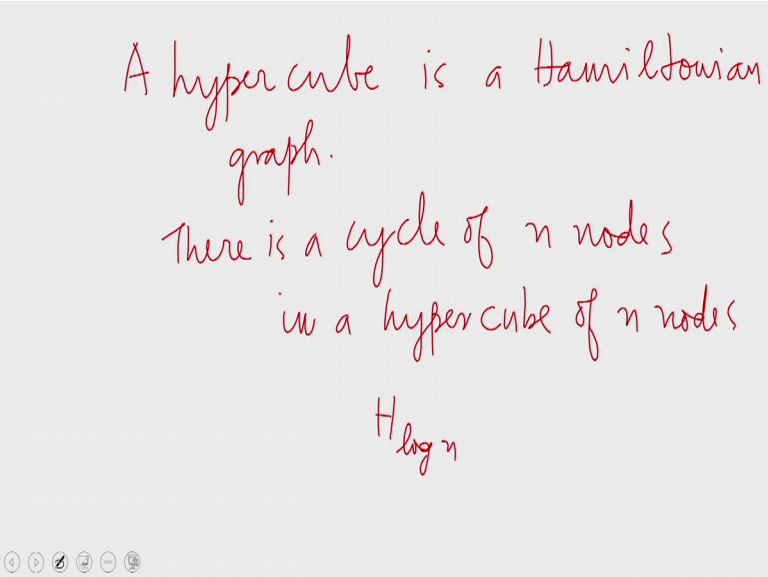
Parallel Algorithms
Prof. Sajith Gopalan
Department of Computer Science & Engineering
Indian Institute of Technology, Guwahati

Lecture - 29
Hypercube Cont'd

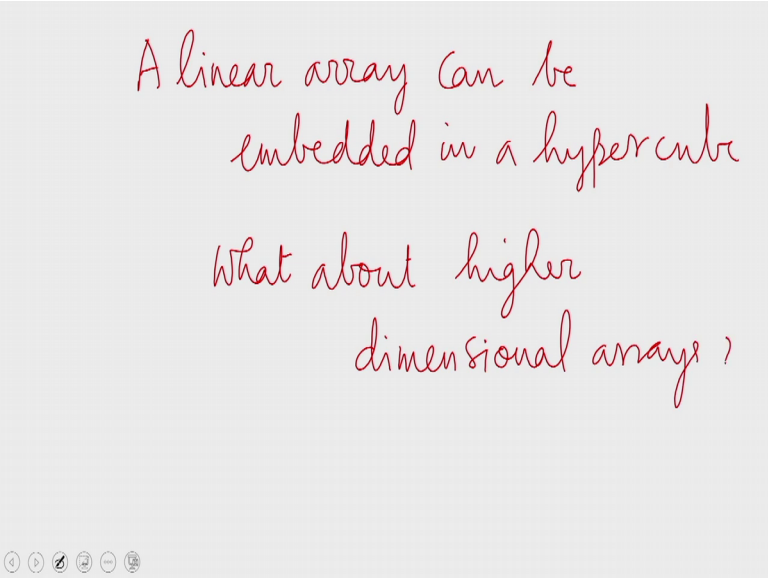
Welcome to the 29th lecture of the MOOC on Parallel Algorithms. In the previous lecture, we had started discussing hyper the Hypercube packet architecture. Today we shall see some properties of the hypercube architecture which establish the established that it is indeed a versatile architecture.

(Refer Slide Time: 00:55)



In the last class we saw that a linear array can be embedded on a hypercube. In the sense that a hypercube, when perceived as a graph is a Hamiltonian graph. What this means is that; there is a cycle of size n, in a hypercube of n nodes. Of course, is the hyper cube of n nodes n is going to be a power of 2 and this hypercube is of dimensionality log n. So, in h log n there is a cycle of size n.

