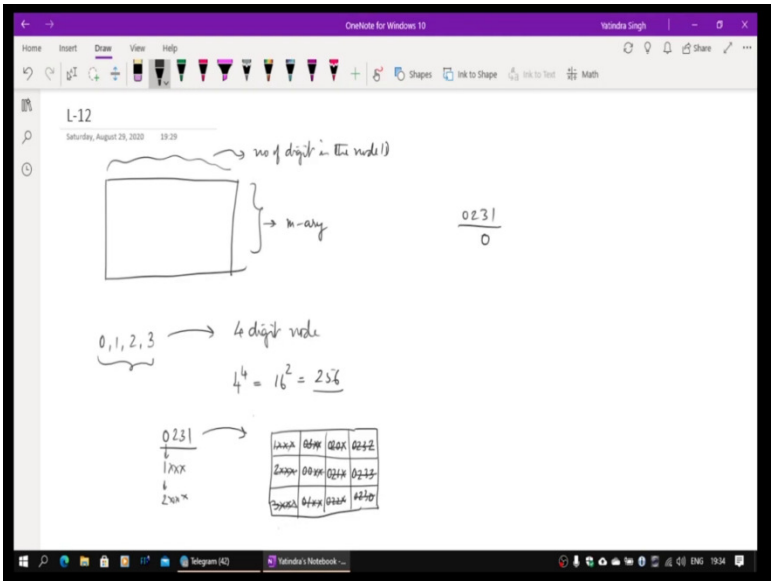
Peer to Peer Networks
Professor Y. N. Singh
Department of Electrical Engineering
Indian Institute of Technology, Kanpur
Lecture No. 12
Understanding the Tapestry Protocol through Example

This is Lecture 12. Now, we will be looking into an example of how the Tapestry routing is built in a system. In the previous lecture we had discussed about the basics of tapestry. So, example will essentially clarify all those concepts which has been discussed earlier. So, basically, the idea is how the routing table will be constructed and maintained. What happens when a node gets added or when it gets removed? And how the query is going to be routed through the Tapestry routing table? In the tapestry table.

(Refer Slide Time: 0:57)



So, we do understand that for a Tapestry, routing table will consist of rows and columns. The number of rows in this case will be equal to number of digits which are going to be there in the routing table. Number of digits in the node ID. And number of rows will essentially be governed by what kind of digit is being used. If it is M-ary digits, there has to be M corresponding rows.

So, to simplify the example, so while in the previous lecture, I had taken hexadecimal digits. For the example's sake, so that it keeps my life simple, I will be taking an octal digits which actually means this will count from, sorry, not octal. I will be taking it a base 4 digit actually here. Not octal, it has to be base 4. So, this will be still simpler for us to handle. Just for that reason, I am

taking this. So, basically any digit can take only these 4 values. And I am also going to take only 4-digit node IDs. We had done a similar exercise when we were actually handling Pastry.

So, in this case, total number of, because there are 4 possibilities for every digit. We will be getting 4 raised to the power 4, total these many possible node IDs which can exist. Which is going to be 16 square and which is roughly about 256 possible node IDs which can exist.

So, we start with a node ID which is 0231. I am assuming it and this will be the first node in the network. So, it has to maintain a routing table. So, normally I will make a modification which I had suggested earlier, in the previous lecture, that whatever is the row for the current node, I need not maintain that because all those entries, I mean all those k entries, if there are k digits in the node ID, will actually be the same, which will be the current node ID. So, I need not keep them.

So, technically, I will be now requiring only 3 rows in this case. So, I will be doing that thing first. That is a slight deviation, that is basically an internal data structure issue. You can actually make 4 rows also and then keep on filling one and row entry with 0231 every time. I will not be doing that. 0231 is a node ID.

First column. There has to be 4 columns here. So, since there is no entry, the first row should contain something which starts with 1xxx. So, basically after 0, we get 1. Then we get 2. Anything with 2 actually should come. So, I will be maintaining in this fashion. But there is no node as of now. So, since there is no node, these are blank entries, technically. So, they do not exist. So, I am just cutting them out. But just keeping them, so that you know what kind of patterns are permitted in these entries.

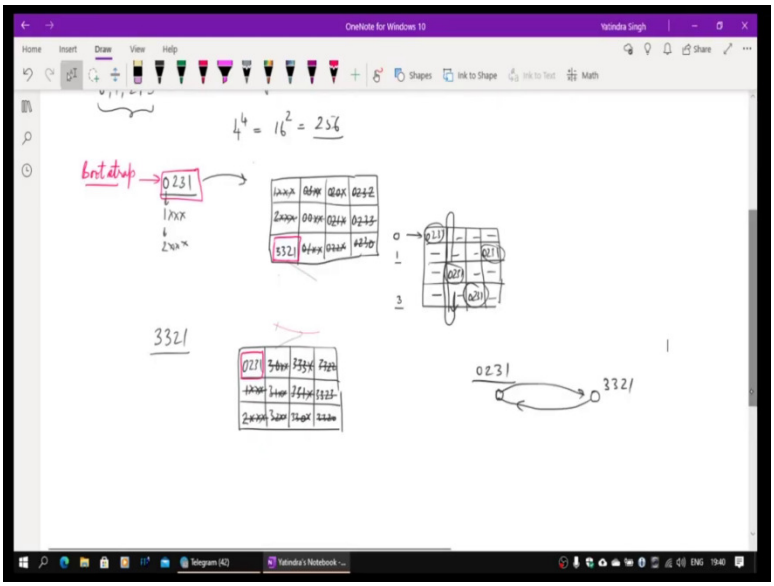
The next one will be containing 0. We have 2 here, so I need to put 3xx. It has to be after 3, I will get back to 0xx. Then, of course, 01xx. But since there is no node as of now, I will again strike them off. But this is the pattern which should be there present in that. For the third entry, I will be starting with 02 and this is 3, so I should start whatever happens in a cyclic way, so after 3 comes 0.

So, I should take any pattern which exists, allowed. 021x and 022x. So, any pattern which matches with this has to be put in there. And the third one will be, obviously, because there is 1, so I will now keep 023. After 1, we get 2, so I will put 2 here. 0233. And after 3 we get, 0. Since

there is no other node ID which is there currently in the network. Only one node which is existing here. 0231. So, I will be, there is no entry. I just struck them off. This is the blank routing table which exist here.

So, let us add one node at a time and see how what happens to the network. So, I am going to just copy this thing, so that I can keep on manipulating. I will keep on moving it actually around.

(Refer Slide Time: 5:41)



So, next entry will be, I can take any node, so I am actually taking arbitrarily any value. I am taking a value 3321, which is the next entry, which is going to be made. So, this is the next entry which is going to be made. Now, this has to connect to Bootstrap node. So, there is only one node. So, this, obviously, is also my Bootstrap node. So, I will keep this as a Bootstrap node. This we had done in Pastry also.

So, 3321 connects to Bootstrap. So, it will normally will request that find my root node. That will be the request which is made and in between also send me whatever is the routing table you will find out from any intermediate node. You also request. So, every guy who actually sends through which the query will be routed, will be sending the routing table. So, 3321 comes to know of 0231 and 0231 will come to know of 3321. So, 3321 will get a blank routing table. So, it has to now only populate the routing table which is going to be based on its own entry and 0231.

