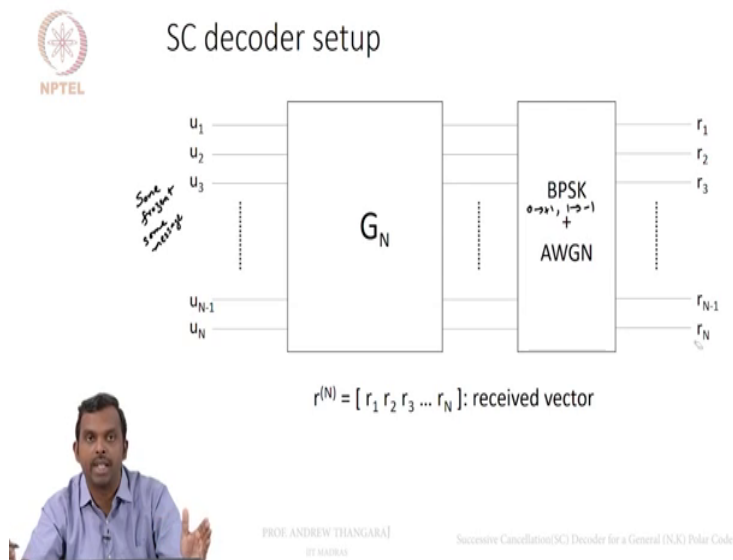


**LDPC & Polar codes in 5G Standard**  
**Prof. Andrew Thangaraj**  
**Department of Electrical Engineering**  
**Indian Institute of Technology Madras**  
**Successive Cancellation (SC) Decoder for a General (N,K) Polar Code**

Hello and welcome to this lecture, in this lecture will talk about the successive cancellation decoder for polar codes, in fact so far we have seen for  $N$  equals 2 and  $N$  equals 4 okay, two small cases where we saw how the message passing happens on the binary tree and what I will do in this lecture is generalised that to an arbitrary block length and journal binary tree and show you how the message passing happens, what are the steps etc and after that we will do coding in MATLAB for that successive cancellation decoder okay, so you will see the steps in the decoder, a quite simple and things also work out quite well surprisingly.

(Refer Slide Time: 1:00)



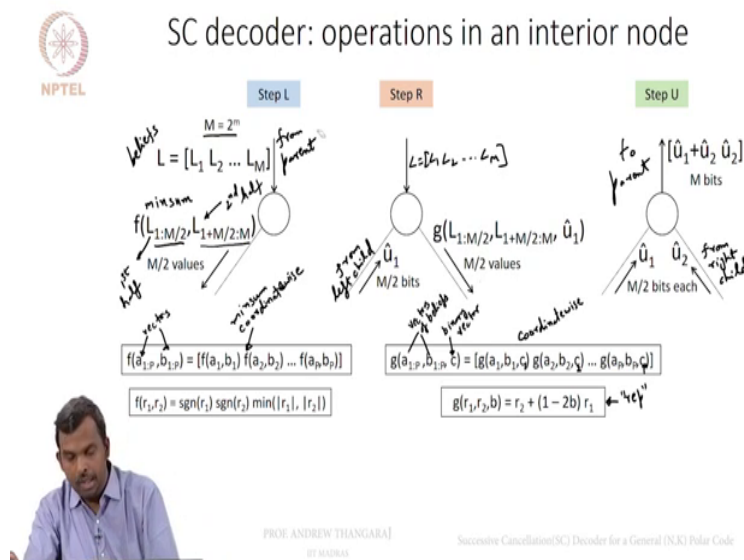
So let us get started, the setup is shown in the slide, we have the bits  $U_1$  to  $U_N$  coming into the polarisation kernel  $G_N$ , now remember some of these  $U_i$ s are frozen positions, some are message okay, so some frozen and some message okay, so that is depends on the reliability sequence and the rate want to transmit etc. and once you send it through the polarisation kernel you get the symbols that you want to try, the bits that you have want to transmit, bits of the code word, you do BPSK, which is 0 to plus 1, 1 to minus 1 okay.

