

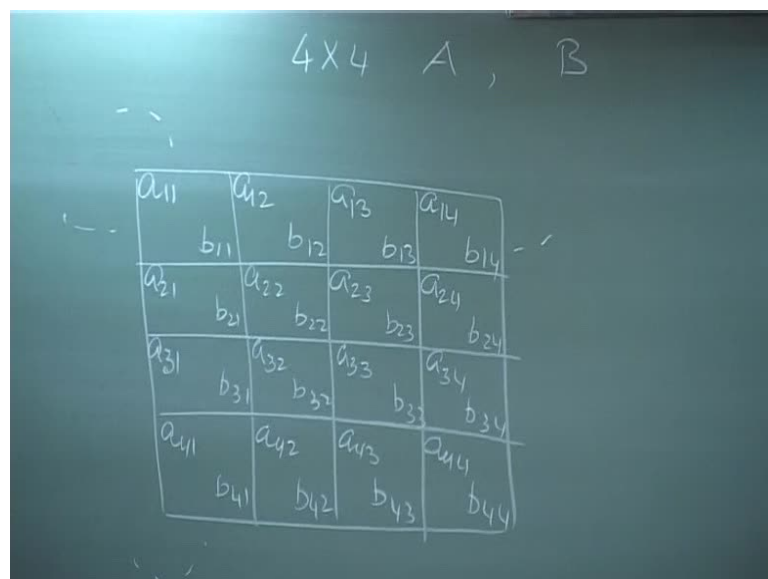
Parallel Algorithms
Prof. Phalguni Gupta
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture - 18

See the last class, we were discussing about the matrix operations and in that matrix operation mainly we discussed the transpose of the matrix, in a matrix operation in that what we were discussing that first one is the transpose of a matrix. As I told you the shared memory model it is not a problem at all, but the problem comes when it is a mesh connected machine. We also discussed about the transpose algorithm for perfect shuffle machine, agreed, and we also mentioned this thing that it is not a good model for matrix transpose or that type of operations.

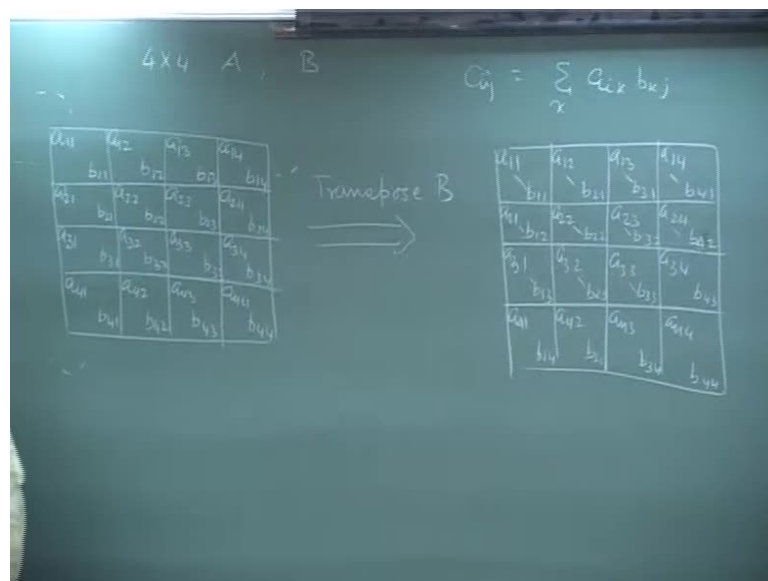
Then we discussed about the matrix quantification shared memory model is not at all a problem; if it is a mesh connected machine then we also measure that the data is inputted and such a way that it becomes a pipeline one and efficiently we can do the multiplication operations, agreed, up to this we covered, yes or no. Now suppose the data is available in the mesh itself because you know sometime I will tell that the data is coming from all the processor and sometimes I will tell no data is available based on our you know how we can solve based on that we are designing that this is not a good thing.

(Refer Slide Time: 02:12)



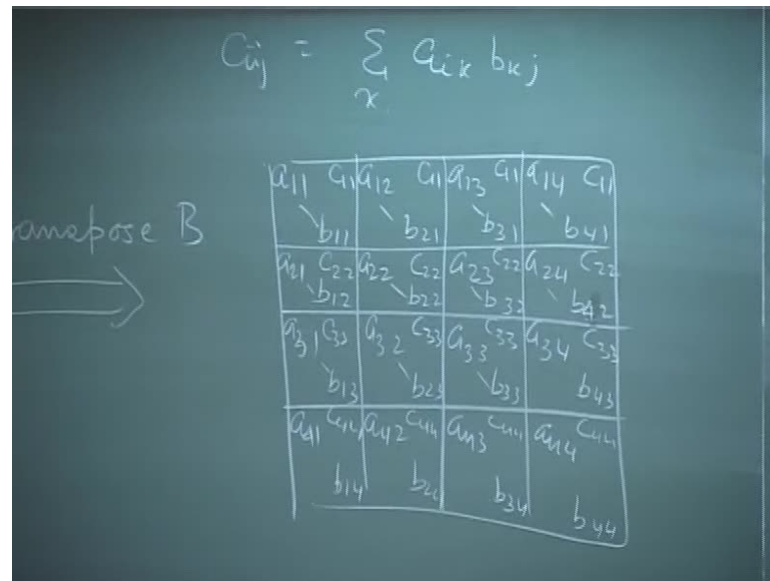
The data is available in the mesh itself. Now spot simplicity is considered one 4 plus 4 matrix and then I like to give it to you as a co-measurement for n cross n matrix multiplication. So, I am assuming that it is a 4 cross 4 matrix A, another one is B, and the model is wrap-around mesh. So, like this you have a 11 a 12 a 13 a 14, a 21 a 22 a 23 a 24, a 31 a 32 a 33 a 34. Similarly you have b 11, you have these two matrices and each processor contains two elements; one is from a elements and one is from b elements, right. Now you observe that if I have to multiply this is this element and not multipliable, these elements are not multipliable.

(Refer Slide Time: 04:29)



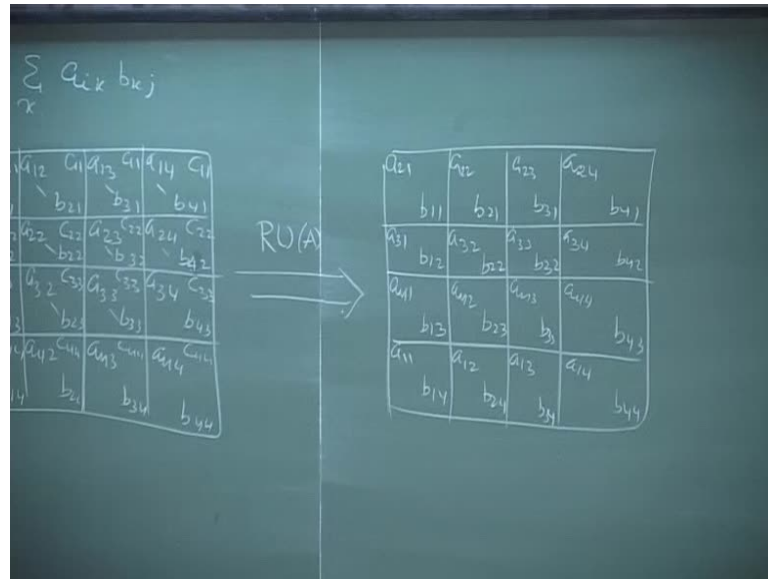
Because matrix multiplication you have $C_{ij} = a_{ik} b_{kj}$ working. So, the two elements are multipliable if ones column index is matched with a row index with the other one. Now in order to make it multipliable you observe that if I make the matrix B I transpose of the matrix then you will observe that it is multipliable. So, transpose of this matrix B if I perform what happens? So, that I discussed in the last class how to do the transpose. In that case you get a 11 a 12 a 13 a 14, a 21 a 22 a 31 a 41, a 42, a 43. Now you observe that every element is multipliable, right.