So, come to the first column. So, 3321 is itself 0231, after that 3, 0 comes, so first entry will be 0231, obviously. Then after 0, I get 1, so 1 nobody exists, I have to struck it off. But this is the

pattern which is required. And 2, again the pattern does not exist. It is required. So, basically, you are kind of starting a cycle from this digit onward. So, this actually keeps the life simple in implementation. Though at some places you might have seen we will start with, we will have 4 columns like this. This is what we had done in the earlier case, in the previous lecture. So, this also is fine, so far, the method remains the same, root node will be always same.

I could have also written 0231 and there is no entry here, no entry here, no entry here. And then, of course, 0231 and no entry here. So, I am just trying to save these spaces actually. And 0231. So, I am just saving these spaces which I have. Put also another way you can route but, in this case, the first row will always start with 0, then 1 and this will go the last one. So, in this case, it will be 3.

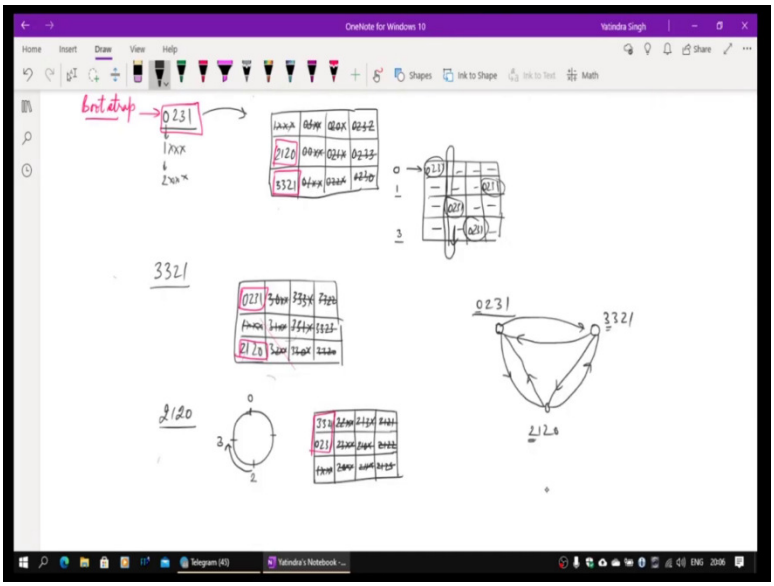
And wherever you are placed, from there you are actually now going cyclically, all the way and then coming back. This is the cycle which starts. I have just written that same table in a slightly different way which actually suits more from the programming sense because I will further be explaining a still, an improvisation on this algorithm which we have been using in our system.

So, 0231 will be here. So, next will be starting with 3. After 3, I get a digit 0. But there is no node on this. 31xx, there is no node available with this. 32xx, no node which is available here. Then 33, I will be requiring 3x, after 2 actually, we get 3, so 331x, 330x. No node is there with this and, of course, 3322, there is no node with this thing, that is why I am crossing it. I am crossing it. There is only one entry which is there, so which is being marked at this position. This is the only entry which is there in this routing table.

Now, 0231 also comes to know of this information. We need to also update this own entry. So, 3321 has to come in somewhere. So, 3321 is going to match with the entry, so this will be updated and we will have now here, 3321. So, this is a valid entry now. So, I have removed it from the crossing.

So, now I have another node sitting in here with this guy and this makes a pointer to it and this maintains a pointer to this. I am just writing these routing table entries as the pointers. Which I have maintained. So, let us now take it out and move it further. So, I will keep on evolving it.

(Refer Slide Time: 10:54)



So, next node we need to take up is going to be taken. And obviously the 0231 will become root node for 3321. So, there is no other node, so no search is required. So, next node we have to take up now.

So, next node we will take 2120. So, 2120, again, I need to choose a bootstrapping node. So, bootstrapping node in this case, I need to take the same one, 0231. So, 2120 will be requesting 0231 to tell what is going to be the my root node. So, 0231 will now search for it and it will find out, where it matches. So, it will now try to find out, it will say 2120, 2120, there is no match. There is no entry here. So, it has to decide now, who lies actually 0, the first digit here. The first digit here and first digit here, where it lies. So, it will now build up a circular arrangement of the first digit 0. There is going to be somewhere a 3. 1, there is no entry which is existing. 2, is now we are looking for. 2 is closer to 3 actually. So, 0231 will send its routing table to 2120. And then the query will be handed over to 3321, because that is closer as per this cycle.

So, remember this is the same way we have been doing in, this is like a chord within a circle. So, once we do this, this should happen to be 1xx, 2xx will not be there. It should have been crossed. So, now let us make the table here. So, 2120 will come to know of 3321, so it will immediately put 3321 as the next entry which is possible which this was blank earlier. And then 0231, after 0, it will come. It is 0231. 1xxx does not exist. It will be crossed. Then you will have 22xx not

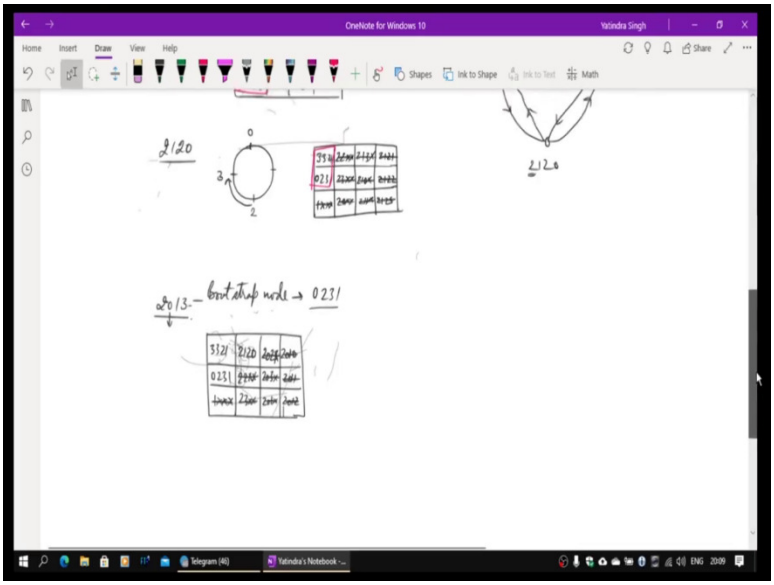
there. 23xx, not there. 20xx, not there. 213x, not there. 210x, not there. 211x, not there. And similarly, 2121, not present. 2122, not present. 2123, not present.

This will now be the routing table. These are the two valid entries which are kept. So, 2120 will have this routing table entries, 3321 and 0231. Similarly, now once, 3321, comes to know of the information because it is the root node, so information goes to 0231 from 3321. So, 0231 also has to first of all make the correction. It will make 2120 entry because now this entries can be pushed here.

So, it will be now 2120, which will become a valid entry. Now, 3321, once it comes to know of this information. So, this will also now make an entry here. This will make an entry here. It will become 2120 and 1xx, of course, remains as it is. That is how the routing table entries will get updated at both the places.

So, now all 3 has been updated and we have valid entries. So, this will become 2120 as it is. There is a valid entry here. There is a valid entry here. 3 nodes and each one of them are pointing to each other. So, I can now write down the third node here. So, this will be, 3321 will be pointing to this. This will be pointing to 3321. So, all will be pointing to each other. So, they are actually from different domains. So, this starts with 2. This starts with 3 and this starts with 0. So, they all need to point to each other. So, that spot will be the updation.

