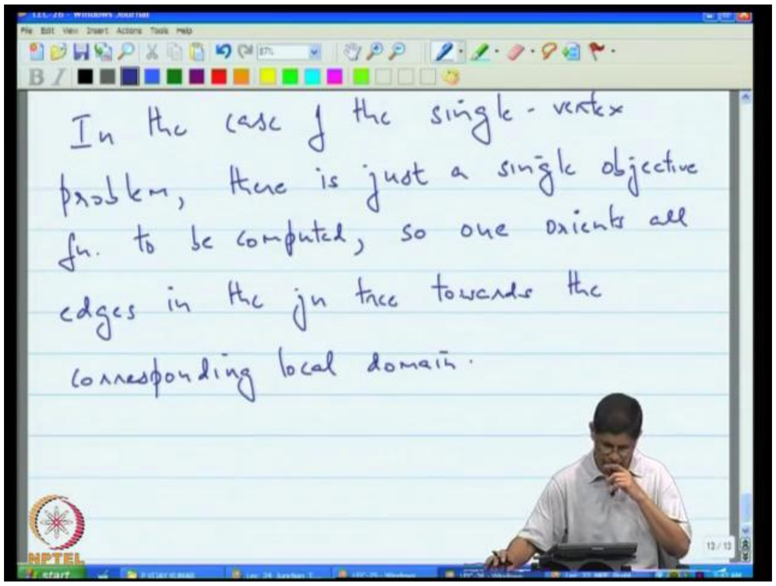
(Refer Slide Time: 38:57)



We will have a theta 1 where ever there is a 0 let us put down a 1 theta these then are your local kernel, you have 1 theta again you have 1 theta here at 4 and then again at 6 and 7 it is 1 theta 1 theta and for the other nodes, we know that the local kernels are theta and 1. Now, we do not need this anymore, because we have got this forget out I will remove that and will write this then is theta 1 theta 1 and theta 1 these, then are your local kernels and message passing now there is another component of message passing which has to do with the schedule is of you go back here. I said that step two is pass messages along the edge of the junction tree in accordance with me schedule.

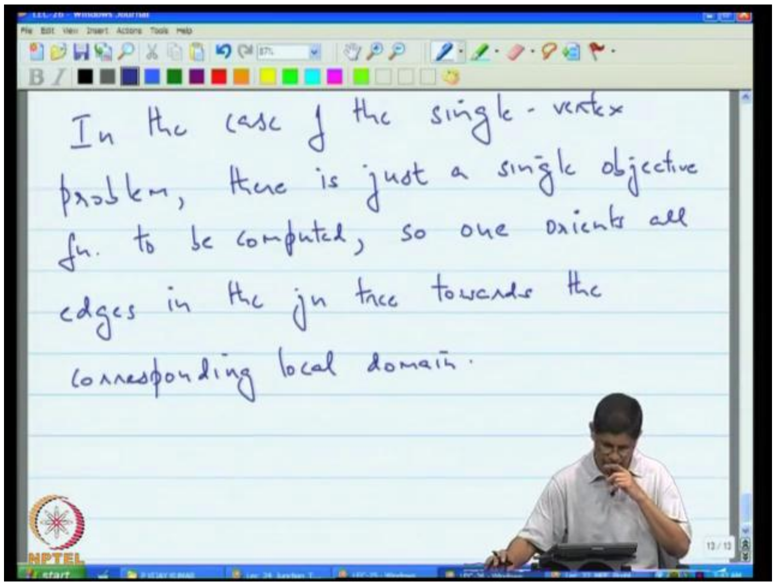
So here, now the schedule particular schedule that you depends on how many objective functions you want to actually compute in this case we are interested in only one objective function, in that case what do you do is your next step is to actually orient all the edges in the graph towards the local domain, whose objective function you wish to compute see, let me put arrows on everyone of these edges. Now, you oriented all the edges towards the central node x 4 and you want to pass messages in the direction of the arrows. Now, this is the case this will always be the case when you have that is called single vertex problem.