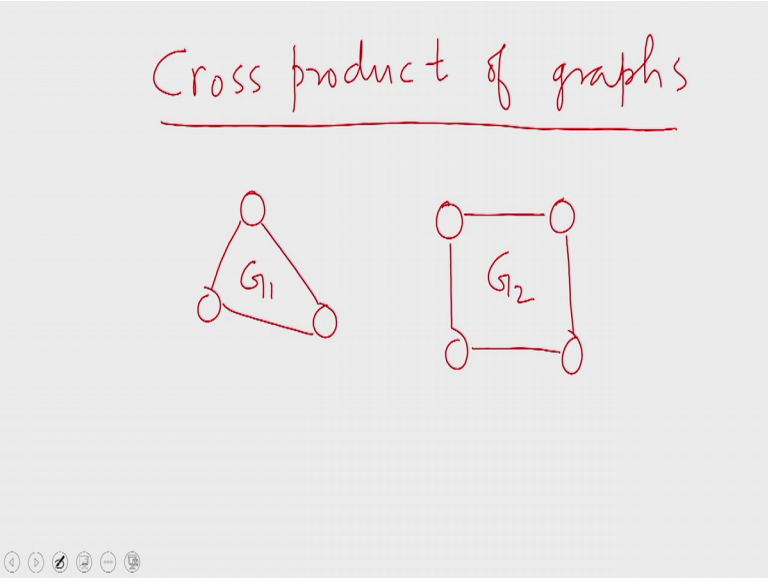
(Refer Slide Time: 01:54)



So, this is the cycle which passes to every vertex exactly once. What this means is that, a linear array can be embedded in a hypercube. That is if you are given an algorithm; that runs on a linear array of n nodes and we are given a hypercube of n nodes. Then this algorithm can be run on the hypercube without any change, that is because, there is a linear array on the hypercube. A linear array is a sub graph of the hypercube therefore, we will activate only the edges that belong to this sub graph.

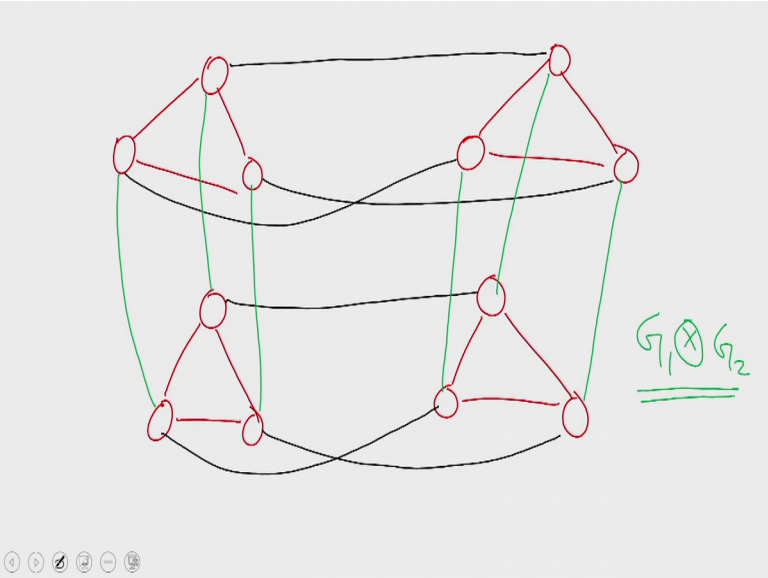
So, these processors will work on the input, the algorithm will be run only on this sub graph only the edges belonging to this sub graph will be used by the algorithm. So, the algorithm will run exactly in the analyzed time that is the time it takes on the simulation, is exactly the same as the some time it takes on the actual run. What about higher dimensional arrays? This is the question we are going to address now.

(Refer Slide Time: 03:21)



For this let us define the notion of a Cross product of graphs. First let me give an example, let us say we are given 2 graphs. This is one graph and this is another graph, this is G1 and this is G 2. To take the cross product of these 2 graphs, what we do is this we take in the second graph there are 4 vertices, in the first graph there are 3 vertices.

(Refer Slide Time: 04:06)



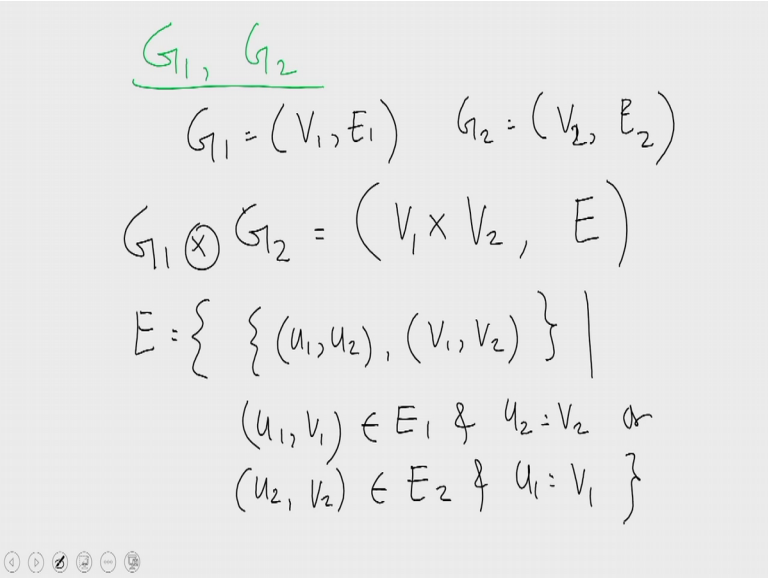
So, we will take 4 copies of the first graph. So, we have these 4 copies of the first graph, these 4 copies correspond to; the 4 vertices of the second graph in the second graph we have 4 vertices. So, the copies of G1 correspond to these vertices. Now we are going to