(Refer Slide Time: 15:34)



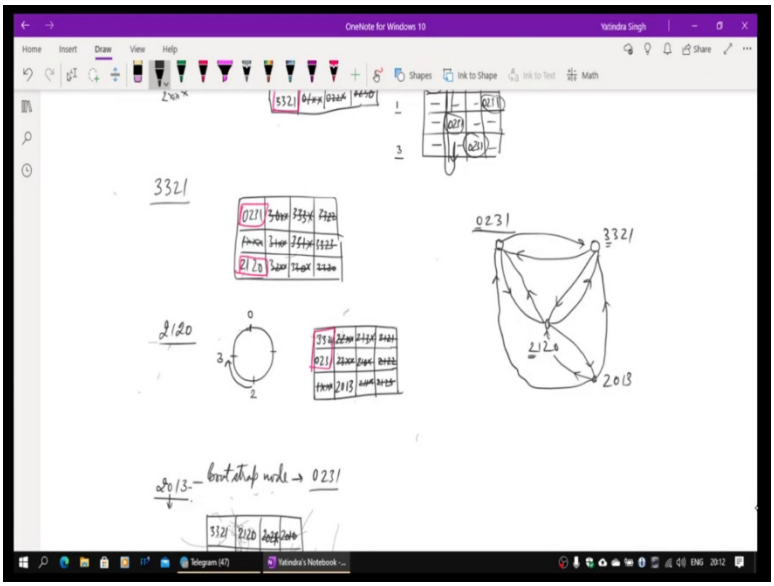
So, next step we are going to add one more node. So, this time let us choose something which is going to have the same digit as we had already added. So, we will be adding now this time 2013. Let us see what is going to happen with this.

So, again, Bootstrap node is going to be here for this guy, 0231. It will be the Bootstrap node, so I have to look at that particular routing table. So, let me see if 2013 goes here. So, 2013, once it comes, it will be simply matching with 2, the first digit. So, 2120, it will be handed over. Once 2120, it has been handed over, it will look into this entry. 2120, there is no other node which is closer than this. So, 2120 is not going to have any entry in the second column. So, this is the root node for this. So, 0231 will fetch the information and the routing table will get updated.

Once the routing table is updated, it will have again 3 rows. So, first of all, it will come from 0231, the routing table entries. So, from 0231, it will get to know of 3321 because those are then, of course, 0231 will give 2120 and 3321 both will be informed. It has to pick up now entry. So, after 2, we will start 3, so 3321 will be present. And 0231 will be there. There is no entry with 111. So, this will be crossed.

So, it will then also come to know of 2120. So, 2120 has to be made somewhere. So, it will be coming, 2120 will be coming because this is immediately after 0, the 1 will come. 22xx entry does not exist. 23xx does not exist. So, this is the only entry matching here. And then we do not have, 2023 does not exist. 2x, sorry. 203x does not exist. 201x does not exist. And then similarly, 2010 does not exist. 2011 does not exist. 2012 also does not exist. So, these will be the 3 routing table entries which will be there in 2013.

(Refer Slide Time: 18:28)



And while now, 2120, because my routing table the way 2013 has been informed is first is 0231 and from there it went to 2120. So, these 2 guys will be now updating their entries. So, 0231 need not, actually. There is only 2120 could have been replaced. So, again, as I told that if you are using proximity metric for optimization, then if 2120 is far off compared to the current node which we are trying to insert 2013, then only you will do the replacement, otherwise you can retain the same entry.

Or if there is no proximity metric **two**, you can just leave it as it is because you have already filled up that entry. Now, this actually also goes to 2120. So, 2120 is now aware of 2013, so where the entry has to go, it has to decide. So, 2013, it will now, 2120 will now move and we will find out that it is about, the next thing, it is going to match with this particular entry, 2013. Because that is the second digit, they are actually having a change. So, we need to make this change. So, you can get 2013 here. So, the way 2013 has 2120 here, it will have 2013 here.

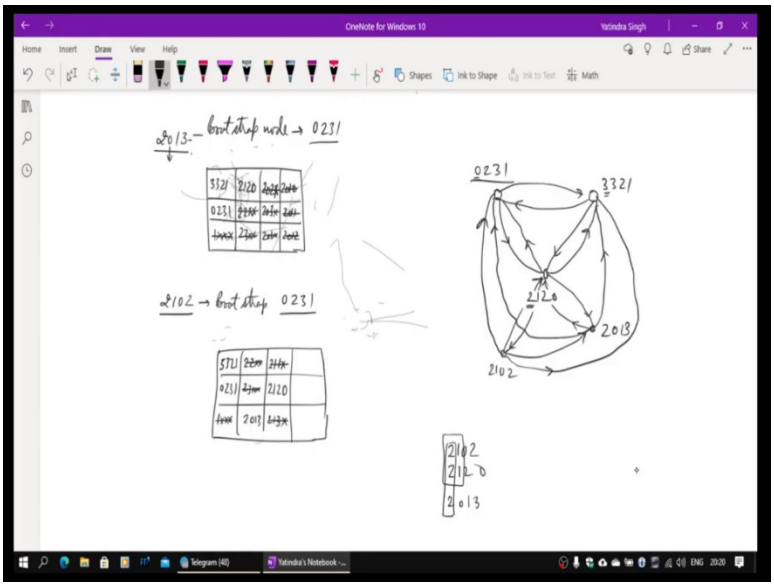
Now, what will happen to 3321? There, there is no update. So, now everybody, 2120, for example, will be doing all routing table exchanges with all the neighbors which are listed here. Okay, with all the 3. 2013 will also be doing. So, 3321 will become aware of both. So, it has an option to choose 2120 or to go to 2013, any one of them. So, otherwise, it is not required.

So, this will be like now creating a third entry. So, let me add that. If I add that entry here, so this is, sorry, I should actually keep it somewhere here, 2013 and let me remove this one. So, now

what you can see is 0231 is not going to, it is going to point to only one of them. 3321 will also be going to point to any one of them. But this guy is going to make a pointer to it and this person will also be pointed by this thing.

Now, 2013 has an option to either point to these, so it will be creating a pointer to these actually. So, the reverse pointer is not there because reverse pointer will be only through this 2120 which is the entry to this 2xxx partition. So now the routing table has been updated.

(Refer Slide Time: 21:39)



Let me add now another node, just to ensure, to actually show the example what happens. So, this time, we are going to enter, make an entry of another node which is going to be 2102. So, I have done it intentionally. 2102, to show an example what happens. So, when 2102, again Bootstrap node is going to be chosen as 0231, Now, let me also put down this figure. This whole thing. So, now, once it goes to 0231, it will ask for where is my root node, search for it. So, 2102 will be handed over to 2120. So, 2120 and 3321 will be informed back. Though the routing table has to come back actually.

So, routing table is going to come. So, we will populate this and so you will come to know of 0231, which has to come after, so first thing which has to come is, come in 3 with 3321 which we have already in our routing table. That information will come from 0231. 2120 will be informed. There are only 2 entries. 2120 and 3321 which comes from 0231.