(Refer Slide Time: 06:55)



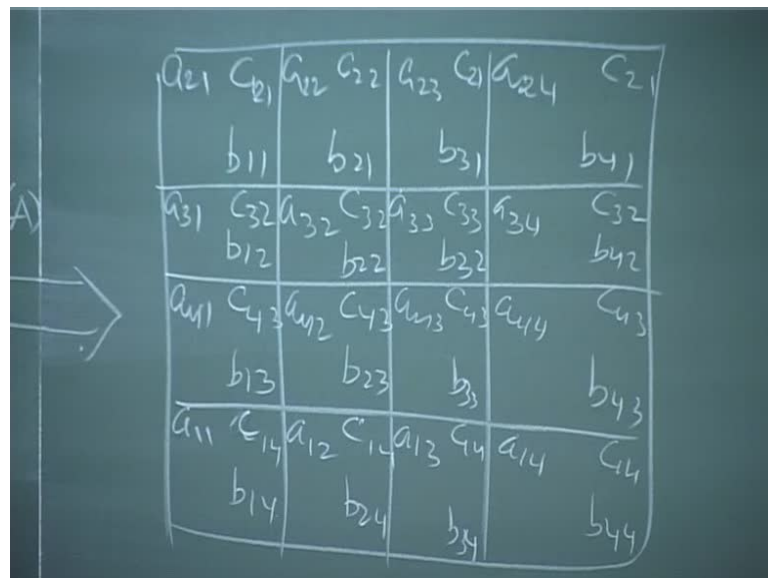
And if I multiply it this is the product of c_{11} , this gives you the product of $c_{11} c_{11} c_{11} c_{11}$, this gives you $c_{22} c_{22} c_{22} c_{22}$, this gives $c_{33} c_{33} c_{33} c_{33}$, $c_{44} c_{44} c_{44} c_{44}$, right. So, at the first quantification, you get the diagonal elements, perform the additions and move the data in to the appropriate positions, half of the addition of c 's and move the data in to the appropriate positions, which you need order and time. If it is n cross n matrix, right, because order and time is okay, because data movement is required. So, whatever procedure you follow order and time you have to deal. So, one way is that you bring 1 by 1 and add it and move the data into the above so order n is sufficient if it is n cross n matrix. So, diagonal element you have fixed, right.

(Refer Slide Time: 08:30)



Then you do the operation rotate up A. So, we told you what is rotate up A operation, because in the wrap around A elements would be rotated up. So, you get a 21 a 31 a 41 a 11, you get b 11 b 12 b 13 b 14, then you get here a 22 a 32 a 42, a 12 a 13 a 14 and a 43 a 44 a 33 a 34 a 23 a 24, b 21 b 22 b 23 b 24, b 31 b 32 b 33 b 34, b 41 b 42 b 43 b 44.

(Refer Slide Time: 10:18)



Now again you observe that you get the situation of multipliability and you get here c 21 c 21, c 32 c 32 c 32 c 32, c 43 c 43 c 43 c 43, then c 14 c 14 c 14 c 14, right. So, you

can again do the addition of the C elements and you can move the data into the appropriate positions in order and time.

(Refer Slide Time: 11:12)

a_{31}	c_{31}	a_{32}	c_{31}	c_{33}	c_{31}	a_{34}	c_{31}
b_{11}		b_{21}		b_{31}		b_{41}	
a_{41}	c_{42}	a_{42}	c_{42}	a_{43}	c_{42}	a_{44}	c_{42}
b_{12}		b_{22}		b_{32}		b_{42}	
a_{11}	c_{13}	a_{12}	c_{13}	a_{13}	c_{13}	a_{14}	c_{13}
b_{13}		b_{23}		b_{33}		b_{43}	
a_{21}	c_{24}	a_{22}	c_{24}	a_{23}	c_{24}	a_{24}	c_{24}
b_{14}		b_{24}		b_{34}		b_{44}	

So, next step again you rotate up A, you get a 31 a 41 a 11 a 21, a 32 a 42 a 12 and a22 a 23 a 24 a 13 a 14 a 43 a 44 a 33 a 34; and here it is b 11 b 12 b 13 b 14, b 21 b 22 b 23 b 24, b 31 b 32 b 33 b 34, b 41 b 42 b43 b44. And you observe that they are multipliable and you get c 31 c 31 c 31 c 31, c 42 c 42 c 42 c 42, c 13 c 13 c 13 c 13, c 24 c 24 c 24 c24. So, you add them and place it in the appropriate place in order and time.

(Refer Slide Time: 13:22)

a_{11}	c_{11}	a_{12}	c_{11}	a_{13}	c_{11}	a_{14}	c_{11}
b_{11}		b_{21}		b_{31}		b_{41}	
a_{11}	c_{12}	a_{12}	c_{12}	a_{13}	c_{12}	a_{14}	c_{12}
b_{12}		b_{22}		b_{32}		b_{42}	
a_{21}	c_{23}	a_{22}	c_{23}	a_{23}	c_{23}	a_{24}	c_{23}
b_{13}		b_{23}		b_{33}		b_{43}	
a_{31}	c_{34}	a_{32}	c_{34}	a_{33}	c_{34}	a_{34}	c_{34}
b_{14}		b_{24}		b_{34}		b_{44}	

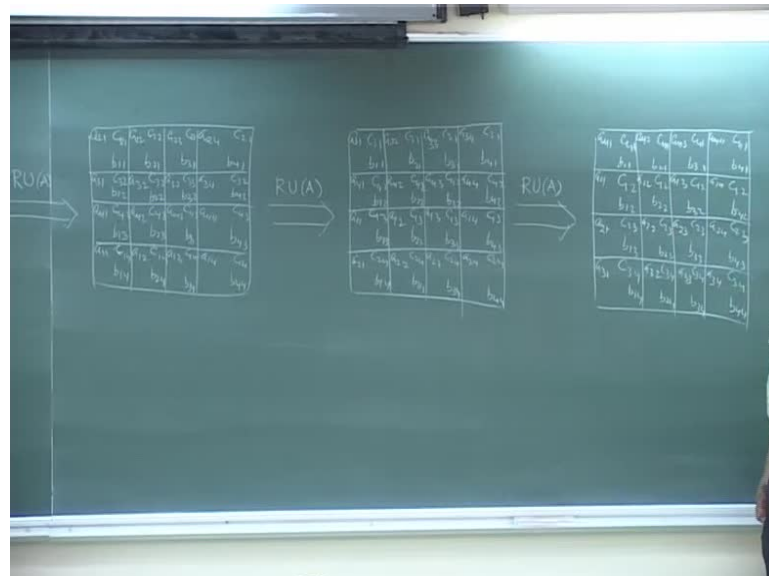
Again, you do the rotate up a 41 a 11 a 21, a 42 a 12 a 13 a 14, a 22 a 23 a 24, a 32 a 33 a 34, a 43 a 44. So, this give you the guarantee of c 41 c 41 c 41 c 41, c 12 c 12 c 12 c 12, c 23 c 23 c 23 c 23, c 34 c 34 c 34 c 34, and you add that and place it in the appropriate position order and time. So, now the question is coming that what is guarantee that it has correctly generated all the numbers of sequences, right. If you can show that rest is. So, first let us assume that every iteration you are generating four elements, right.

Now these four elements have not been occurred in any one of them that is the thing you have to show. You observe that in the first iteration the difference between i and j is zero, second iteration say this second iteration here difference is one, difference is one, difference is one, and difference here is not one, it is three, right. Here all the elements the difference was zero, here all the elements are having the difference either one or three. Here all the elements have difference 2, what about here?

Student: Three n.

Three n one, now the question is coming that you have to show that whatever difference seen here that has not occurred here, right

(Refer Slide Time: 17:25)



So, this part difference is one, second difference two is taken care, difference zero is taken care, you have to think or justify that whatever has occurred here they are not here and within the cluster they are distinct, right, that is the only thing you have to show, and

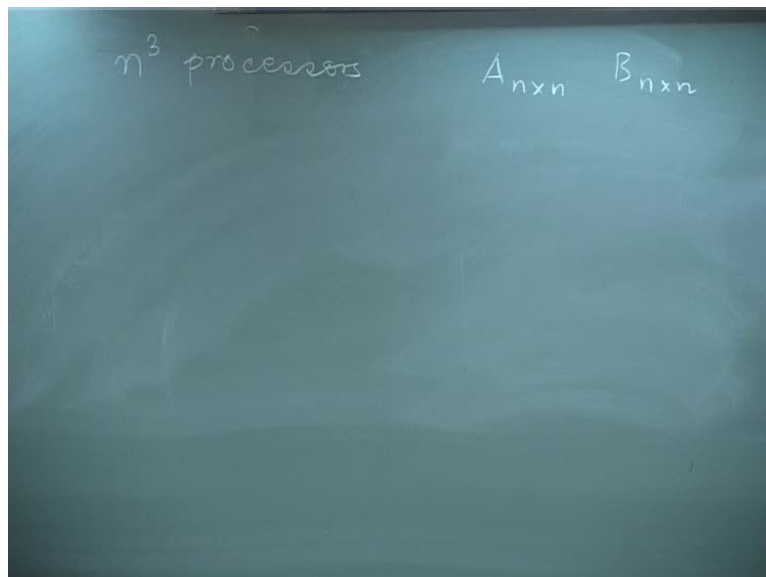
it has generated all 16 numbers that is in this the numbers are lying between 1 to 4, right. That is the thing, this is of what? All the numbers are lying between 1 to 4, so 16 numbers if I tell that they are lying between 1 to 4, then that means partial member of that family only; it cannot be the outside the link, right.

So, now everything is clear only except of this two fault; you have to show that whatever difference three occurred here they are not here. Now this you can easily show that here the first index is one, but here the difference is three where first index is not one, right. So, similarly is the other case also we can show. So, this is the flow of this. Now you have to write down the algorithms for n cross n matrix, right, and see that if we give the correct output; any questions at any stage here? So, what should be the complexity of the algorithm? Order n cube, complexity of the algorithm time of it

Student: n rotators are there; each rotator takes order and there are order and additions in each.