And then you do I mean noise gets added on the channel and then you receive  $R_1$  through  $R_N$  okay, so this is the received vector and the goal of course is to go from the received vectors to the use okay, once you find the  $U$  you are done okay, we saw all this and how do you set it up

in general is the question okay, so we saw that polar success of cancellation decoding involves passing messages on the binary tree and every node in the binary tree is supposed to do some operations, we saw before that every node receives some beliefs from the top from its parent okay and then it first sends something to its left child and then receives, after that receives decisions from the left child, it sends a some beliefs to right child and once it receives bits, decision bits from the right child it combines the two and sends them up back to the parent okay.

So these are the three different operations the node has to do, so I am going to write down that in a general way now okay, so far we saw it only for a specific example, we saw that sometimes the node is to deal with you know multiple incoming messages not just 2, it could be 4 or 8 or something and then how do you write it up in general that I will do in the next slide.

(Refer Slide Time: 2:47)



So that is what shown here, so what are the operations in an interior node, so now what is an interior node, any node which is not a leaf in a binary tree is an interior node, so the interior node will always have this kind of setup, there will be three different steps or operations it has to do, I am told to call the first one is L because it involves sending beliefs to the left child, the second one is R because it involves sending something to the right child and the third one is U which you can think of as up, it involves sending messages back to the and okay.

So let us have some notations, first is this  $M$ ,  $M$  is 2 power small  $M$  okay, the  $M$  can vary now and the interior node will receive a set of beliefs, so these are beliefs okay,  $L_1$  through  $L_M$ ,  $M$  believes it will get okay you can organise that in a vector, you can think of it as a vector and this is what is received by the interior node okay, so some interior node it receives  $M$  incoming beliefs okay.

In step  $L$  which is sending it to the left child what it does is? It does minsum okay, this you know is minsum okay but it does minsum in a particular way, so you see what it does minsum on? It does actually vectors, there are two vectors it divides the vector  $L$  into two okay, the first half, this is the first half, first half of the vector, this is the second half of the vector right, you can see the indices here, indices go from one to  $M$  by 2 and then this goes from  $M$  by 2 plus 1 all the way up to  $M$ .

So it divides into two, first half and the second half and it does minsum,  $F$  represents minsum coordinate wise okay, so here is what I have done here, so if you have two vectors here of the same length okay and when you do minsum  $F$  on the two vectors, I simply apply the minsum coordinate wise okay so I applied coordinate wise okay and that is what happens there on the node also okay, so the step  $L$  the node takes its  $M$  incoming messages and splits it into two and applies minsum coordinate wise okay.

So  $L_1$  and  $L_{M/2+1}$ ,  $L_2$  and  $L_{M/2+2}$  like that okay, so that is how it does, so it gets  $M/2$  values,  $M/2$  beliefs that it will send to the child okay and the child will do whatever it has to do, it will again repeats process like this but as far as this node is concerned, it has send, it has finished the first operations step  $L$  which is sending  $M/2$  values, it got  $M$  beliefs in, it processes it using minsum and sends  $M/2$  values to the left child okay, so that is done.

Now the next time this node comes into operation, it will receive something from the left child okay, so this is from left child; I am not shown that node here but there is a node their with sends  $M/2$  bits okay, so when the left child is done finishing all the processing it will send the bits back to its parent and that is the step here, so once that happens the same node gets activated once again okay.

So it has to do the next step which is step  $R$  okay, in step  $R$  what it does is? It does the  $G$  operation okay, so you remember the  $G$  operation,  $G$  was the repetition code sort of operation, once again the  $L$  is split into two parts, the first half and the second half and then

these decisions are also used and together these G operation is done and you compute M by 2 values, how is this done? If you have two vectors here, this is vectors of beliefs and this is a binary vector okay, so this is the decision here okay.

So I think this is something missing here, so this is a binary vector that the decision is being received here and in the first step you do G of A1, B1, C1, here G of A2, B2, C2 and here G of AP, BP and CP okay, so you have basically to coordinate wise, you apply the G operation okay, coordinate wise, we have seen these G operation before, it is either  $R2 \text{ plus } R1$  or  $R2 \text{ minus } R1$  depending on whether or not the bit is zero or one okay, so that is what is done here, so this is the so repetition code, I put it to the codes, it could be repetition or it could be the complement of repetition, so based on that it does this  $R2 \text{ plus } R1$  or  $R2 \text{ minus } R1$  okay.