introduce a bundle of edges in the cross product graph that correspond to the edges of the second graph.

So, for every edge of the second graph we are going to introduce a bundle of edges, that go between the corresponding copies of G1. For example, for the top left to right edge of G2. We will introduce a bunch of bundle of edges between these 2 copies of the top 2 copies of G1, but then we will connect the corresponding vertices, we will connect vertices in this manner. The corresponding vertices are interconnected, between the first top 2 copies of G1. Similarly, the bottom 2 copies of G1 are also interconnected, but these 2 bundles of edges correspond to the 2 left to right edges of the top edge as well as the bottom edge, then the 2 top to bottom edges will also be translated into 2 bundle of edges.

So, this will be between the left copies of G1 and between the right copies of G1. So, let me draw those bundles, using a different colour. So, here between the top left copy and the bottom left copy; we connect the corresponding vertices like this. Similarly here too, a graph obtained in this fashion is called the cross product of G1 and G2, it is denoted in this fashion. Now you can see that you could have obtained this graph by taking 3 copies of the second graph and connecting the corresponding vertices; you find that the graph that you obtain is exactly the same therefore, G1 Cross G2 is identical to G2 Cross G1. So, this is a commutative operation.

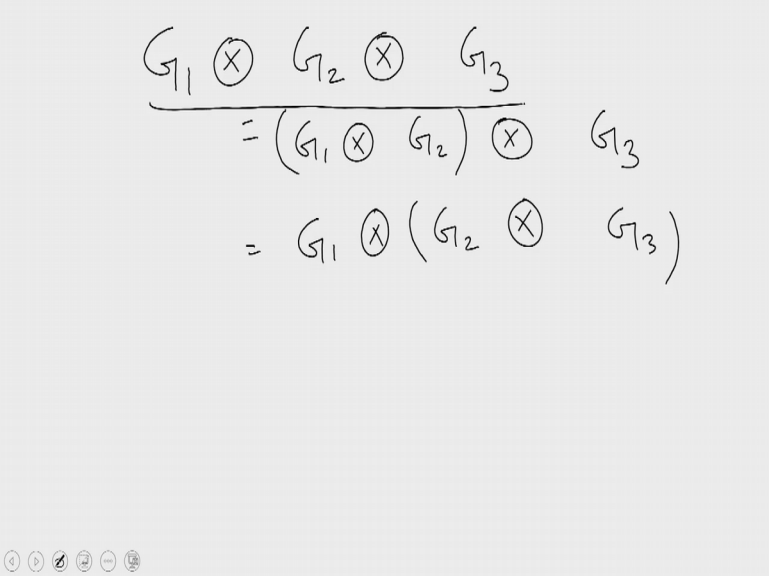
(Refer Slide Time: 06:46)



So, in general for 2 graphs G1 and G2; let us say we are given 2 graphs G1 and G2, where G1 has a vertex set of V1 and h set of E1 and G2 has a vertex set of V1 and then V2 and an edge set of E2. Then we define the cross product of G1 Cross G2 as a graph on V1 cross V2 with edge set E. Where we define E as, the set of all edges of this form the vertices of G1 cross G2 is the cross product of the 2 vertex sets V1 and V2 and then the edge sets are defined like this. From ordered pair u1, u2 to ordered pair V1, V2 we will have an edge if and only if, either u1 is equal to u2; u1 is equal to V1 and u2 V2 is an edge of E2. The second graph or u2 is equal to V2 and u1, V1 is an edge of the first graph, with respect to our example here.

We have 2 graphs G1 and G2, where G1 is a 3 degree vertex and G2 is a 4 cycled. Then for a vertex u1, u2 and V1, V2 in this graph, we will have an edge between them, precisely when u2 is equal to V2. Which means they correspond to the same copy of G1 and then u1 V1 belongs to G1 or u1 is equal to V1, which means they are corresponding vertices of 2,3 cycles and u2 is equal to V2. So, what we get is precisely this graph.