(Refer Slide Time: 41:15)


In the case of the single - vertex problem, there is just a single objective fn. to be computed, so one orients all edges in the jn tree towards the corresponding local domain.

In the case of the single vertex problem, there is just a single objective function to be computed, one orients all edges in the junction tree towards the corresponding local domain that is exactly what we did and now we pass messages.

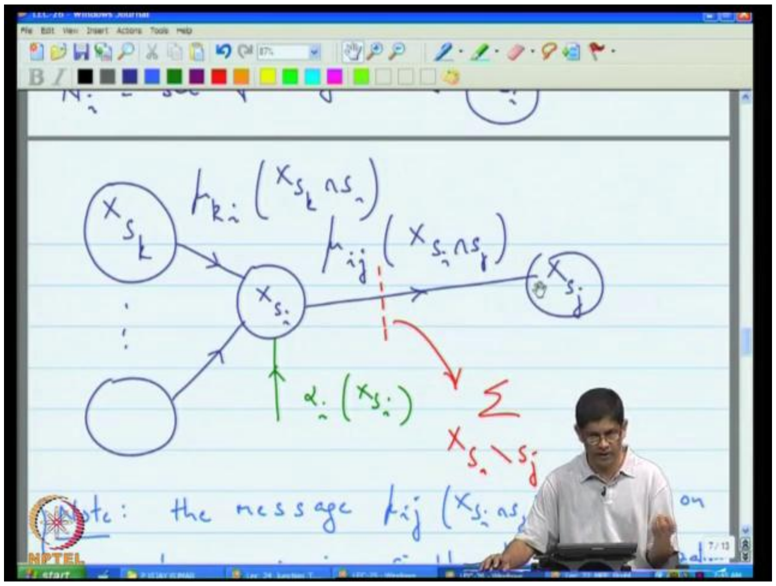
(Refer Slide Time: 42:35)


In the case of the single - vertex problem, there is just a single objective fn. to be computed, so one orients all edges in the jn tree towards the corresponding local domain.

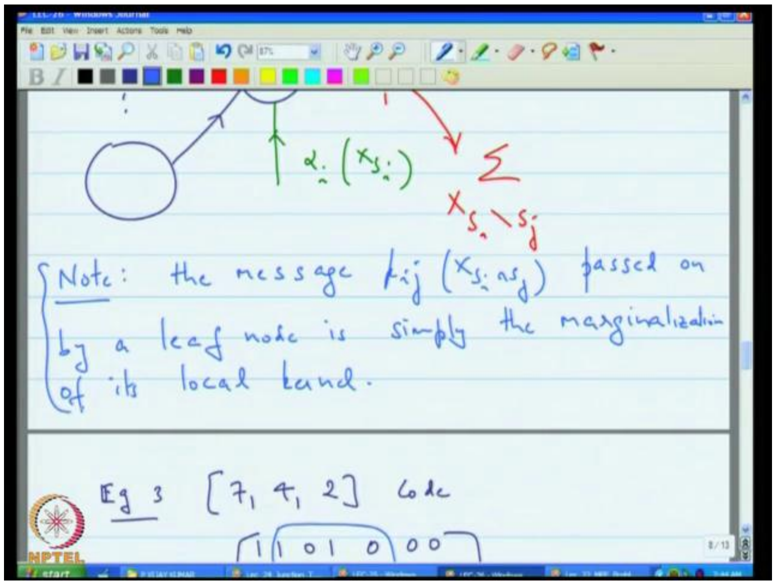
Now, remember what I told you earlier that there is no memory, but it is most effective to pass messages when a node see it is when you must it is most effective if no passes messages only one set has heard from all his neighbors.