Rotation of why you take order n ? Each rotation only one movement, p i this will move the data this will move that simultaneously, order n square so it is not cost optimal. You have n square plus of order n square time complexity.

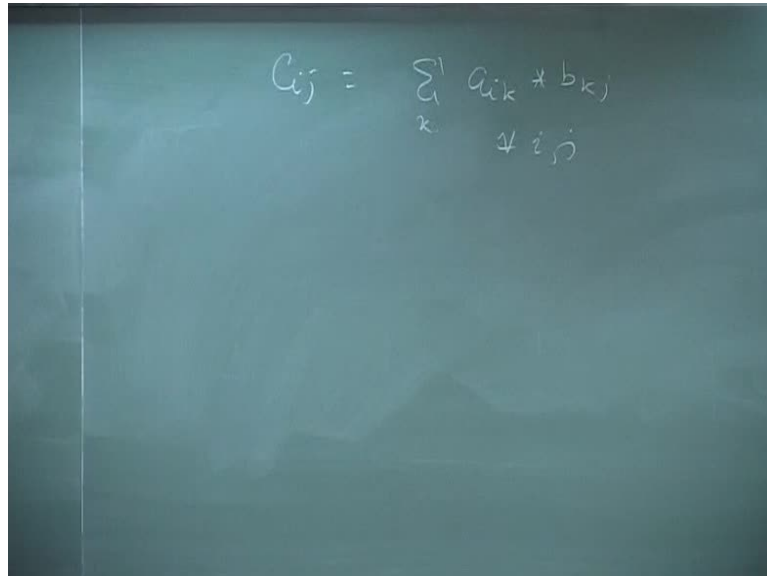
(Refer Slide Time: 20:17)



Now let us think about that hypercube, right, you would remember hyper cube where n cube processors and you have the matrix A and B . So, you observe that basically if I

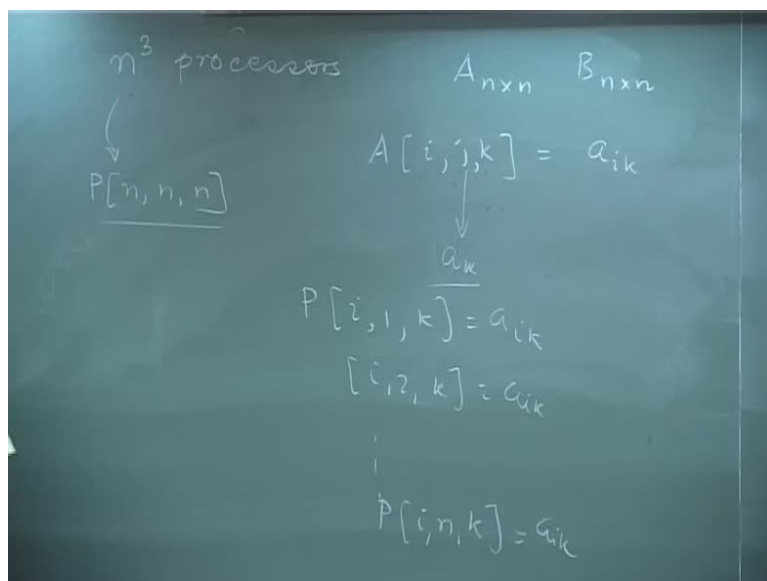
break the matrix multiplication theory, what it means that you need n cube multiplication, right.

(Refer Slide Time: 21:01)


$$C_{ij} = \sum_k a_{ik} * b_{kj} \neq i, j$$

If I see that it is like that C_{ij} is equal to summation over $a_{ik} * b_{kj}$ and over k for all i and j , this is the thing and that is n^3 multiplication in that classical algorithms you need. Now what it did in the hypercube a very simple idea. He has broadcasted a over n^3 matrix over n^3 hyper cube that is a_{ik} has been broadcasted to all j .

(Refer Slide Time: 21:50)


$$n^3 \text{ processors} \quad A_{n \times n} \quad B_{n \times n}$$
$$P[n, n, n]$$
$$A[i, j, k] = a_{ik}$$
$$a_{ik}$$
$$P[i, 1, k] = a_{ik}$$
$$P[i, 2, k] = a_{ik}$$
$$\vdots$$
$$P[i, n, k] = a_{ik}$$

That is instead of A_{ik} you assume that, right. This n cube processor I can write in this form $n \times n$ right, n cube processor I can write $p \times n, n, n$, right. For all i, k , I have broadcasted that, is it possible, it is clear. That every j you have a i, k , every j you have a i, k ; that is $p_{i,1,k}$ is your $a_{i,k}$, $p_{i,2,k}$ is also a i, k and so on. $p_{i,n,k}$ is also a i, k by some means, how we will discuss later.

(Refer Slide Time: 23:10)

$$C_{ij} = \sum_k a_{ik} * b_{kj} \quad \forall i,j$$

$$B[i,j,k] = b_{kj}$$

$$P[1,j,k] = b_{kj}$$

$$P[2,j,k] = b_{kj}$$

$$C_{ij} = A[i,j,k] * B[i,j,k]$$

Similarly we define $B_{i,j,k}$ is your b_{kj} ; that is $p_{1,j,k}$ is your b_{kj} ; $p_{2,j,k}$ is equals to b_{kj} and so on. Now, you observe that $p_{i,j,k}$ for all k they contain the element of c_{ij} ; c_{ij} is nothing but $A_{i,j,k} * B_{i,j,k}$; this is it, is it okay, because $A_{i,j,k}$ contains a i, k , $b_{i,j,k}$ contains b_{kj} so they are multipliable.

(Refer Slide Time: 24:52)

The image shows a chalkboard with the following handwritten equations:

$$C_{ij} = \sum_k a_{ik} * b_{kj}$$
$$A[i, j, k] = b_{kj}$$
$$A[i, j, k] = b_{kj}$$
$$A[i, j, k] = b_{kj}$$
$$C_{ij} = A[i, j, k] * B[i, j, k] \Rightarrow P[C, j, k]$$

If I multiply it for different k I can get in the process of p i, j, k. The processor p i, j, k multiplies the content of A i, j, k with B i, j, k and stores it in C i, j, k, right. So C i, j, k for all k they are the element of c i j of the matrix multiplications, right.

(Refer Slide Time: 25:35)

The image shows a chalkboard with the following handwritten equations:

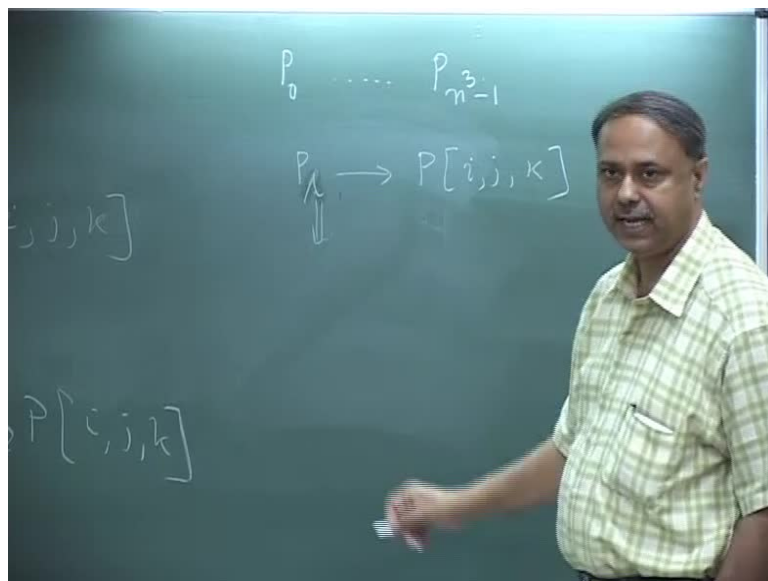
$$C_{ij} = \sum_k a_{ik} * b_{kj}$$
$$A[i, j, k] = b_{kj}$$
$$A[i, j, k] = b_{kj}$$
$$A[i, j, k] = b_{kj}$$
$$C_{ij} = A[i, j, k] * B[i, j, k] \Rightarrow P[C, j, k]$$
$$\sum_k C[i, j, k]$$

Now the next step is that this C i, j, k for all k you take the summation that will be your C i, j, k. So, where is the problem? Some of your face has become little pale, what is the problem, tell me? Let us start again. I have a hypercube of n cube processors, these n

cube processors are arranged in the form of three-dimensional array; each dimension is having size n . So, this process can be expressed in the form of $p \times n \times n$, agreed.