So, 2120, where it is going to go? So, there has to be an entry which is 0231, which will come and, of course, there has to be 1xx which is not there as of now. Now, we also get a information about 2120. So, 21 matches with this, so it has to go to third column whereas 210 is there. So, you need to make 21, there has to be 1 here. 1x has to come in. This entry does not exist. 212x, 2120 will come here, sorry. There is 2120 which will be coming here which is the entry which has been learnt from there and 213x which also does not exist.

And then there will be no other information which is available from 0231. 2120 and 3321, both have been now inserted. And, of course, then Bootstrap node itself. Now to whom to 0231 will hand over? It will hand it over to 2120. Once it goes to 2120, 2120 will give him another information about 2013.

Now, this 2102 is not going to have any impact on the routing table of 0231. The entry is already exists, 2120. Same is going to happen with now, we have come to 2120 now. So, 2120 will get affected by 2102. Let us see if it gets. Yes, 2 bits are matching, so it has to essentially go here and it has to now update this particular entry.

This will be 2102, which will be created and, of course, it will, 2102 will be, once the information is available, that is being updated, 2120 also tells about 2013. So, 2013 has to be populated somewhere. There is 21, so there has to be entry, 22xx has to be there which is not present. 23xx not present and 2013 which will be populated here. That is how entry will come.

So, this has learnt from 2120. So, 2120 gets updated. This also gets updated. But 2120 will now search for, will become the root node for 2102. That is the only node which is, because 2102 is the new entry being created. When it was not there, the search would have ended here itself. In third column it would have come and there is nobody, so 2120 is the root node. So, that is the update which has happened.

Now, only problem now which is left is how 2013 will get to know of this new guy. So, next time, this routing table is there, so this invariably will it send its routing table copied to all these people in routing table management. So, 2013 will also come to know of it. In fact, 2120 will also be doing this. And once, 2120 is doing it, so this is going to update the value. So, it has to go to now, we have 2120 or 2102, one of the only two entries can be kept here. There is no

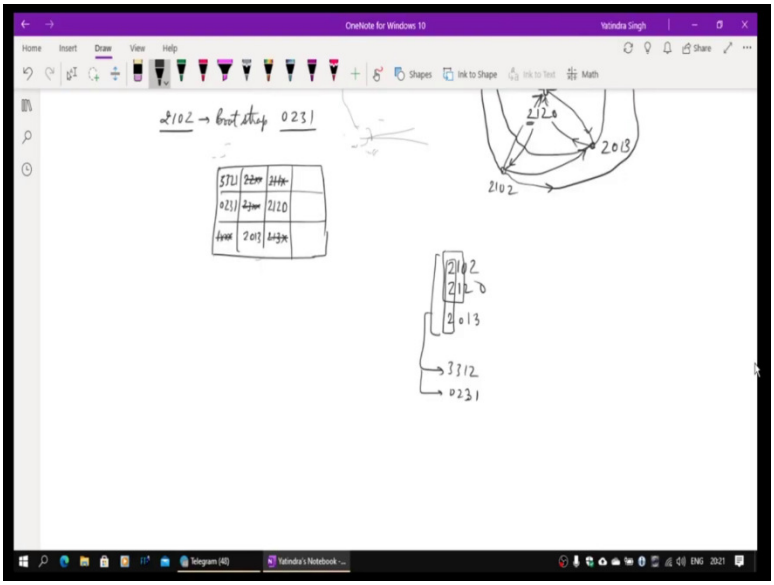
other entry. So, this may not actually change. 2103 will remain as it is. So, now situation will be like this.

You will have another node 2102, so this points to this guy. This points to this person. There has to be some error which I have created. This figure actually should go from here to here. So, that will be additional entries which will come and it will also now point to 0231, and of course, this guy.

So, that's how. And 2120 will also be now maintaining an entry. So, 2120 will also maintain an entry for this and will be having an entry coming back in this direction and again these two. So, 2120 and these guys will be mapping to each other. But there is only unidirectional mapping which is going to happen from here. 2013 will not be maintaining an entry. That is how the routing table will get updated.

So, this will keep on happening. If a node dies off, again routing table exchanges will ensure that you get connected to the partition. So, remember now, 2102 and 2120, they are in a smaller partition. Now, this whole thing, are in another partition which is 2013. So, these are actually smaller partitions, so that is what is happening here.

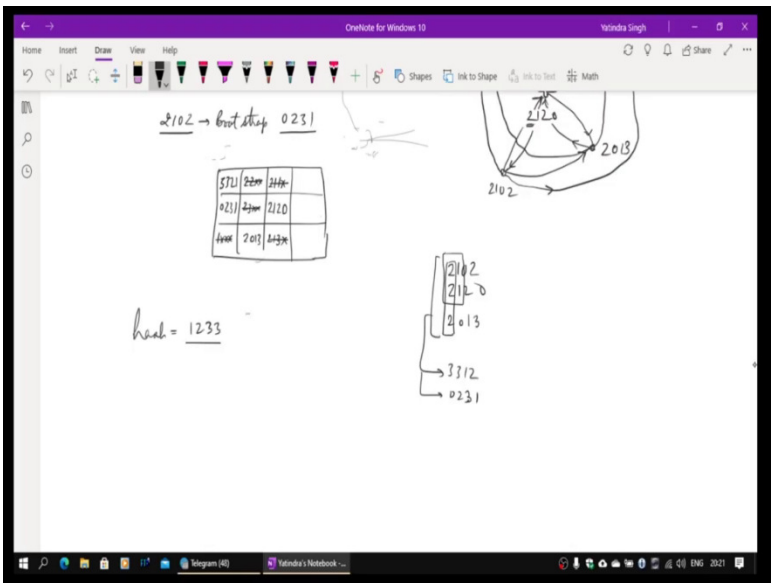
(Refer Slide Time: 28:49)

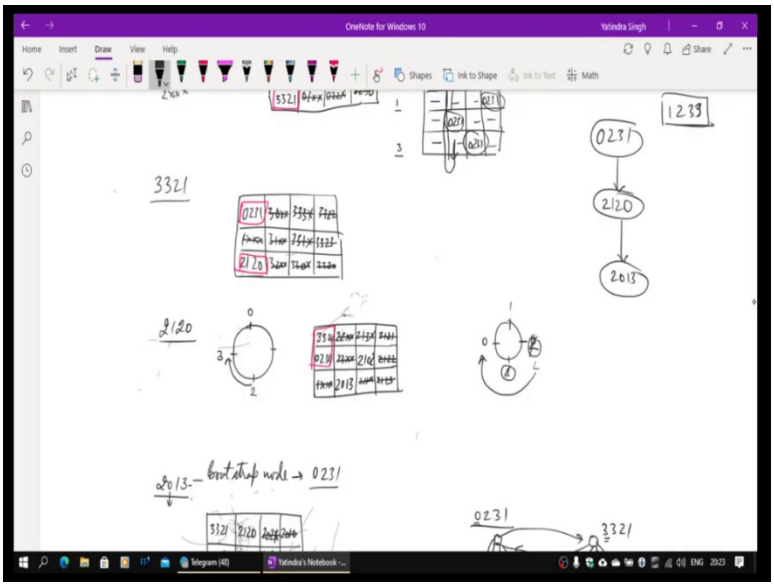


And then, of course, you are going to have, any one of these entries can only be there in 2013. That's the meaning. And these, everything is now going to be this whole thing is being mapped

to 3312 and 0231. So, any one of these 3 and each one of these have to be maintained in the highest level. So, that's how this routing table is going to be there.