So this is a simple enough competition, you can see what it has to happen that M values, the M by 2 bits coming in, there are M values here, there are M beliefs here right, the same L is here on the side, we should retain the same L, L1, L2, LM is still here, you use this believes that where incoming in the node and then once you get some decisions from the left child you combine the two using this function G and send it to the right child okay, so that is step R okay, so that is concludes step R okay.

So now the node can be quite now, it does not have anything to do till it receives something back from the right child also okay, so this is what comes back from the right child, the right child does finished all it has to do and it has send back the decisions back to its parent, so this node is getting back something from the right child, so you remember it already had, U1 hat from the left child, it is getting U2 hat from the right child, remember these two U1 and U2 are now M by 2 bits each they are not single bits, this is M by 2 bits in general we are looking at it and what will the node do? It has to do step U once it has received incoming decisions both the left child and right child, once it has received that, it can do the step U in which it does U1 hat plus U2 hat and U2 hat which is M bits and send its back up to the parent okay.

So we saw this also, this is how the operations go, so once again to summarise there are three operations in the interior node, step L, step R, step U okay, the moment the node receives any belief from its parent, it will do step L okay, which is minsum coordinate wise, splitting the beliefs into two okay and then once it receives decisions from the left child it will do the G operation, combining its incoming beliefs again divided into two along with the decisions in the G operation and the repetition code format and then send it back to the right child and it will wait for the right child to get that its decisions, once the right child gets back its decision,

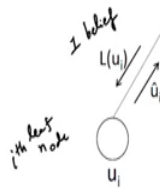
it will do the combination and do the step U in which it will compute what it needs to send back to its parent okay.

So this goes to parent, this came from parent okay, so hopefully this is clear, these are the three different operations an interior node has to do okay, this is how it works okay.

(Refer Slide Time: 9:57)



### SC decoder: decision in a leaf node



If 'i' is a frozen position:  $\hat{u}_i = 0$  ( $L(u_i)$ : ignored)  
 If 'i' is a message position:  $\hat{u}_i = 0$ , if  $L(u_i) \geq 0$ ;  $\hat{u}_i = 1$ , if  $L(u_i) < 0$  threshold



PROF. ANDREW THANGARAJ  
@THANGARAJ

Successive Cancellation (SC) Decoder for a General (N,K) Polar Code


So what about leaf node, now a leaf node does not have any children right, so and it is just one bit okay, so when it receives any belief it will receive only 1 belief and that is a belief for itself right, it receives a belief for itself okay, so now leaf could be in two different situations, the Ith node, this is the Ith node, Ith leaf node, now this Ith position maybe frozen or maybe message okay, so that is the important thing here to take care of, if the Ith position is frozen then your decision  $\hat{u}_i$  will always be zero okay.

So in fact this case L of  $\hat{u}_i$  is sort of ignored okay, so I will say for now later on you will see we figure out where to use it but for now it is ignored okay, it is frozen okay, if it is not frozen of course L of  $\hat{u}_i$  plays an important role, it is the soft information for  $\hat{u}_i$  which is actually a message bit, so if it is greater than zero you decide that  $\hat{u}_i$  is 0, if it is less than zero you decide  $\hat{u}_i$  is 1 okay, so this is the threshold decision okay.

So that is story here, so the leaf node is sort of simple, I gets one belief whenever, it has to keep waiting till it gets its first belief, till it gets its belief, once the belief comes in, it is going to either use it or not use it depending on whether it was frozen or not, if it was frozen is going ignore and always send back  $\hat{u}_i$  is 0, if it is not frozen, it is going to use that belief, use thresholding, make a decision and send back a decision okay, so that is what happens in

the leaf node okay, so we have seen now what happens in the interior node? What happens in the leaf node and now it is just a easy process to put everything together and get the overall decoder okay.