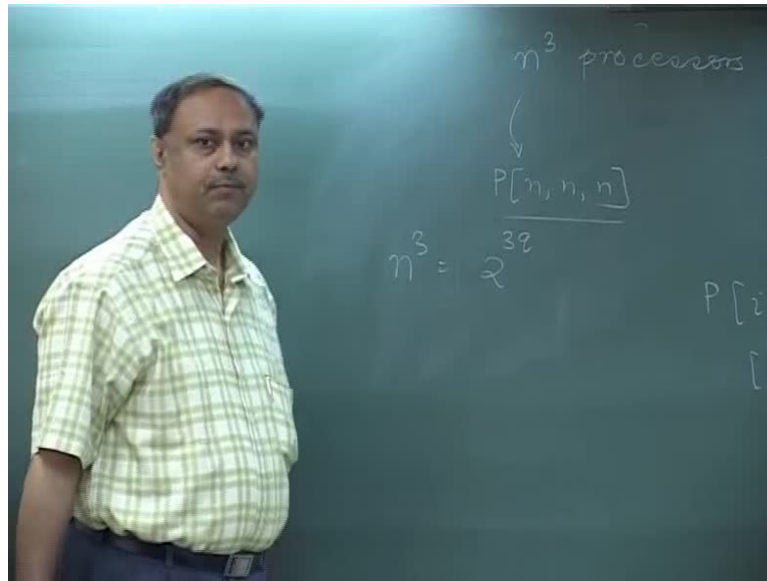
Now a_{ij} is the ij th element of A and b_{ij} is the ij th element of the matrix B . Now what I am defining the processor $P_{i,j,k}$ contains the three elements $A_{i,j,k}$, $B_{i,j,k}$ and $C_{i,j,k}$ is the final element after multiplication. So, $A_{i,j,k}$ for all values of j it contains a_{ik} . How we will be discussing that is implementation details. So, a_{ik} has been said for all $A_{i,j,k}$ or $P_{i,j,k}$ are different j . Similarly, $P_{i,j,k}$ for all i there is a $B_{i,j,k}$ context b_{kj} , right. Then the processor $P_{i,j,k}$ multiplies its $P_{i,j,k}$ with $B_{i,j,k}$, stores a result in $C_{i,j,k}$, then all the processors having the different k , but the same ij they are adding together and you get the C_{ij} ; that is the C_{ij} is the ij th element of the multiplication, okay. So, now the problem is clear. Now let us think how we can do it.

(Refer Slide Time: 28:33)



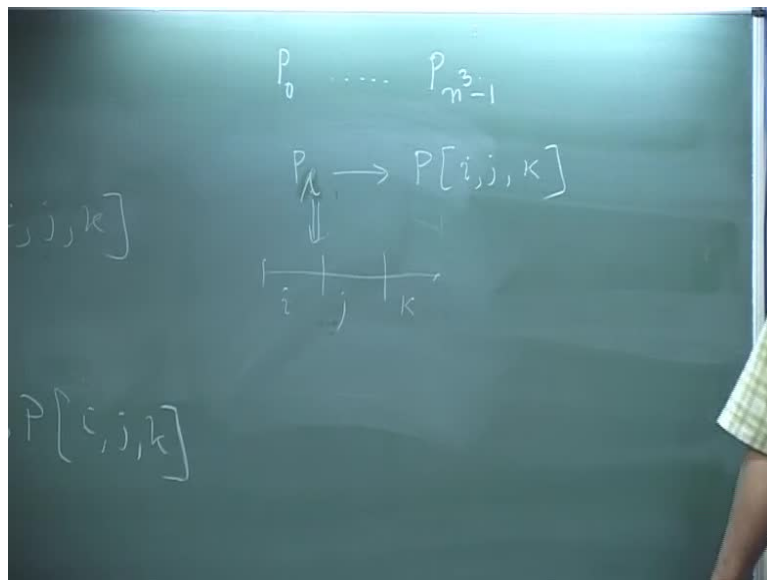
Now since it is an n cube processor, so the processors can be defined as p_0 to p_{n^3-1} , right. Now this index p_i you are writing in the form of $P_{i,j,k}$, instead of p_i you put something else p_l , right. Now this index p_l you are combining into this, so this l I can always write in the form of a three cube base, because you know that hypercube is always only in the form of 2 to the power cube.

(Refer Slide Time: 29:26)



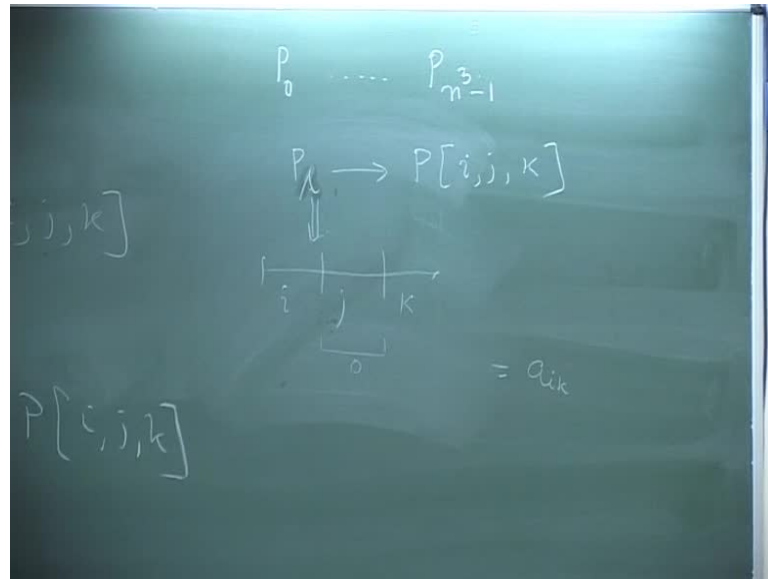
So, this n cube we can write 2 to the power 3 q form, right, 2 to the power of 3 q form; that means this I must have 3 q base.

(Refer Slide Time: 29:45)



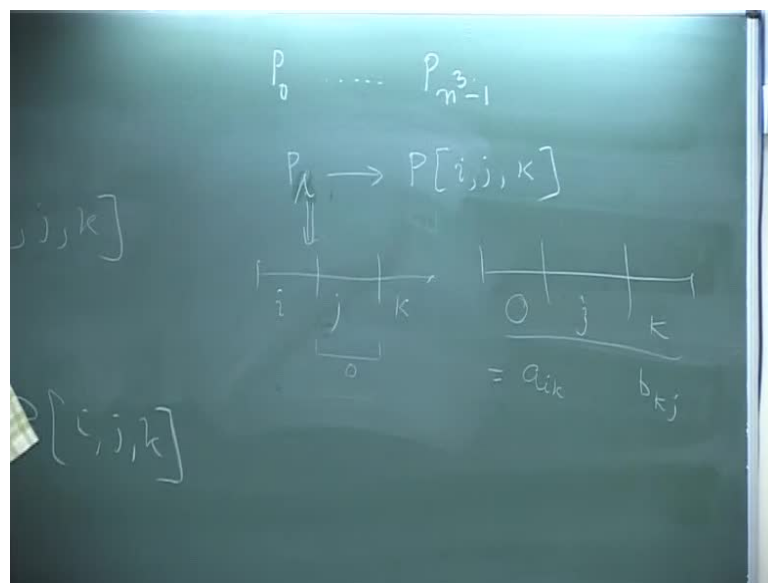
This p I must have 3 q base, right. So, first these are the things, right, and also in the hypercube the processor is connected. So, if I put this is i part, this is j part, this is i can put k part, right. First cube meets this for i, second cube it is for j and third cube it is for k that I can do. Now what I told that initially a i k has to be stored somewhere.

(Refer Slide Time: 30:38)



Initially I put a i k this bit is zero, this bit is zero contains zero, and initially I can put here a i k. This j is zero, right, initially j is zero where I have put a i k. Now after q operations I can broadcast to to all j because they are connected, right. For all j i can broadcast a i k so that A i, j, k contains a i k. Yes, next part is now you have broadcast this part is done, this part is done. Now you are here, how to broadcast this b k j?

(Refer Slide Time: 31:34)



Now initially this is 0 and here j and here k; initially you have put in this position you have put your b k j. You remember b k j is the transpose of B actually you are providing

here b_{kj} , right. Now you broadcast it to all I ; you need q steps to broadcast, agreed. So, b_{kj} has been broadcasted. Now perform the multiplication, which takes cost and amount of time one unit of time. Now you have to perform this one. This is again summation of over q summation over q so you need the q of operations to collapse it, right. In hypercube the question of collapsing is coming. So, this summation you need only q operations to collapse it, agreed.

So, whole thing can be done in q steps, right. If you have n cube processor then $\log n$ step is sufficient to give you the matrix multiplication. So, cost is $n^3 \log n$. So, any query in this? Here you observe that matrix multiplication has been cleverly he has used it for hypercube. If the discussion is relieved the number of processor is less than n^3 . In that case you may find the difficulty to achieve this.