(Refer Slide Time: 29:13)





Now, you can actually try searching for something. So, you want to, for example, search for any node, any hash ID. So, hash ID you can take as 1233, for example, where it will go? You give it to any node; it should be able to go to the same root node. Let us try it out with these given nodes.

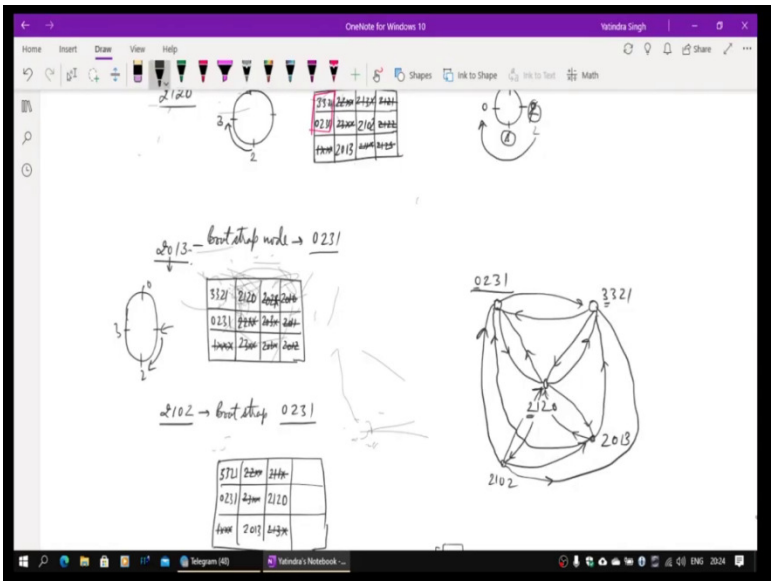
So, 1233, let us see what will happen to them. So, 1233, for example, if I start with 3321 or say, 0231, where it, how it moves and whether it goes to the right guy or not. So, if it starts with 0231

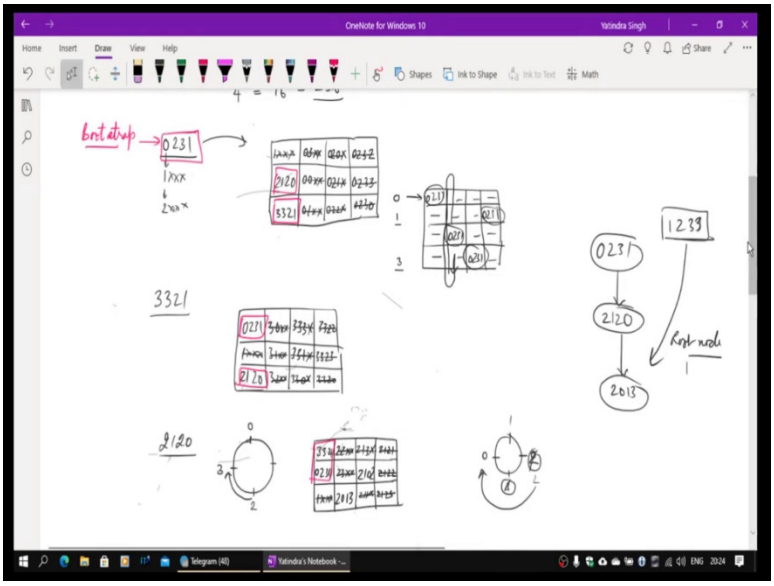
and I am talking about 1233. So, when it goes to 1233, this goes 0. 1 is not there, so it will go to 2120. So, 0231. So, I am going from here to, I am actually searching for 1233, that is a hash ID which I am searching. For 0231, I will go to 2120.

Now, 2120, when it will search, it will find out 2120 itself is the, in the first column search, it is starting with 1233. So, it will always come to back to same node. So, it has to go to the second column and it will find out to whom it is going to match. There is only 2013 and 2120. So, there is 2 which is available. So, this is going to go to 2013. Okay, because there is, 1 is there. There is a 0.

Sorry, this is 0. This 3 and this is going to be 2. These 2 nodes, there is no node correspondent to this in the second column. You only have 0 and 1. So, when 2 is going to be come, so this is going to be the next guy. So, 2013 will be handed over. It will go to 2013. And once it goes to 2013, we will now, we have to figure out, where it will end from there.

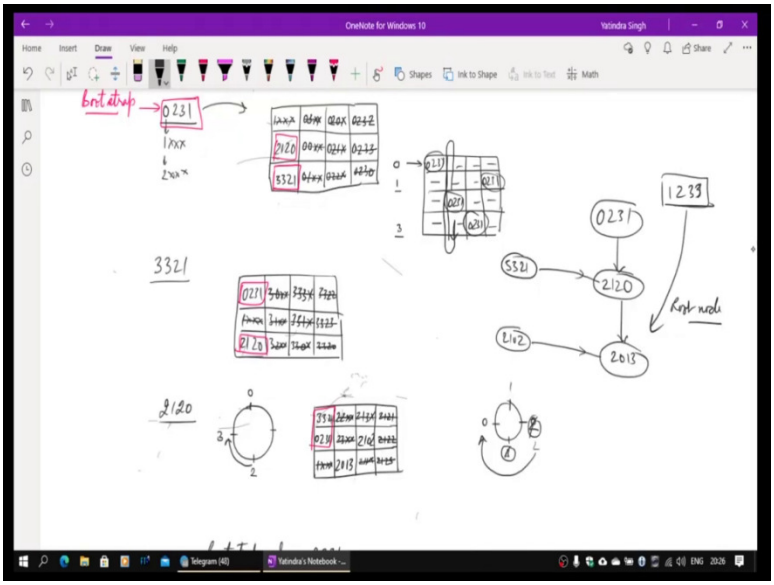
(Refer Slide Time: 31:23)





So, let us go to 2013. 2013, it will now search for, again, start. So, it will look for 1 in the first column. So, 0 comes, 2 comes, so in the cycle, it again will see a 0, a 2 and a 3. So, 1 will be somewhere here. So, it is going to always map to 2. So, 2013, so it goes to next column. It will search for now 2 because we are looking for 1233, remember. So, once it looks for 2, so I have in the second column is 0, then 1. So, it will come back to 2 only which will be 2013 and there is nobody after that. So, 2013 is here the root node. You start with any node; this will be the root node for this guy. So, we can try for some other node and see what happens. Let us see.