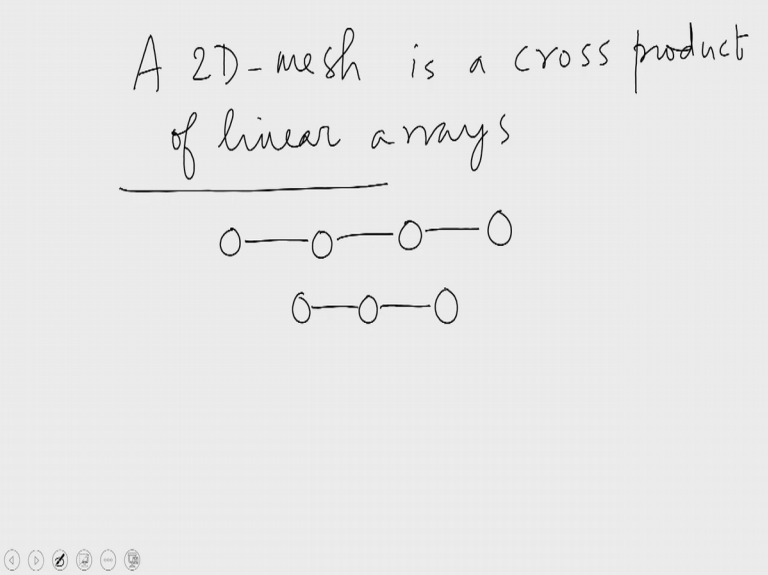
(Refer Slide Time: 09:15)



So, this is the cross product of 2 graphs, when we have given 3 graphs, we define the cross product of the 3 of them as you can readily verify that; this is the same as this. In other words the cross product of graphs is an associative operator. So, this is an operator which is commutative as well as associative therefore, in general we can write the

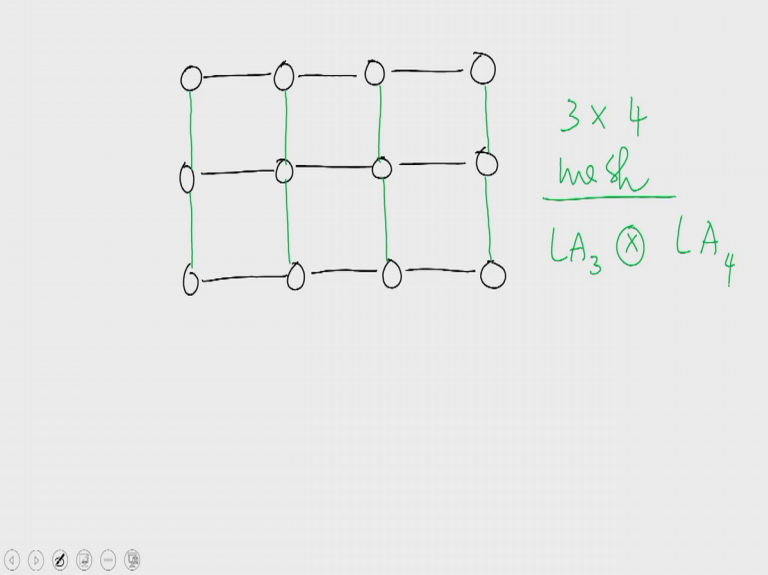
expression G1 cross, G2 cross, G3 without parentheses. So, the order of evaluation is not relevant.

(Refer Slide Time: 10:08)



So, why are we talking about cross products of graphs? This is because a 2D mesh is a cross product of linear arrays, to see this take an example. Let us say we are given 2 linear arrays, 1 with 4 nodes and 1 with 3 nodes.

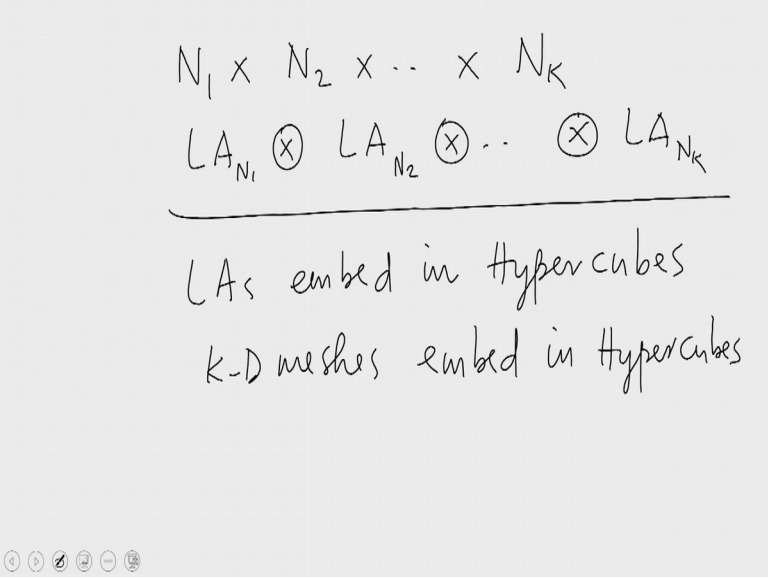
(Refer Slide Time: 10:44)