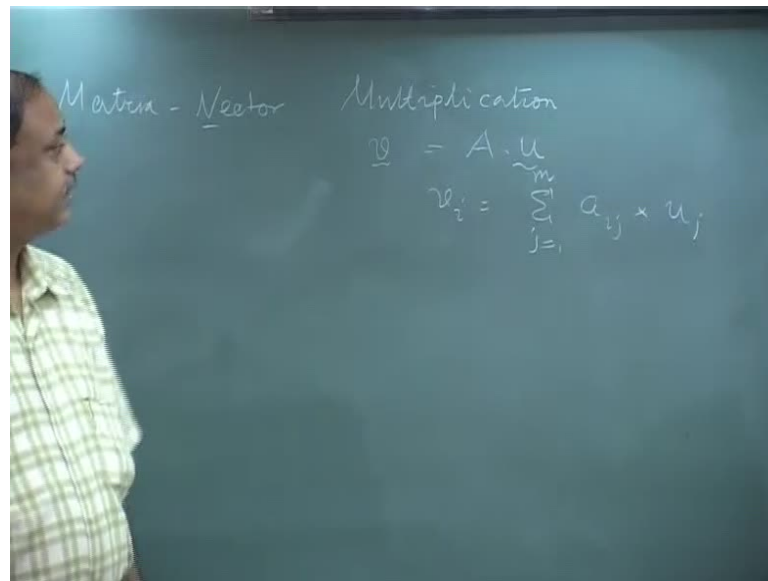
Student: For scientific computations we will leave additional large.

Oh, say let it be clear to you. While we explain the parallel algorithms we always assume that the number of processors is a function of the number of input parameters. Now yes in reality you will be telling sir it is not possible; yes, it is not possible, because we cannot take, there may be infinite number of processors that n^3 is a very large number for $n \times n$ square matrix. But if you tend to design in your algorithm it is good; it is good for the people to use it. There are different ways we can solve that you assume first partitioning the matrix and partitioning it in such a way that your n^3 that is falling in that criteria and after that again it can be combined. So, that is why you know that finding the solver if it is a good solver it is a very difficult problem and then Ishfaq Ahmad and Anshul Gupta they are working last 30 years only on this.

See for them they do not want; their idea is not the cube connectors. Their idea is that even if it is a shared memory one, can we have a large matrix solver. Now once you define that matrix that matrix of difference type; one may be you can define that it is a finite matrix. Finite matrix means these are the final field; that is these elements are integers modulo $p-1$ vertices prime or modulo p vertices prime, but the interior element may be very large. Not only the matrix size is large, the content element itself may be 50 digit number, right. This is one type of problem. Another type of problem is that no it is not real domain, the number is on real field, matrix is very large. Another worry is then no matrix is a sparse matrix. All matrixes are symmetric matrix symmetric

positive matrix. Every one of them is approved but do not ask what is the size of the matrix? The size of the matrix may be very, very large that 50,000 cross 50,000 matrix you have to cross, right, and you know they do not have a real data whose stressor values are also available, right. This is a difficult processor. So whatever those you would be giving they would be thinking yeah this is the correct one.

(Refer Slide Time: 37:41)



The next type of problem is matrix vector multiplication, right, that is also here the problem is instead of the two matrices you will be performing $A \cdot u$ and you will be getting the vector v , right. You remember that, right, what is that? v_i is equals to summation over what is that? Matrix vector on, u

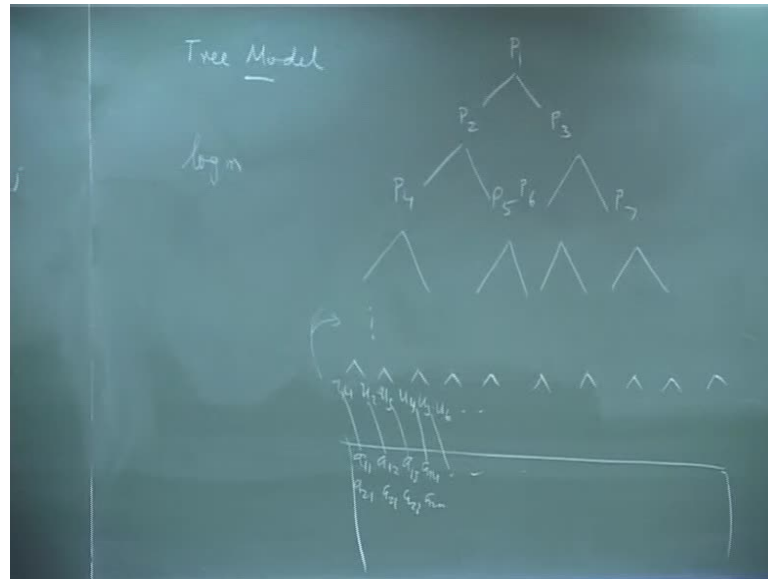
Student: $j = 1$ sir.

$j = 1$ means vector is having only one. Now can you suggest a method tell me. Suppose j is

Student: Sir we can make each of these a_{ij} if you take each row, these rows you can connect them to like a tree, like a_{ij} will be connected to a_{i2j} and a_{i2j+1} .

Yes yes, that is the tree model. Yes, let us discuss this one, very good.

(Refer Slide Time: 39:25)



This is known as the tree model. So, here the leaf node contains all u 's u_1 in the first leaf node, u_2 in the second leaf node, u_3 in the third leaf node and so on, and you are inputting the elements of a row wise. So, here you have a 1_1 1_2 1_3 1_4 and so on, right. Here you have a 2_1 2_2 2_3 2_4 and so on. So, a 1 is inputted here, a 1_2 like this, right. At the first instant of time first row has been inputted and multiplied, right,

Student: It can be passed on to the next on.

Yes, resultant could be passed in to the higher while he is passing the result down the second row will come over here, right, and once the higher node receives the two data here and the result is sent to higher node and so on. So, it propagates from the bottom to up. So, after ex-iteration the d_1 will be out and x is nothing but the height of the k which is $\log n$. So, you need which is $\log m$ which is n , this is $\log m$. So, after $\log m$ iterations first hazard will be out. So, can you tell me how much time you need to complete this multi vector multiplication?

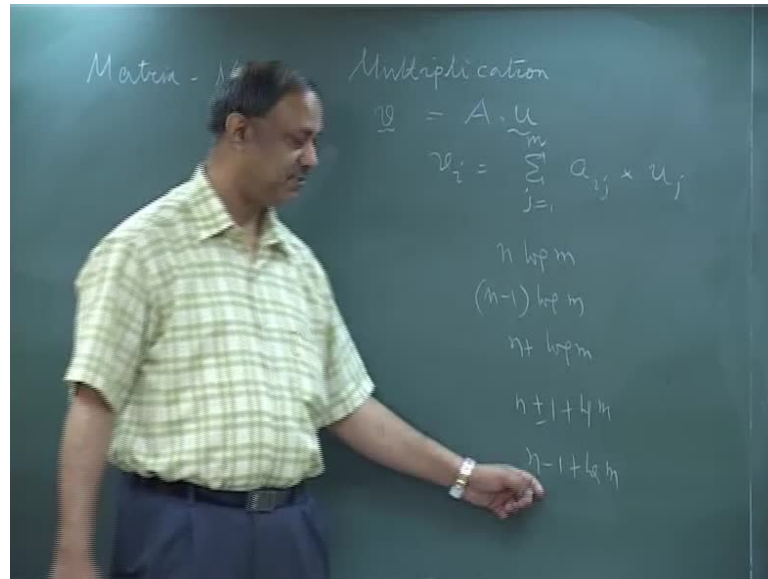
Student: $n \log n$.

$N \log n$, n minus $1 \log n$

Student: n plus $\log n$.

n plus $\log n$.

(Refer Slide Time: 42:35)



So you have given $n \log m$, you have given me n minus $1 \log m$; you have given n plus $1 \log m$.

Student: median factor of plus minus 1 in the last row.

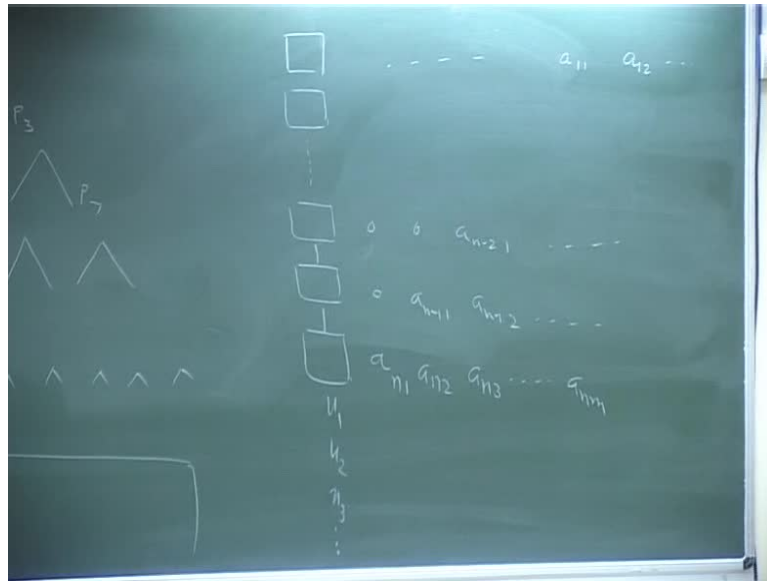
Factor of plus minus 1 n plus minus 1 plus $\log m$

Student: n minus 1 plus $\log m$.

n minus 1 plus $\log m$, so from where we are angrier? So, see the first one first row will be coming up after $\log m$ and next one onwards it is coming one by one, right. So, you need n minus 1 that type of thing, so n minus 1 plus $\log m$, right, to generate all the elements of it. Now in this case you observe that this type of problem is very easy to solve one tree model.