(Refer Slide Time: 11:52)




### SC decoder: sequence of operations

- Start at root

At every node... *(when it is activated)*

- If not leaf, do the following in sequence
  - Do Step L and go to left child
  - When decision is received from left child, do Step R and go to right child
  - When decision is received form right child, do Step U and go to parent
- If leaf, make decision and go to parent



PROF. ANDREW THANGARAJ  
of IIT Madras

Successive Cancellation(SC) Decoder for a General (N,K) Polar Code

So I am going to describe the steps in decoder, the decoder works in a certain sequence, you remember there is a binary tree, there always root, the main top node is called the root, so you start at the root and every node you do the following steps, the steps are always the same okay, if you are not a leaf, if you are node is not a leaf for instance the root is not a leaf okay, so any other node, interior node which has children it is not a leaf okay.

If you are not a leaf and you have to do the following things okay, you first do step L and then you go to the left child, what do I mean by to the left child was the beliefs are given to the left child, the left child becomes activated okay and the left child will do whatever it has to do okay, so at every node when it is activated it has to do the following, activated meaning when it receives beliefs okay, if it does not receive any belief it can be quite, there is nothing much it has to do.

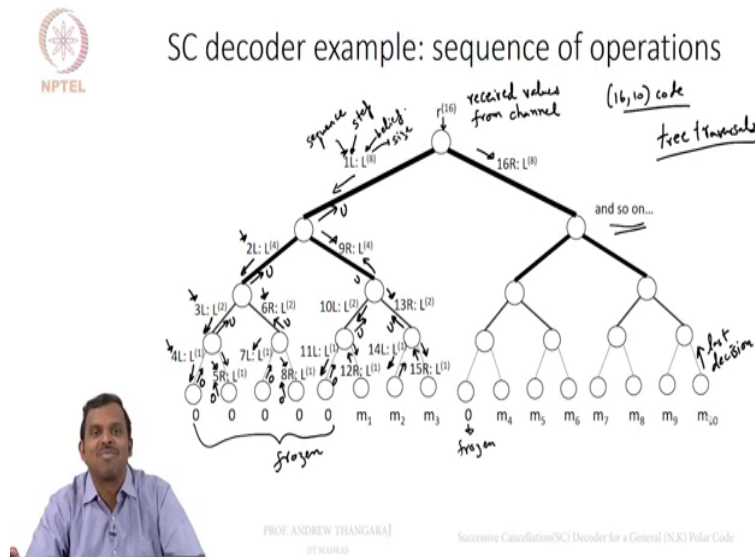
Once it receives beliefs from its parents it has to do the following, it will do step L and it will pass on control to its left child okay and when will it get back control? When will it be activated again? When it receives decision on the left child, when it does it will do step R and it will go to the right child okay and the controller be given to the right child, so after that it can relax, there is nothing much to do after that, it can wait for the right child to get back its

decision, once the decisions come back from the right child, it does the step U and it goes to the parent, it hands over control to the parent and it done okay.

So it is very very simple operations for the node, it is very well minus defined and clear okay, what about the leaf? The leaf whenever it receives its belief, it will check whether it is frozen or not, frozen send back 0, not frozen make a decision using threshold okay send back the bit and transfer the control back to its parent okay, so this is the message passing sequence, it is a definite way in which it happens, it is not everything does not work in parallel, very different from the LDPC decoder in that fashion, you remember the LDPC decoder on the Tanner graph everything was working in parallel.

Here it does not work like that, it works in a certain sequence and there is a good reason for it, you remember the bit channels are, they are ordered in quality, there is a reliability sequence, there are very bad channels, there are very good channels, you want to go from bad to good in some sense and there are enough frozen things in the middle to make sure that you do not get diverted too much and end up making mistakes okay, so this is the summary.

(Refer Slide Time: 14:15)



Now let us see for example, a very small example that will follow through the operations, I will indicate what the sequence of operations are okay, so this is the received values from the channel okay, I am calling it R16, this is I think 16,10 code, okay you can see the frozen positions here, whenever I put 0s frozen, this is also frozen, the remaining positions are message bits okay, so what happens first the root receives its incoming beliefs and this is the first operation, so this number one indicates the sequence, sequence of the operation and this

L indicates the step okay, the step could be L or R or U also and this is the belief and this 8 of course indicates the size of the belief okay.