I will illustrate for example here this node initially, what will happen that the local kernels like this node have not initially received any message from either these two nodes. It is not quite ready to pass on a message to down four another hand, there is no harm in doing because node 4 can simply weight and tell edge series is correct message, but as on the other hand wasting a competition, if you did not want to do that would d could do you could actually start with those nodes that do have correct messages, through port for example these called leaf nodes. In this tree have their local kernels report, I should add to these that this message schedule is for interior nodes what happens is that the edges of the network the message that is passed is simply the local kernel.

(Refer Slide Time: 44:14)



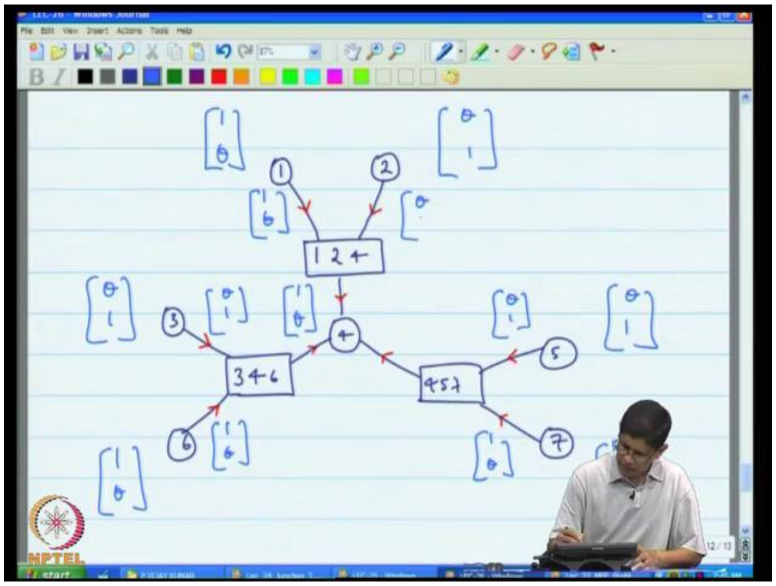
$\mu_{ki}(x_{s_k \cap s_i})$
$\mu_{ij}(x_{s_i \cap s_j})$
x_{s_k}
x_{s_i}
x_{s_j}
$\alpha_i(x_{s_i})$
$\sum_{x_{s_i \setminus s_j}}$
Note: the message $\mu_{ij}(x_{s_i \cap s_j}$


Note: the message $\mu_{ij}(x_{S_i \cap S_j})$ passed on by a leaf node is simply the marginalization of its local kernel.

Eg 3 [7, 4, 2] Code

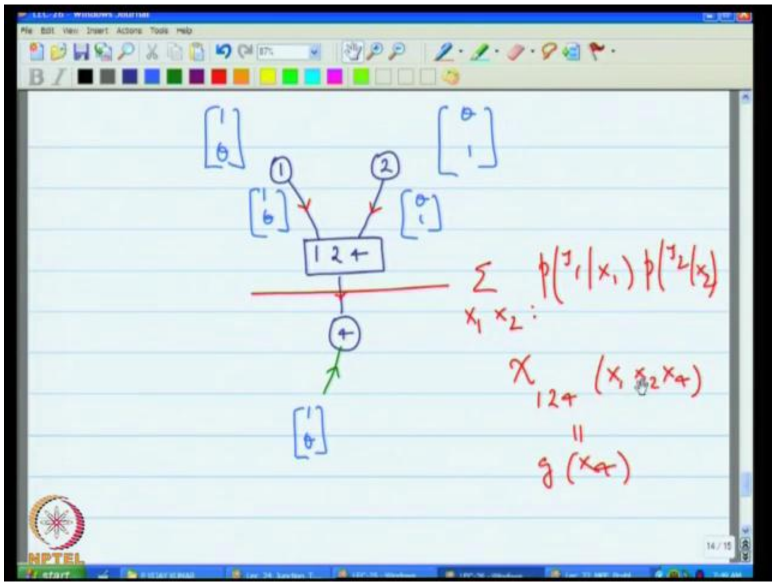
I should perhaps had that me where let me introduce here a note the message mu i j x of s i intersect s j passed on by a leaf node is simply the marginalization of it is local kernel. So, here in this expression here, you can see that because both you have realize simply saying as that low, if this node where a leaf node, then they would be no income edges or nodes to feed in to it. They would be no incoming messages what it would do is simply take it would multiply all the incoming messages together with the local kernel. In general, but since there are no incoming messages you simply take the local kernel and then you marginalize it as necessary and pass it all that is all we are seeing here, in other words it is fit is in to the general frame work, but it is good to point it out separately, and that what will happen in our network initially.