Now can you define a problem of similar type of things which can be used efficiently the pyramid model, just you do not want to think now because if it is so easy then I could have not taken. So, you can think to find out some problems which can be used for pyramid model, right, can be used on pyramid model. Now in the case of linear array, can you use or can you give me an array we used to solve this problem and a linear array to solve. So, you have idea? No, prepare the idea.

(Refer Slide Time: 45:13)

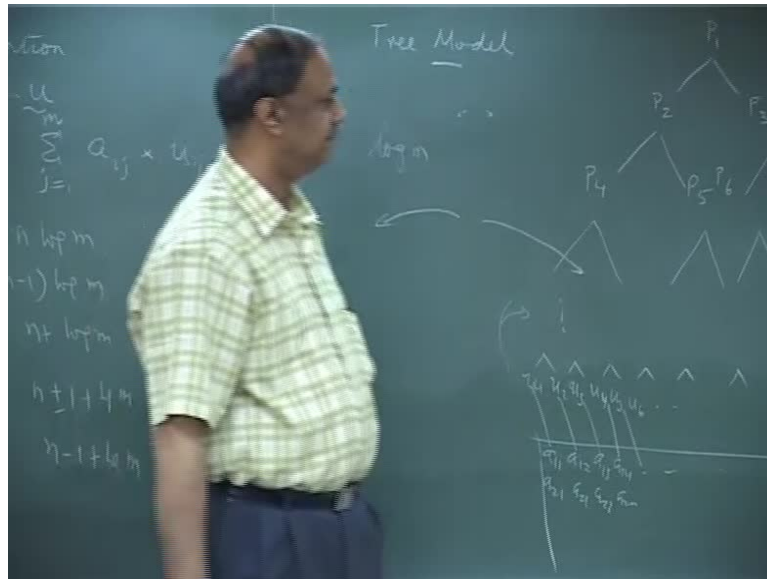


So, you have the linear array. You have here $u_1 u_2 u_3$, you have here say $a_{n1} a_{n2} a_{n3} a_{nn}$, $a_{n-11} a_{n-12}$, right. So, if I design this way and propagate the data this way, so first time this two are multipliable. You get the b_1 , b_1 it comes here that time u_2 and n_2 be there, b_2 has been formed. So, b_1 moves up, b_2 comes here and at that time second part of b_2 will be coming and to add it and so on, agreed, yes no, no yes, no, no or yes.

Student: No sir.

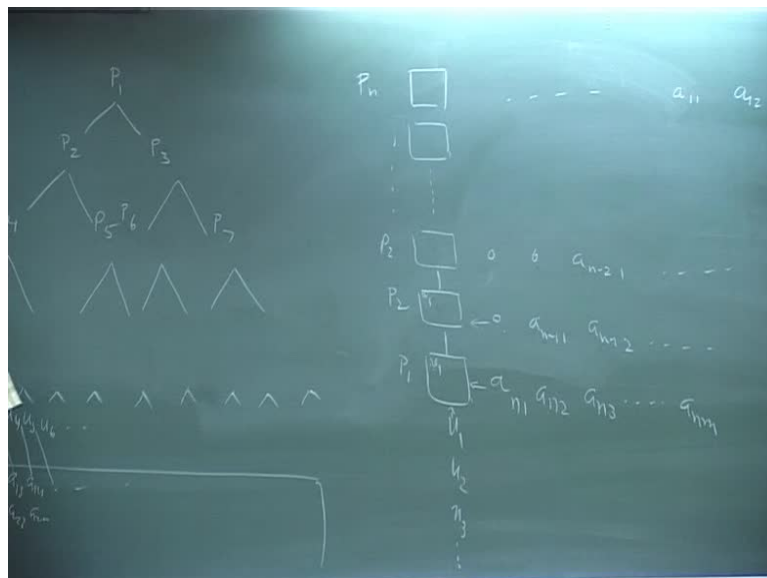
So, tell no. See you are tilting the head.

(Refer Slide Time: 47:06)



So, that does give me anything, either you tilt this way; see rotating this indicate something, that I will ask some question. But if we do like this it is less than paper value. If you rotate this way I will ask you whether you are in India as south Indian or north Indian, so based on that I will decide.

(Refer Slide Time: 47:43)



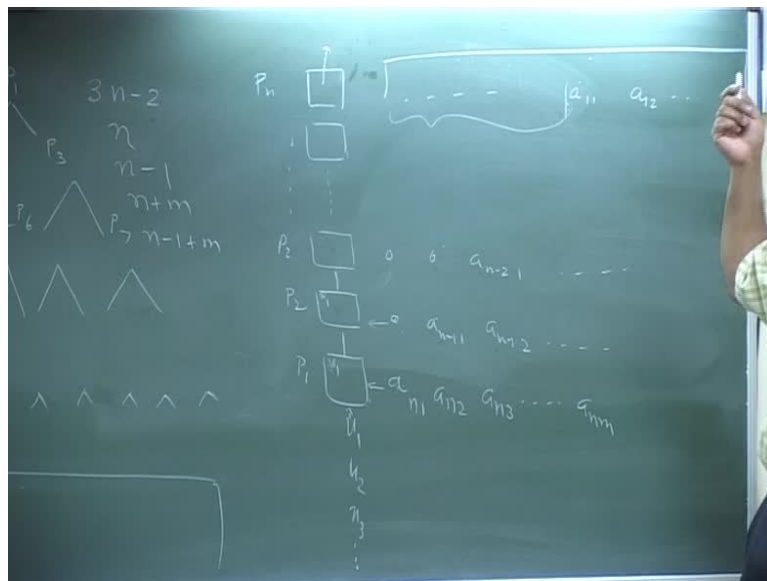
Now here it is a linear added. This is the processor p 1 processor p 2 p 3 and so on, right, and here it is height so it is p n. Now u 1 while I am putting counting here a n also is formed into this and they combine or multiply and the data is stored here as b 1, right.

Now b_1 moves in the second iteration and in the first iteration while it is coming here this has come here, this is come here, this data has come here and so on. The second iteration b_1 has been moved on and this element has come here, this element has come here; you can multiply and add with b_1 .

At the same time this element has been moved here, this element has been moved here, this element has been moved and you can multiply for and store it in b_2 . Third iteration b_1 has been moved here, b_2 has been moved here, u_3 and a_{n3} has been moved here to compute b_3 and so on, agreed. So finally, after x iteration you will be generating the matrix vector multiplication result, what is the value of x ?

Student: $3n - 2$.

(Refer Slide Time: 49:30)



$3n - 2$, this is the first answer I got. After n iterations we will be getting the final results, remove $n - 1$; after $n - 1$ you will be getting the final result.

Student: $n + m$.

$n + m$.

Student: Sir $n - 1$ and m .

$n - 1 + m$, you have to find out what is the distance from here to here; that is the only thing you need to know, right. See last element has to add here and to move it out.

Student: n minus m minus 1, n plus m .

n plus m or n plus m minus 1. This element is here, how many blanks? Second row one blank, third row two blanks, n th row n minus 1 plus how many elements are there? M . So, it is n minus 1 plus m , agreed or not.

Student: Yes sir.

Now if you think no no I have to pump this out also, then one addition.

Student: Coming back to this point, m and n .

Where are you?

Student: In the previous one till model.

So, like which one it is?

Student: Are not n and m equal?

Why, need not be.

Student: Are not they of same order?

If you take the same order then it is order n .

Student: Yes yes.

Then it is order n . See I have taken this thing because in the vector size is not known, right, they may be different.

Student: I can take any tree size and.

Tree size is dependent on the vector u .

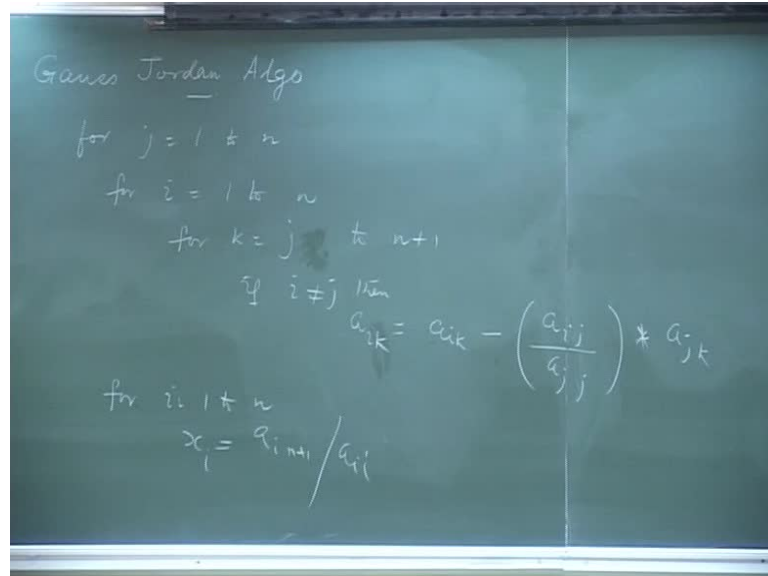
Student: Then it will be moving.