So initially there are 16 beliefs that the root receives from the channel and when it does L in its first step, in the sequence number one L, it goes to the left child okay and gives control to the left child but it sends only 8 beliefs right, it would have done a minsum to combine the incoming 16 into 8, it sends 8 and what does the left child do, so this gets activated now and it will do this step okay, so this is the second step that gets done, it will send to its left child, it will do the L step again, next the second operation that happens is the L okay and but it will send only 4 beliefs right, it got 8 beliefs, it will send 4 beliefs and then what will this child do, this child gets activated now okay.

So once this third child gets activated, it will send, it will do this step okay which is again an L okay, remember whenever node gets activated it always does L first okay, so it does L then 4 beliefs becomes 2 beliefs and then this node gets activated and the fourth step it will do L again and now you have a leaf node, it just receive a single belief and it is actually frozen, so it will send back 0 okay.

So I am not, what I have not shown here is the U step okay, I am not shown the U step at all, just on the L and the R just to keep the picture free of clutter and you can imagine the decisions in the and the U steps coming in right, so this will be a 0 that goes up here okay and then this node will do a the fifth sequence in the, the sequence after four you have five, fifth will be an R okay, the first time you will get an R okay.

So this node will do an R now, step R which will be again one belief and this will send back just 0 okay, so because it is a frozen bit, so it sends back 0, you will have U step here okay which I have not shown okay after this you will have U step and then you have this, the next step is the number six which is the R okay after you have the U, you have the R and then this is the seventh step which is L and then anyway it is frozen you get 0, this is the eighth step okay and then this will send back 0 again because it is frozen and then this will do a U okay.

So the Us I am not showing explicitly to avoid some clutter, so U will happen here, so both Us are happened here, so this node can now do a U okay, this node at this level will do a U and then you are ready to do one R okay, so the R happens after that, ninth this node will do an R because this received its message from the left child, once it is receives the decisions from the left child it will do the R okay and then you do the L again, then the L again, then



the decision 0 will come and then you do the R here you will get the nontrivial decision okay, it will not be 0, it will be whatever that decision is okay.

So that will come up and then once the two decisions are up this will do a U and then you do an R, then you do an L again okay and then your decision comes up and then you do this fifteenth and then you do thing, then you do decision back up again and then you do a U okay, U, U has done here, you will do a U here, then U, U done here, you will do a U here okay, once you do a U here, you can do the R okay and you go down the right side okay and then you continue okay, I have said just and so on, I am not going to repeat the whole thing but this is what happens okay.

So such things are called tree traversals okay, what is the traversal? Traversals is just going from one place to the other and that is what you are doing on this tree right, so you start from the root and you traverse the tree okay, so this kind of traversal has a name, it is depth first traversal okay, so you go left first down and then keep going down and down and down and down on the left and then only then you come back, once you have gone to the leaf you come back.

So those kind of the things are called depth first traversals and a various ways to implement such traversals, we will do that soon enough but this what happens in a successive cancellation polar decoder, it is a very simple decoder to describe, you can picture it on binary tree like this, then you start from the root and then you traverse okay, traverse in the certain particular way, you do some operations when you go either left or right or up and that is it, whenever you have the leaf you make a decision, once all decisions are made your decoders is complete, remember once you make this decision here okay, so this will be the last decision you make and you can stop your decoder okay, so at that point everything is done because all your messages are decoded okay, if you want to you can go back up and compute the code word it is not really needed, that is the message bit once it is decoded you are done okay.

So this is complete polar decoder, this is for  $N$  equals 16, even for  $N$  equals 1024 is the same thing okay, so just that the picture becomes so much more complicated, it is difficult to draw it on a slide, on my own definitely not but you can imagine it is the same thing okay, so you do the same sequence of operations and you will get it to work okay, so this is the end of this lecture and hopefully you got a clear idea of how the success of cancellation decoder works

because in the next lecture, we are going to code this in MATLAB and then get it to work hopefully okay. Thank you very much.