(Refer Slide Time: 46:22)



The message that will be passed by each and every one of these nodes will actually be there local kernels, this will be theta 1 and 1 theta and theta 1 and 1 theta and at similarly, it gives nodes can all send in 1 theta and theta 1 and once they have done that now at that point these parity check nodes are ready to pass the messages, because they have received there incoming messages parts what we will do next will copy this page and focus on the action at particular parity check node. After that little bit careful make sure that I insert the page at correct spot I think I wanted to insert it at the edge. Let us see if I can actually do that I think had modeless existence here, I want you to focus on just the computation that takes place at a particular parity check node for that I want to remove of all of this which is scatterings the view here.

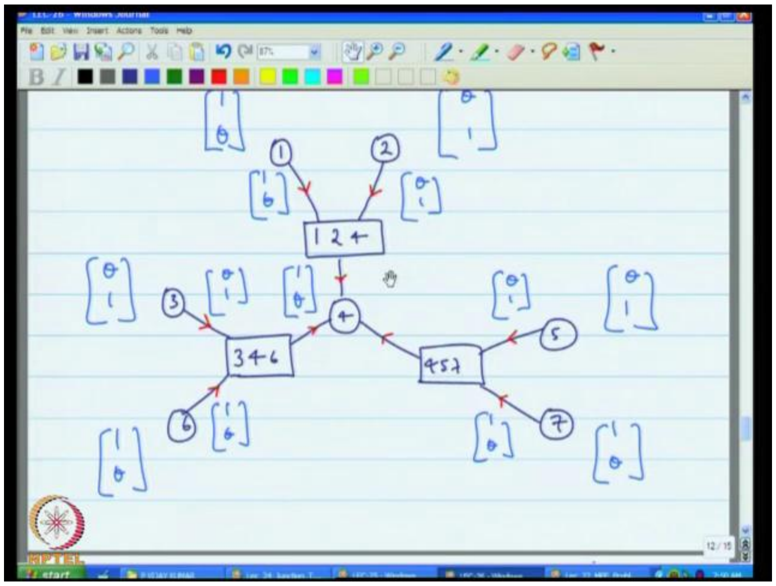
(Refer Slide Time: 47:45)


1 2 4
$\sum_{x_1 x_2 :} p(y_1|x_1)\, p(y_2|x_2)$
$\chi_{124}(x_1 x_2 x_4)$
$= g(x_4)$

Let us get remove these this part of the network, we have this we have saying that look and also I want know of this little bit let us move this is the local kernel, but let us move it here let us see and the message that is passed here really calls for a marginalization. The marginalization is over x 1 and x 2 what you want to do is sum over x 1 x 2 as p of y 1 x 1 p of y 2 x 2 times phi 1 2 4 keep 1 2 4 for x 1 x 2 x 4, and this is the function of x 4 that you wish to actually passed this whole thing here what is that mean? It means that let see you are interested in a x 1, x 4 equal to 0, you have in x 4 equal to 0 the value of this that means here x 1 plus x 2 plus x 4 which must be equal to 0 first is x 1 plus x 2 equal to 0 which means, if you choose x 1 equal to 1 we will choose x 2 equal to 0 and vice versa or if you choose x 4 equal to 1 then when x 1 is 1 x 2 is 0 or vice versa. That in mind this expression here simply means that based upon what we have here that the message that passed here, which is a marginalization of with respect to x 1, x 2 is simply on the 1 hand it is for 0. I would take 1 plus theta squared.