What if we take the cross product of these, to take the cross product of these we need to take 3 copies of the first graph G1; G1 is a 4 vertex node in this case. And then we can

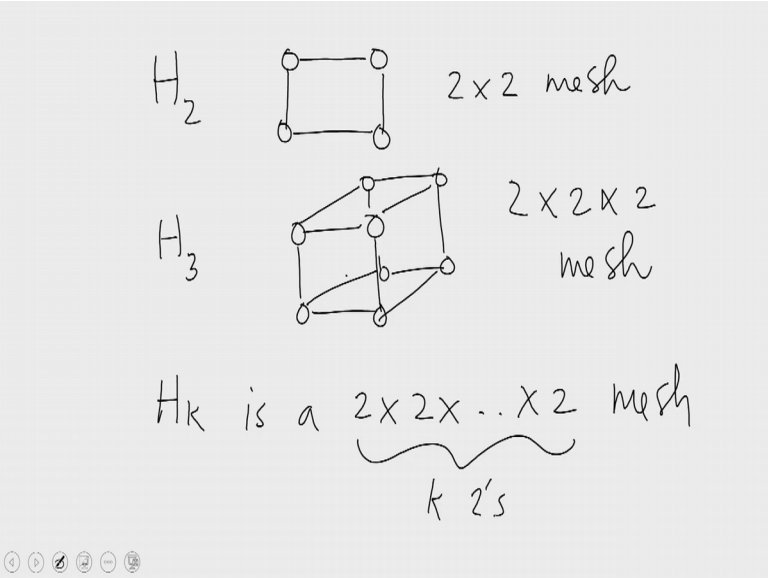
we have to connect the corresponding vertices. So, the word vertices which are aligned vertically or are corresponding vertices, so among these corresponding vertices we have to interconnect exactly as they are interconnected in G1 or other G2. G2 is a linear array of size 3. So, G2 has 3 nodes that are interconnected to form a linear array. So, when we interconnect in this manner, what we get is a 3 by 4 mesh. So, what we find is that a 3 by 4 mesh is a cross product of a linear array of size 3 and a linear array of size 4.

(Refer Slide Time: 11:46)



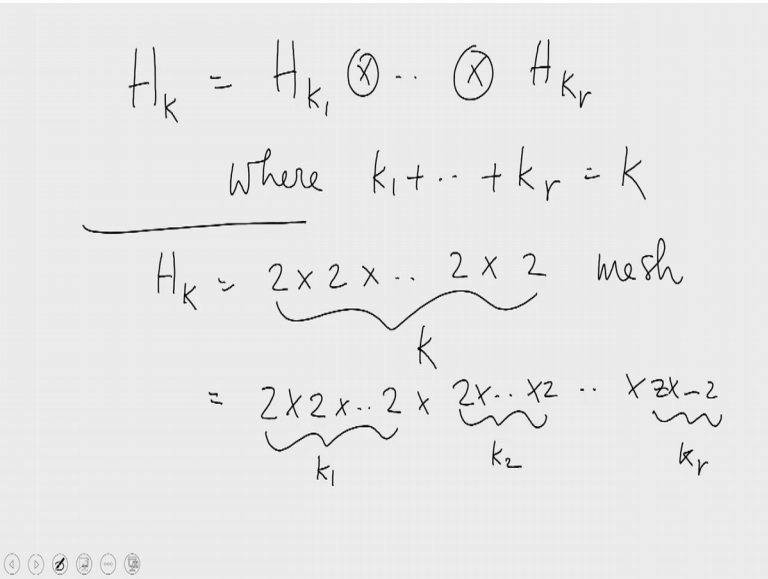
In general for higher dimensions or k dimensional array, with N1, N2 etcetera dimensionalities for the mesh can be expressed as a cross product of a linear array of N1 nodes with a linear array of N2 nodes and so on. So, an N1 by N2 by etcetera Nk mesh is nothing but, the cross product of these linear arrays. Now why are we talking about cross products and what are we gaining by expressing multi dimensional meshes as cross products of linear arrays? What we are going to achieve is this? We know that linear arrays embed in hyper cubes. Therefore, we will be able to show that k dimensional meshes embed in hypercubes, because k dimensional meshes can be expressed as cross products of linear arrays and hypercubes can be expressed as cross products of hypercubes.