Then it will be?

Student: Will it not?

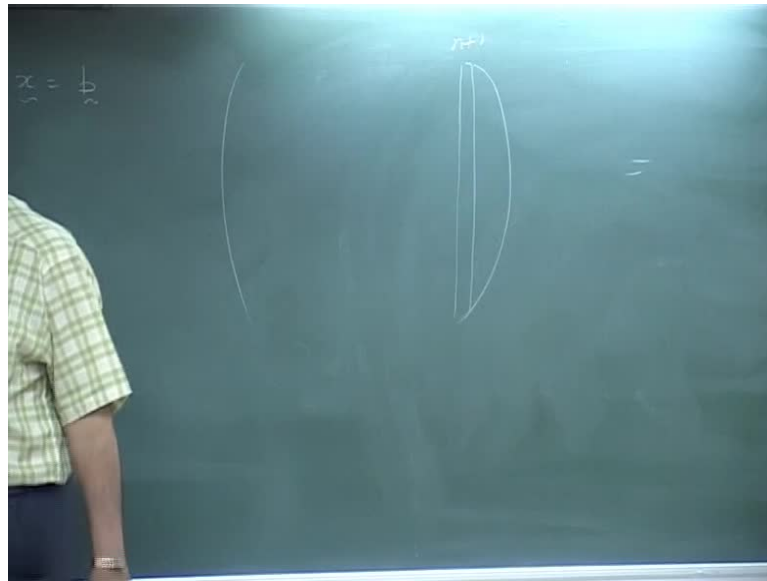
Height $\log n$ but if you use i's, use i's is n height will be $\log n$, yes. No dispute in that I am also agreeing to that. Now if you have the number of rows is m then it is n minus 1 plus $\log n$.

(Refer Slide Time: 52:25)



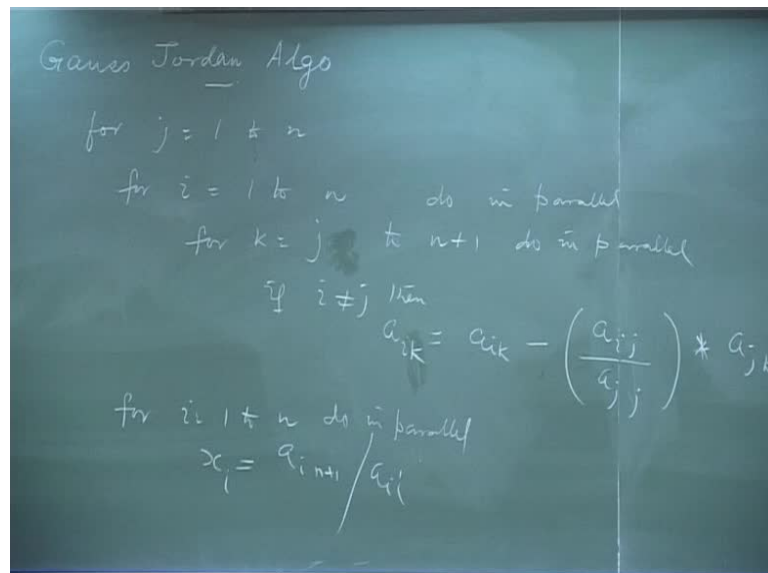
Now, the first finding the solution of simultaneous linear equations, but I also use algorithm; you have for j is equal to 1 to n for i is equal to 1 to n for k is equals to that is the thing I forgot j plus 1 to n plus let me right down the sequential algorithm. Then we will discuss. If i is not equals to j then a_{jk} is equals to a_{ik} minus, then we have for i equals to 1 to n , x_i is equals to a_i , very, very tight because operating the inverses are there in vectors. So, Gauss Jordan algorithm tells what? He is diagnosing the thing and they are finding the value. Now what is n plus 1 because I told you it is n cross n , n plus 1 column is nothing but the b has been moved to n plus 1, that is the only thing.

(Refer Slide Time: 54:36)



You have A matrix and then you have B vector; instead of putting B here I have put B here which is n plus 1 at the column, okay. And this is a standard formula now nothing to this I have told. See these algorithms can easily be implemented on parallel machine if you have order n square processor.

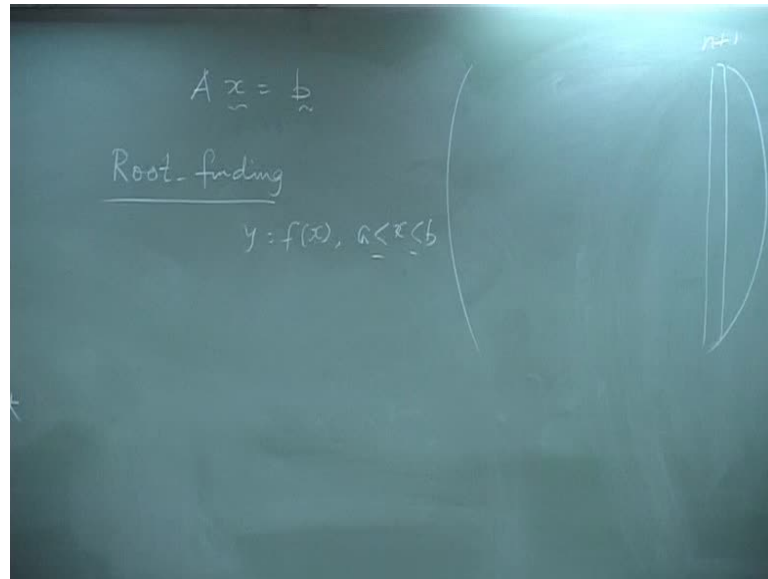
(Refer Slide Time: 55:20)



See just you observe that if I write here do in parallel do in parallel, because of all this things are dependent on j only. So, the j i cannot write do in parallel, right, agreed. Rest i can easily write. Similarly here also I can write do in parallel. So, this is an order n

algorithm basically, can you tell me why i cannot be equals to j ? If i equals to j then this becomes zero, right. So, next retardation you must be careful. So, basically you are doing non-biotic element. So, for this A and B you can use it on shared memory model. See one thing you remember numerical methods problem generally we work on shared memory model but people are working nowadays on cluster machines.

(Refer Slide Time: 56:52)



Now the next problem is that you have to tell me how I can do it root-finding problem, right. What is the answer? y equals to $f(x)$ and this and you need to find out the root of that row, find out the roots of this function, right. What is root? What value of x ?

Student: Y is zero.

Y is zero, fine, this is known, right. For what value of x y is zero. Now what is the method tell me.

Student: Newton method.

Newton method and?

Student: Bijection.

Bijection method, which is the simplest one?

Student: Bijection is the simple one.

What is the problem in that?

Student: a plus b by 2.

Now what is the b value I did not ask that, what is the b value?

Student: It takes more time then.

Root is?

Student: Not in middle. Each generation reduces it by half.

What is the b value?

Student: If the root is in one side

If root is in one side means?

Student: No, one side means near the B or near A it takes so much time because you are doing half back half.

So, log n time

Student: Sir finding the initial values, initial guess.

Initial guess, there is no initial guess here.

Student: Sir f x.

F x, no no, that ways method is simple. Method is that you find out the middle one and you look for whether it is possible; that method I do not want to tell now. I want to how much you could recollect from your school days; school means that undergraduate days.

Student: Yeah check for the sign at the middle point.

i at the middle point, f x will not give you any sign.

Student: Then how are you finding the root?

You tell me.

Student: Here we are finding the root and the point where $f(x)$ is zero. On either side of $f(x)$ you have a negative and a positive.

See you have a ground like this say. Now this is negative, this is negative that is what you are going through; it does not give anything. $f(x)$ when it is negative, middle value is also negative.

Student: The root is that it intersects with the x axis; does it not intersect at the y axis?

But any function if i , okay now you tell me. Now you middle one is here. Now you tell me. You are telling what? You will be finding the middle value f of middle, right, that is the thing you told. Then you got this middle value which is negative. What it means?

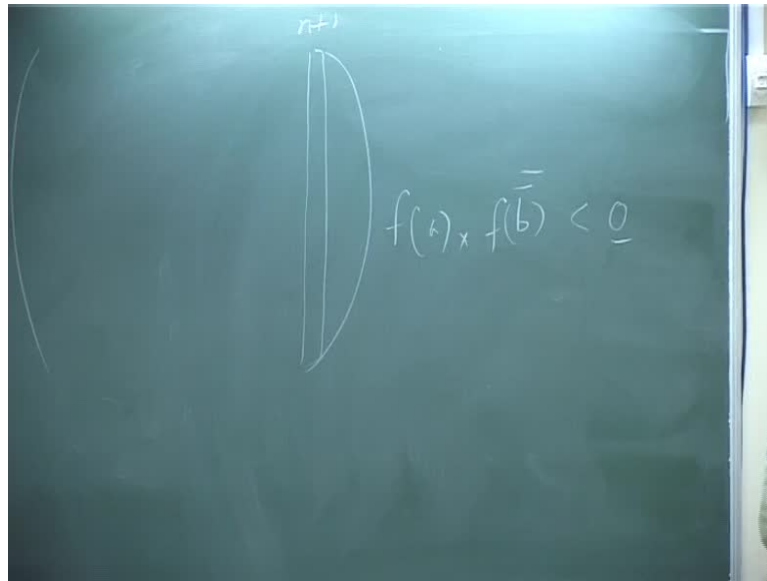
Student: Root belongs to the middle and at b point.