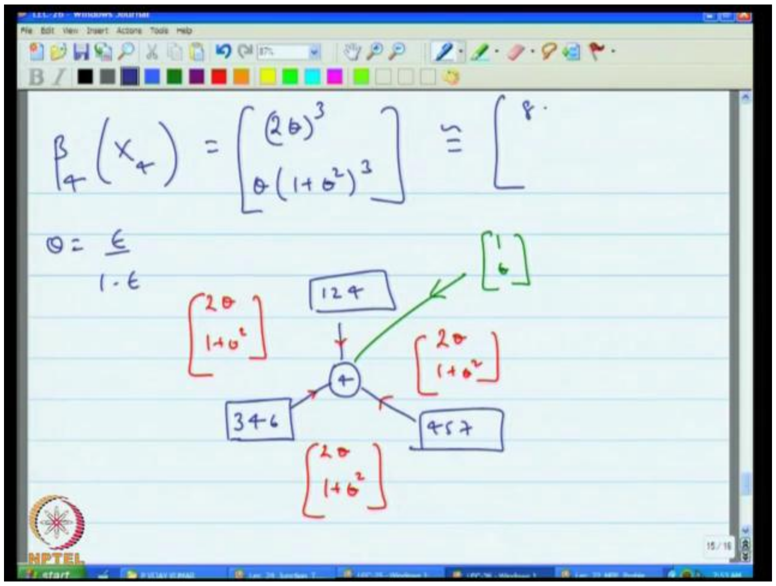
I would take 2 theta, that is I would take 1 and theta and 1 and theta or on the other hand, I would take 1 plus theta square, this entire computation here keeping in mind that functions are represented as vectors will result in this computation over here.

(Refer Slide Time: 50:37)



In fact by symmetry that is all the message that is past along each of these other edges as well and this diagram getting a little bit cruder, but I think it is worth it to let us list the extra of it through focus on 1, last computation.

(Refer Slide Time: 51:43)



$$\beta_4(x_4) = \begin{bmatrix}(2\theta)^3 \\ \theta(1+\theta^2)^3\end{bmatrix} \simeq \Big[\ 8\cdot$$

$$\theta = \frac{\epsilon}{1-\epsilon}$$

Now I want to focus on the computation here, let the computation that is, so we know that each of the incoming messages by symmetry, from the computation. We did earlier is 2 theta 1 plus theta square, this is also 2 theta 1 plus theta square, and this is also 2 theta 1 plus theta square. And then there is the entire local kernel, we should not forget that which in this case was let us go back and see the local kernel was 1 theta. What happens is that? At the end, what you want to do is, when you want to compute the objective function is just simply to take the product of all the incoming messages along with the local kernel. So, then your objective function that you compute is and we have just two minutes left.

So, I will write this and summarize in the next class, so this turns out to be what you are doing is you are just multiplying all of this functions, so that turns out to be 2 theta the whole cube times theta into theta plus 1 squared the whole cube which and remember that theta is equal epsilon 1 minus epsilon. So, this thing seems epsilon is small this quantities are small quantity, we can approximate this by just taking the leading term, so which is eight theta cube here and theta we will neglect higher pass of theta in the denominator.

What this is saying is that look, what I am going to give at the end of this computation is the likelihood function of is the likelihood function of that we set out to computerize in the beginning and it was here of x 4 given y. And what you shown at the end is this result here, which says that probability that existed like theta cube the probability that is says one is like theta, we think it is actually a 1, but I will just quickly review out in the next class basically we have load at further example of message passing decode right, so we will stop here. Thank you.