(Refer Slide Time: 13:25)



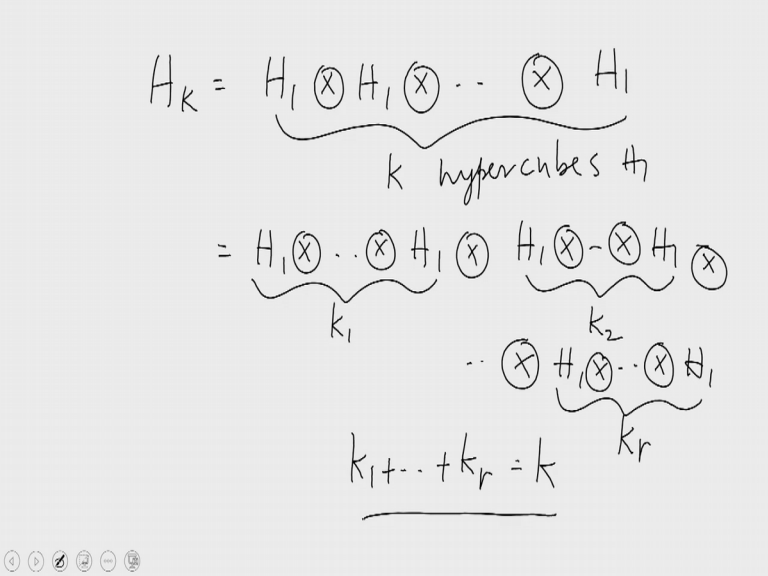
So, that is what we are going to show next. Consider H2 and H2 is obtained by taking 2 copies of H1 and interconnecting them in this manner. So, as you can readily see this is a 2 by 2 mesh, a 2 dimensional hyper cube this is a 2 dimension 2 by 2 mesh. H3 is obtained by taking 2 copies of H2 and interconnecting the corresponding nodes as we know, you can see that this is nothing, but a 2 by 2 by 2 mesh. So, continuing like this we find that Hk in general is a mesh of the sort a k dimensional mesh where the dimensionality is 2 in each dimension.

(Refer Slide Time: 14:45)



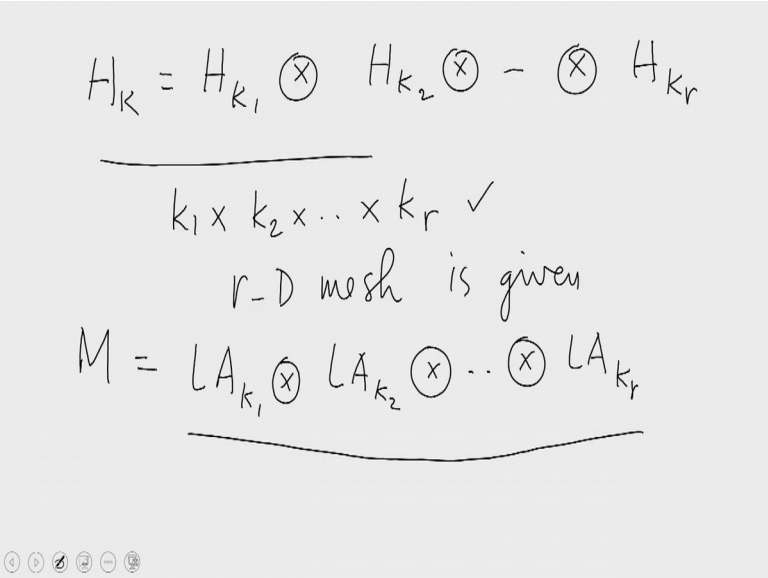
Therefore, in general we can say that a k dimensional hyper cube is a k dimensional hypercube can be expressed as, a cross product of graphs in this fashion this is because Hk is a k dimensional mesh of the sort; Hk is the k dimensional mesh of the sort. Now this can be written like this.