Then tell me what is the d value I am asking.

Student: There are two roots and of those sign interchange.

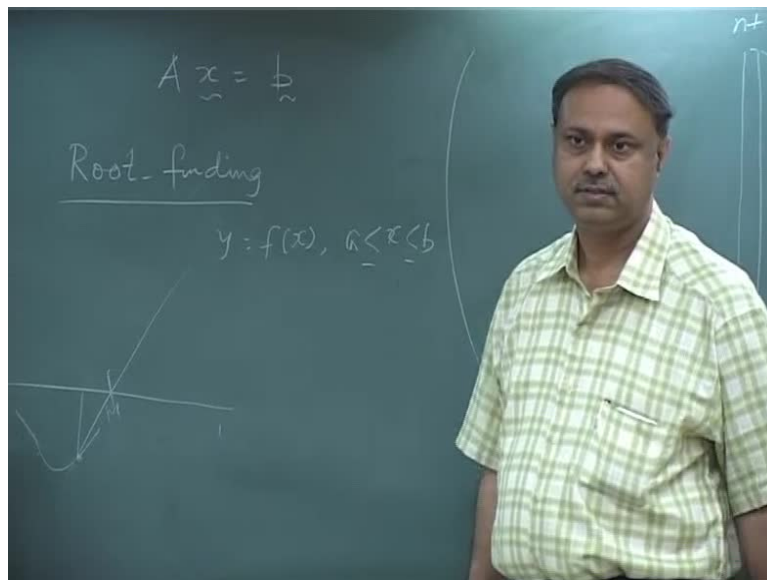
Yes, that bisection method fails if you have even number of roots. So, I assume that there is no even number of roots, right. There exists at least one or three or five that many numbers, what it means? It means that $f(a)$ into $f(b)$ must be negative or $f(a)$ and $f(b)$ they are the two different signs; that is the assumption we are making. That means that exist; that means that it has odd number of roots, right. It does not make the complete real root or complex root. I never told that they are all complex.

(Refer Slide Time: 1:01:58)



See now the scenario is that you have $f(a) \times f(b)$ is less than zero; that assumption you have taken.

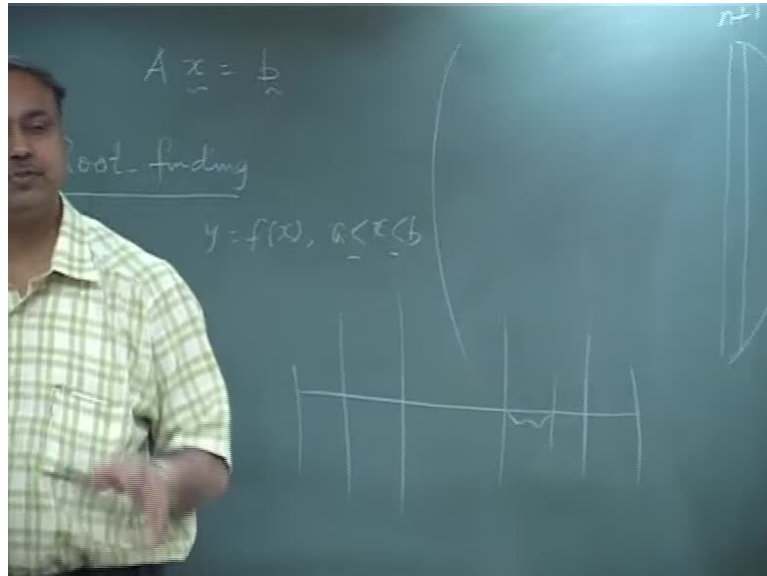
(Refer Slide Time: 62:15)



Now procedure is that you take the middle value and then you multiply this with this one; if you find we are having this at zero then the root is lying this side. So, you will be starting from M and B this becomes your A and again you divide and so on till you get the root or the distance between this two is very small, right, range is very small because

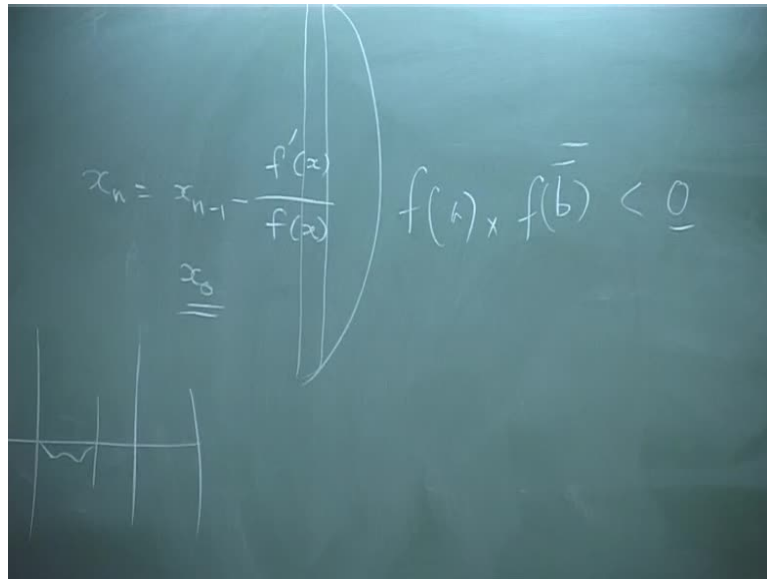
after that you will find that it is not convergent. Now in this algorithm can easily be implemented on parallel machine even though you feel that it is sequential in nature.

(Refer Slide Time: 01:03:05)



The simple thing is that rate you have a and b and suppose you have n processors. So, you divide into groups, right, and you assign one group to one processor and it finds that you need to zone utilize. Suppose you find that this is the zone where you get the f a and f b after multiplication you get the negative sign, then this is the zone. Now you divide this zone again into n groups and push it, right. So, root finding even though you feel that it is very complex very sequential in nature, but writing a parallel algorithm for root finding is not a problem. Now you think about the Newton-Raphson method. Newton-Raphson method I forgot the formula; who can help me in writing the formula?

(Refer Slide Time: 1:04:04)



x_n 's is equals to.

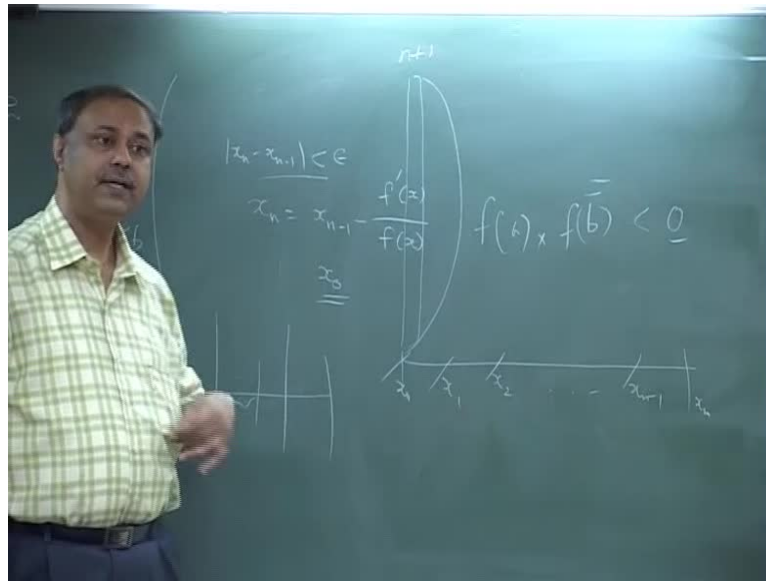
Student: x_n minus 1 minus f' of x by f of x .

Over?

Student: Yes, this one is correct, that is it.

Correct, good and you need something $\times 0$; that is the thing he has hid values here and Newton-Raphson method is selecting the seed value important. Suppose you have very good seed value and other things here also a function is always defined between a and b , right.

(Refer Slide Time: 1:05:02)



So, instead of selecting the seed value what you do you have you divide this into n groups, right, x_1, x_2, x_n 's, right. So d and a are divided into n groups, right, where a is x_0 and b is your x_m , right. So, you use this seed value; use this seed value processor p_i use x_{i-1} as the shift value and then you obtain the distance between. See you have to obtain this distance, right, if it is less than epsilon then only we will get, right. But you remember as in the case of sequential algorithms this algorithm also does that DBO the quick or guarantee of convergence.

Student: N different processors are trying to converge at the same point.

No, it will not because its door is this. Every zones I will not allow him to access the other zones, right, then the conflict and other things will be coming into the picture. This zone will be up to this, this zone will be up to this.

Student: So, obviously some of your numerical methods.

I do not know. There are several parallel numerical methods books available.