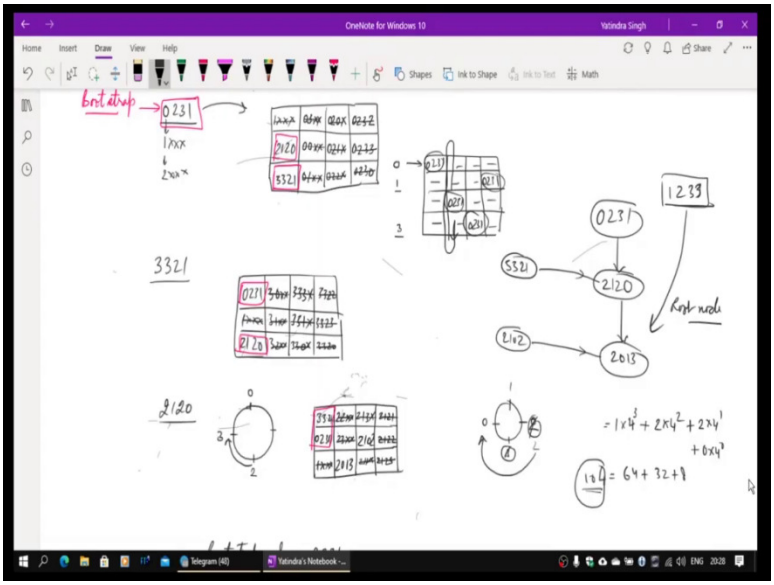
(Refer Slide Time: 32:23)



So, if you are actually searching for say, 1233, 3321, what is going to happen? So, 3321, if you come, so it will search for 1. So, it is 0, 2 and 3321. So, 1 lies between 0 and 2, so it will be handed over to 2120. It comes to 2120, it will come here. So, once it searches, it stuck at first column, so it goes to second one. Handed over 2013 and it will come from there. So, if you actually start with 3321, then also you will end up in the same place.

You start with something else, say, last one. 2120 is there. We also have 2102. And let us see what is going to happen with that. So, if I start with 2102, then what is going to happen? So, if you start with 2102 and you are trying to search, so 2102, when you will do, it will come, we will search for 1, so it will stuck with 2102 in the first column. Go to the second one. Second one again, it will now search for the second digit. It has 1, 0 and we have 2. So, 2 will again land up with 2013 and you will end up in directly to 2013 here. So, in anyway, there is going to be unique root node, which is 2013 or 1233. So, for any node this can be done.

(Refer Slide Time: 34:08)

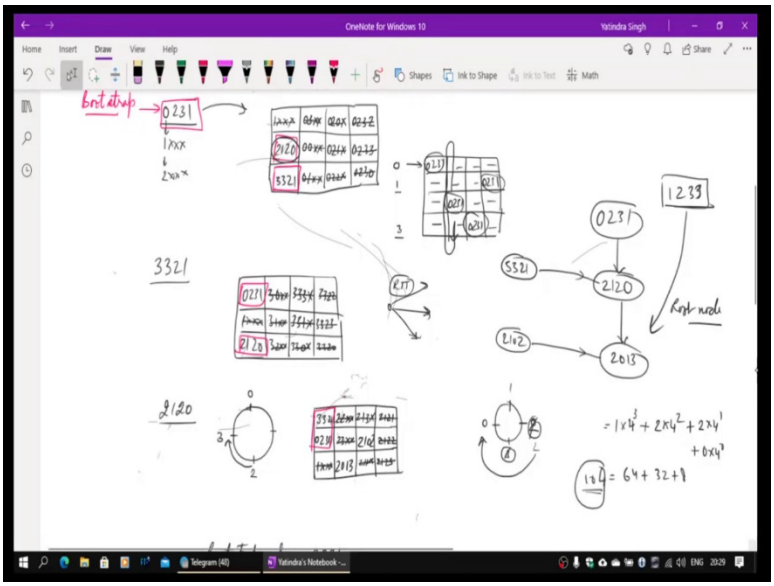


So, this was an example by which the tapestry actually work. And you can find out the distance actually between these, so doing in a modular operation from this hash ID to this node ID distance, you want to measure. So, first digit 2 minus 1 is 1. It has to be, because it is a 4 ary system. 4 cube plus 0 minus 2. So, 0 minus 2 is minus 2. Minus 2 plus 4. It will be 2. So, 2 into 4 square plus 1 minus 3 is going to be minus 2. So, this will be again, because you are going to

add and do modular 4, it will become 2 into 4 raised to the power 1 plus 3 minus 3 is 0. So, 0 into 4 raised to the power 0.

So, this will be the numerical value of the distance which will be there. This becomes 64, 32 plus 8. So, that will be the distance which you are going to have. So, 104 will be the distance and you take 1233 with any other node, the distance will be larger actually. So, you can actually do that and this will be a larger value than 104. So, 104 is the least distance from the hash ID to root node. So, 2013 will be the root node in this case.

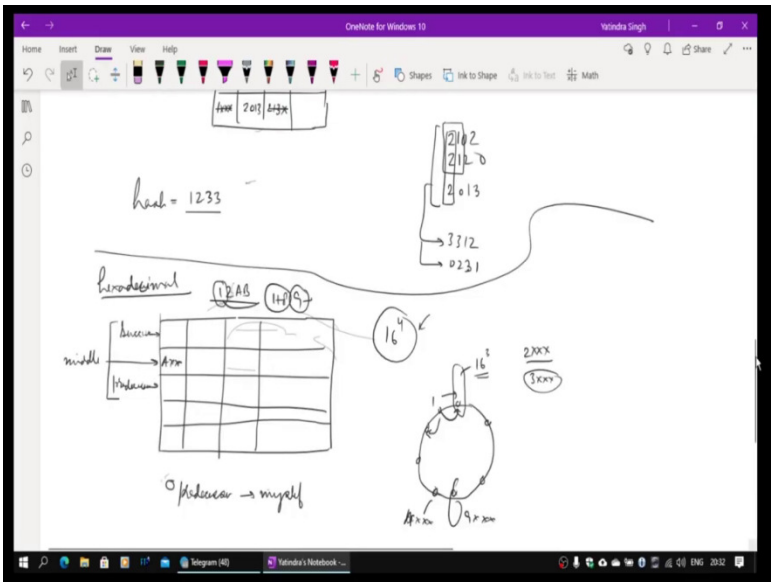
(Refer Slide Time: 35:35)



Now, what we did is, when we are actually doing this conversion. So, first improvisation as I said earlier, we can actually do, for example, we had a choice of putting up 2120 or the other combination. Other any one of 2013 or 2102. Any one of these 3 can be put here by 0231. So, 0231 should actually now do RTT test, there's a proximity metric, whatever you are deciding, maybe number of hops, RTT, geographical distance. To all these 3 nodes. Whenever the opportunity comes, whenever you have an option, you always choose the most optimal one and keep that entry here.

So, this way you are going to use less internet resources when you are actually doing the DHT routing. So, that is a proxim so far that you have to maintain a neighbor table, the way it is designed in Pastry. So, this can be a modification to this thing.

(Refer Slide Time: 36:20)



We further modified this protocol for the ((Refer time : 36:13) 4. So, typically, what happens when we are taking hexadecimal digits, what will be the routing table will look like? The routing table will be now then of size, will have 16 rows. Actually 15 rows, 1 more will be for 2 itself. So, let me see, if I can actually show you or you can take any number, so we will do that.