(Refer Slide Time: 16:13)



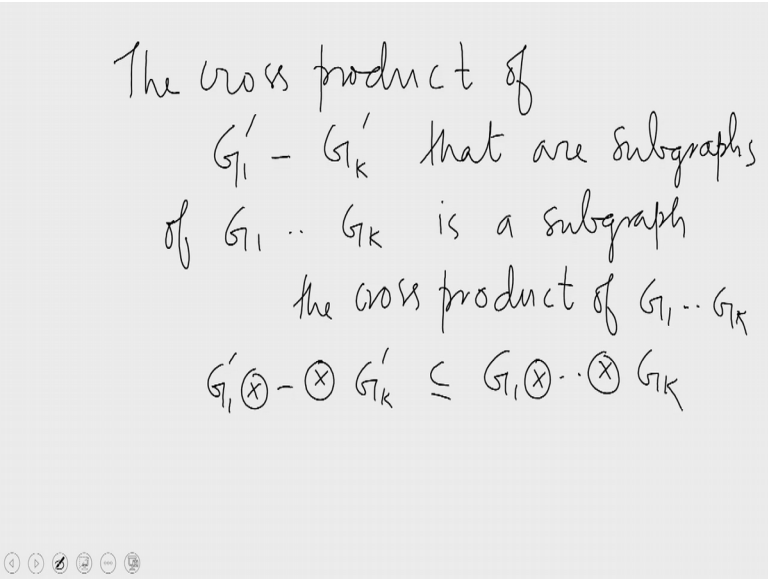
HK can be written like this. Where we have k hypercubes k copies of H1 when taken when operated under this operator will get will give us Hk, this is because as we have just seen H1 is a Hk is a k dimensional mesh of dimensionality to along each dimension. We can express this in this manner because of the associative property of the cross product operator and so on, until we come to the last group. Where the some of the case is k, but then as we have just seen when we take the cross product of k1, H1 what we get this Hk 1.

(Refer Slide Time: 17:49)



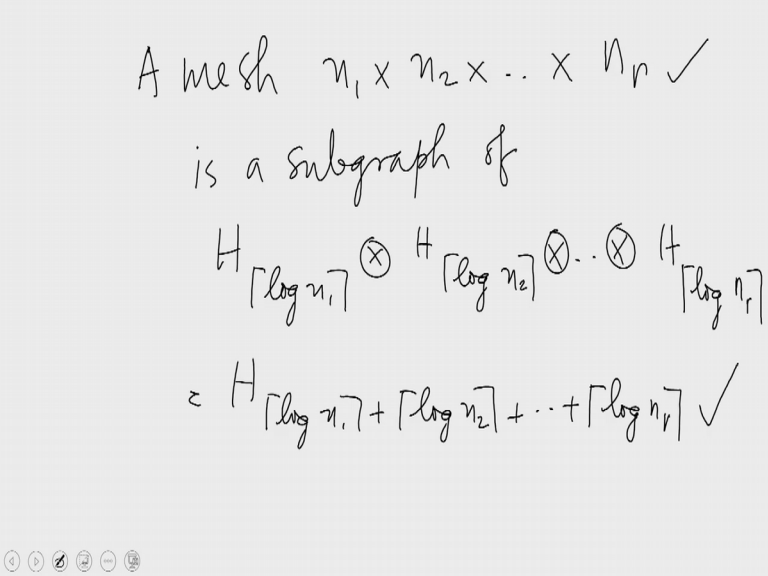
Therefore Hk can be expressed as the cross product of these graphs. Now let us see we are given, a mesh of the sort in r dimensional mesh with ri as the ki as the dimensionality along the i th dimension is given to us. This can be expressed as the cross product of LA of k1, LA of k2 etcetera. So, an r dimensional mesh of the sort can be expressed as a cross product of several linear arrays of this form. Now what we find here is this LA k1 is a sub graph of Hk 1 and LA k2 with a sub graph of Hk 2 and so on and LA kr is a sub graph of Hk r. Now Hk is the cross product of these super graphs and the mesh M that is given to us is a cross product of these sub graphs.

(Refer Slide Time: 19:23)



I will stay here without proof that, the cross product of G1 prime through Gk prime; that of sub graphs of G1 through Gk is a sub graph of the cross product of G1 through Gk. This is easy to visualize, but I am leaving the formal proof to you, in other words the cross product of G1 prime through Gk prime is a sub graph of the cross product of G1 2 Gk. So, when we apply the principle here what we find is that M which is a cross product of the linear arrays is a sub graph of Hk. In other words since LA k1 is a sub graph of Hk 1, LA k2 with a sub graph of Hk 2 and so on. The Cross product of the LA is a sub graph of the cross product of Hk 1, Hk 2 etcetera. In the former is nothing, but M and the latter is nothing, but Hk.