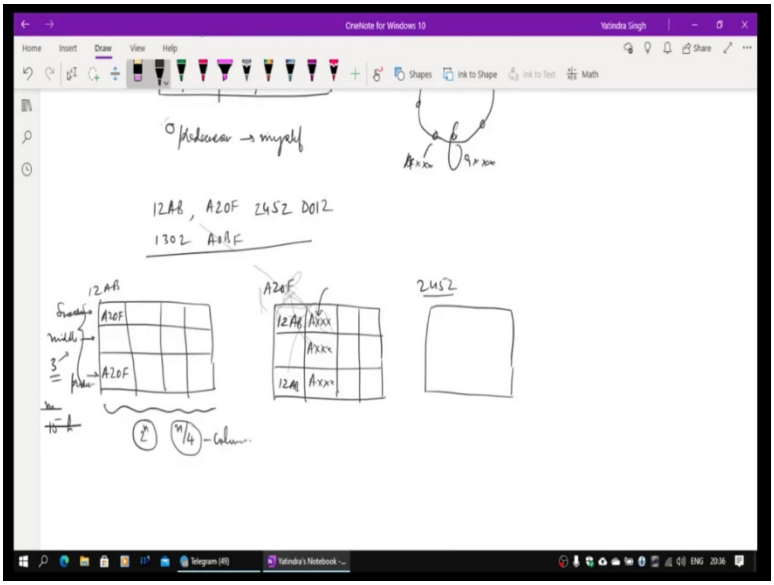
So, in the first column, so for example, if I am taking hexadecimal digit now, so what we will do is, if my number is, say, 12AB, for example. So, I am only still looking at a 4-digit system. But this going to be 16 raised to power 4 possible node ID space I am looking at. So, first thing has to be, so this, I will only look at this first digit. So, first digit is going to be there. I can put all the nodes, all 16 partitions in a circular fashion. And I will only keep, if this is the one. I will keep a successor. Immediate successor.

So, there will be actually, in this case, for example, there will be 16^3 possibilities which can exist. And some of them will, node IDs will be allocated to nodes. Only those node IDs which are allocated can be used. So, one of those entries, I will be keeping here. And one I will be maintaining a pointer to the predecessor. So, if 1 is there, so whatever is the next ID. So, if there is no node with this pattern available, I will search for then 3xxx. If even one single node, so 1, after that I will have 3. So, I will maintain a successor. I will maintain a predecessor. And in order to make it fast, in fact these two are good enough, but we can also make something which is middle.

So basically, I will now choose, if this is the 1 digit, I will add 8 to this. 1 plus 8 is 9. And whichever is the successor of 9, from that block, if 9 is sitting here, for example, but I do not have anybody with 9xx, but I have somebody with 10xxx. A, actually you should call it. Then I should use that Axxx here. So, we use only these 3. So, in this case, normally, I will just hand it over to the any node which is higher than the digit in the hash ID. If my hash ID digit is actually between my predecessor and myself, then I will just move onto the next column.

So, I will keep on doing it, moving on to the next column. The moment you go to the last column or you do not find any entry in a column, you should be the root node in that case. So, that is a technique which we use, which is actually much faster.

(Refer Slide Time: 39:33)



So, let me take, with I think, very few examples how this actually happens. So, let us take a node which is going to be, say, I have already taken a node 12AB. So 12AB has already been taken. This is the first node. So, let me take another node. So, few nodes and then I can maintain the tables.

That is the node 1. So, let me take then, A20F. Maybe 2452. Maybe D012 and then I can take some node like 1302, A0BF. Let me just take this and see what happens. Now, I am going to build it. I have taken something arbitrarily again here. Just to show it as an example. So, I will just keep on creating the routing tables with this.

So, first of all, let the node come one by one into the system and then see what happens. Now, good thing is that I am still now making number of digits. So, if there are n bits which 2 raised to the power n node ID space is there, I am taking n by 4 columns. And every column will have only 3 entries now. It is not equal to m. It is not equal to 15, in our case, in case of hexadecimal. I am not using this. It is not m. It is going to be always 3. I will be always choosing successor. I will be using a predecessor and a middle entry.

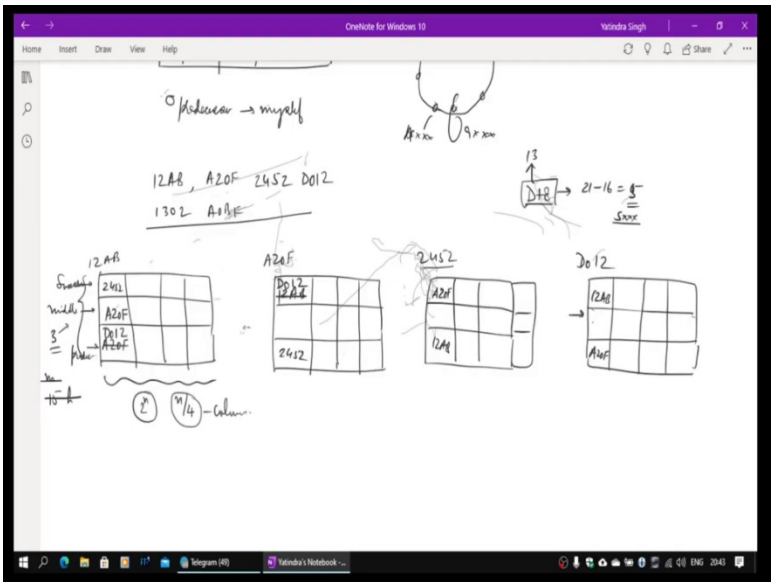
So, let us start with first node, 12AB comes in. So, this is the entry for 12AB router table. So, there will not be any node, so everything will be empty. So, I am leaving it blank, there is no entry. Let us assume that. I will only put where the entries are there. So, unnecessarily, I do not have to delete the things. So, next entry I will now make what? A20F, this is the new node which joins in. So, this will connect to 12AB. So, if all these nodes are there, how our routing table will look like, after the stability.

So, 12AB when it comes, there is no entry. A20F comes, so it will make, only A20F is there. So, it will make an entry A20F at successor. After 1, the A actually comes. And after, this also will act as a predecessor actually. So, there is no middle in this case. It will also act as a predecessor. Only middle can come, if there are actually 3 partitions. So, 1 and A, there are only 2 partitions as of now. And each partition has only one node. So, same way happen the A20F. It will now maintain 12AB and 12AB, there is no middle, it will be vacant.

Let the third node comes in now, 2452. So, when 2452 joins, it will talk to 12AB, get the routing table and based on that we make an update. So, they will actually keep on talking to the people and keep on updating. And the good thing is that if you are actually having 12AB, A20F, A20F is going to send its routing table. So, it will also be giving you lot of other entries which are here in the second column, which all will be starting with A.

So, they will all be starting with Axxx. Axxx, based on this is going to be the next higher bit. Maybe 3A3 can be there or anything, A1 can be here. So, anyone of them is going to be closer. You will simply replace your entry actually. In this case, you need not, because you are representing a block. But some node 2, for example, comes in, so this can be replaced by that. So, there should not be an issue.

(Refer Slide Time: 43:43)



And let me now remove, 12AB this remains as it is. Now, let us come to A20F. So, 2452 has come in. So, 2452, the moment 12AB comes to know of it, it then figures out it is better entry than A as a successor. After 1, the 2 actually immediately comes, so this guy will be removed and once this is removed, I will now make an entry which is going to be 2452. This A20F will remain as predecessor, because 2 cannot come after A. After A, there is 1, so there is no middle, other entry which is coming. If it comes, we will replace that.

So, now what will happen to A20F? So, A20F will now figure out. A20F once it sees, it will be coming to, A20F will look at 2452. So after A, what should come? 12AB. 2452 can only be a predecessor. After 2, A comes. So, this entry will be now removed. This entry will be purged. So, let me remove it. So, I will make it 2452.

Now, similarly the entries here now need to be made. 2452 will come to know from 12AB about 12AB as well as the other entry which is A20F. So, now it will replace the entries here and 2452 will now put in, after that A comes, so it will be putting A20F and after that there will be 1, so it is 12AB. So, all three entries have been put. So, they are actually becoming, they are only 3 nodes. They are acting like a predecessor and successor in this case.

So, now coming to next entry, D012. When it comes, so there will be third table which have to be created. Let me create the third one. So, it will again talk to 12AB and it will come to know of all 3 nodes. So, this has to be populated soon. D012, after D, what comes? After D, you will get

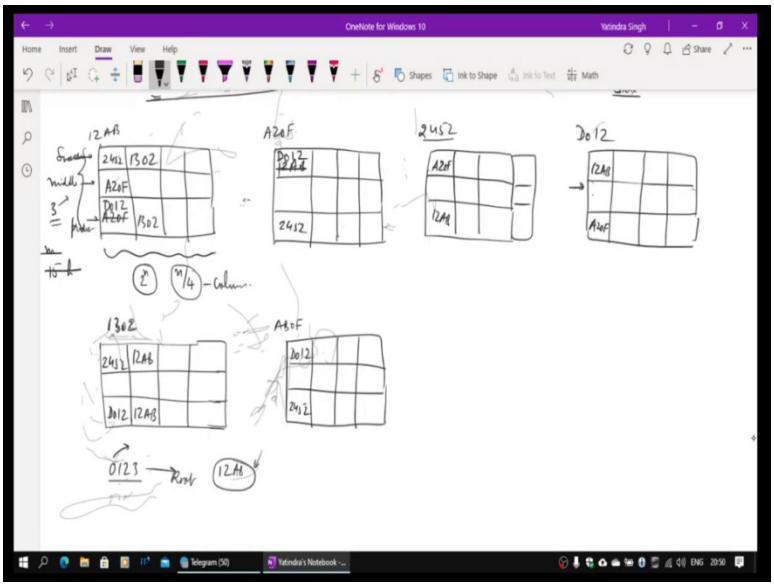
actually 1. So, it will now make 12AB as the entry. And after 1, what comes? It will be, before D, just A actually is going to come. So, it is going to put A20F. In the middle, there is only one node as of now which is available between 1 and A which is going to be 2. So, if some better entry comes, that is going to be put. But, of course, as I told in the definition, it has to be D plus 8. If anything comes, then only I should put that entry. So, I will not be putting this. It will remain vacant.

So, D plus 8 is going to be, this is 13, 21. 21 minus 16 is going to be 5. So, anything with 5xxx is going to be there then or higher, then that should have been put here, otherwise not. So, we can leave it otherwise. So, let us do this. Now, D012, because once it talks to 12AB, it will be informed. So, after 1, there is 2. Before is A. So, now this will get replaced by D012. And 1 plus 8 is 9, so A is greater than 9. So, A will now come here. A20F. That will be the change.

Now D012, once it actually gets its own routing table, it will be also sending its information to 12, A20F. D012 will also come to know of 2452 when it gets routing table from 12AB. But that entry has not been, has been discarded by D012. It was not required. But it will keep on informing. So, 12AB will inform about D012 to 2452. So, 2452 will now make an entry.

So, A205, let us see what it will do? A205 will come to know of D and it will now replace 12AB by D012. So, A is 10. 10 plus 8 is 18. So, 12AB will not be used as the entry here. And 2452 again, 8 plus 2, 10. So, the next entry is A2. D will not be there. So, D will be, that is the next successor. But A plus 2 is 10. So, which actually belongs to this only. So, there is no middle entry being put. So, this is the entry which will be maintained.

(Refer Slide Time: 49:44)



So, I can actually now do the similar thing for, in the same fashion now 1302, I can actually add. In fact, ideally speaking, when I am looking at middle value, if there is nothing else, whatever is available in between them, I should try to make the best choice. We should actually put an entry, but it does not matter. So, for your predecessor, you should always be able to converge. There should not be any issue.

So, when now, okay, this has to be corrected somehow. This has to be 2452. This is 1302. So, with 1 actually, similarly when routing table exchanges will happen, it has to just hook onto 1. Then it will keep on getting at least this information, the 2452 will tell about the next guy. And so on. So, this will keep on happening and then ultimately you will tend to get to a convergence. Always. Because you are always maintaining predecessor, successor in the larger circle. So, the first bit is making their own quadrant technically here. Using same principle so far, predecessor and successor are there, you are going to always move to the better and better entry.

So, 1302 will converge onto, we have now A0BF also is left. So, let me put that entry also and see what happens. AB0F, you can see is, we have already another entry with A20F. So, this should actually contain the same thing. D012 and 2452. When 1302 will come. So, 1302 will talk to 12AB. Interestingly now, first time, the second row will be now used. So, 12AB will become the predecessor. There is no successor, so it is going to be 12AB at both sides.

Now, for the first digit, anything with 2 will be doing. So, 2452 can be maintained here. Anything lower on the side, whichever is the highest. So, there is no 0, so it is A is only there as of, and D is there. So, D012 will come here. Populating. That will be the entry here. So, once 1302 is known to him, so this will also make 1302. And there is nobody else, so it is 1302, that entry will be made here. Rest everything will be vacant. AB0F, will similarly, we will do for this thing. So, there is no other better entry. So, other columns will be free. So, now you can actually from here, you should be able to find out, who is going to go root node.

So, you want to find out the root node, say 0123, how that will be happening? So, 0123, you start with any node. So, maybe AB0F, you start. So, you will be looking for 0. So, 0 is, again 0 hash ID is being looked. Who will be the node which is higher than this? So, 0 is certainly is greater than, it is between D and 2. So, you will hand it over to D012. So, once it goes to D012, D012 will now look for 0123. It will say successor is 12AB. It will hand it over to 12AB. So, the root node has to be there only with 12AB, there is no other possibility for this.

So, 0 it is not lying between A and D. If it is lying between A and D, it has to go to the next column in that case. So, 12AB, it will come. 12AB will look at 0. If 0 now lies here, 0 lies here between D and 1. So, 12AB is the choice. It has to go to the next column. Next column, the next digit is 1. So, next digit is lying between 3 and 1 only. So, it is 12AB which will be the root node because rest the columns are, for this the root node will be 12AB. You start with any node; this is what the result which will be coming. So, you started with 0123, went to D012, from there to 12AB.

Suppose this one is at AB0F, then what is going to happen? So, this one is from AB0F and then you will do, it is actually between D and 2, so it has to go to D012. D012 will hand it over to 12AB and again it will go to the same place. You start with A20F, the same procedure is going to happen. And if you are at 1302, then what is going to happen? So, you will find out, it is between D and 1.

So, it has to move to second column. Second column, it will now search for where the 1 lies. So, 1 lies between 3 and 2. With 3, it has to take 3,4,5,6 till F and then 012. So, it will hand it over to 12AB. And 12AB with the same procedure will be able to, will become again the root node.

So, that was the improvisation which was done to make the system actually work. So, with that we close this video recording and then in the next one, we will be looking at some more topics.