(Refer Slide Time: 21:07)

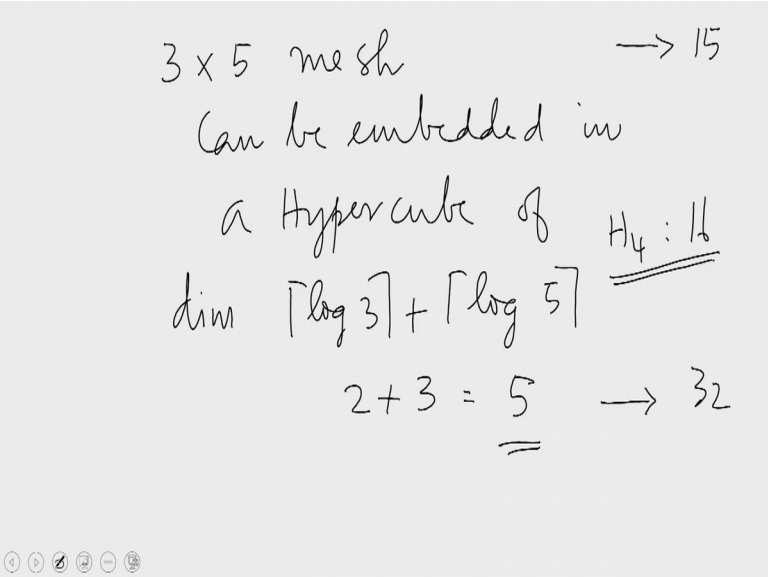


Therefore, what we established now is that; a mesh of this form is a sub graph of, why is that when we are given a mesh of the sort. This mesh can be expressed as a cross product of linear arrays of size N1, N2 etcetera up to Nr, a linear array of size N1 is embedded in a hypercube of dimensionality log N1 a hypercube of log N1 dimensionality, will have N1 vertices therefore, if you take a Hamiltonian cycle of this hypercube. It will have more than N1 vertices.

So, N1 embeds in this hypercube, linear array of size N1 embeds in this hypercube. Similarly for all the other linear arrays therefore, the mesh which is a cross product of all these linear arrays embeds in a hypercube of the sort. But this is nothing but a hypercube

of for these many dimensions. Therefore, a mesh of dimensions like this can be embedded in a hypercube of the sort.

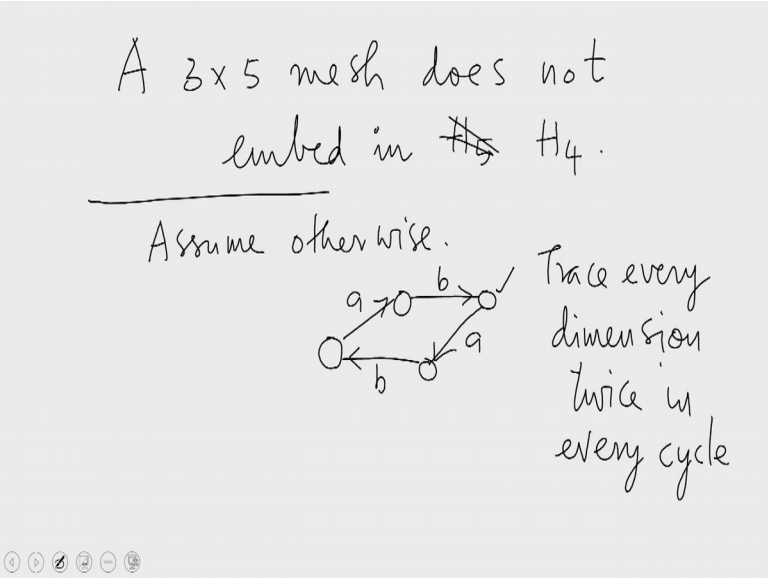
(Refer Slide Time: 23:04)



In particular a 3 by 5 mesh can be embedded in a hypercube of dimensions, ceiling of log 3, plus ceiling of log 5. Ceiling of log 3 is the same as log 4 which is 2 and the ceiling of log 5 is the same as log it which is 3. So, you require a hypercube of dimensions 5 to embed a 3 by 5 mesh, this establishes that a 3 by 5 mesh embeds in a 5 dimensional hyper cube, but then a 3 by 5 mesh has only 15 processes. Hypercube of dimensions 5 has 32 processors.

So, what we have established is that a mesh 3 by 5 mesh with 15 processors embeds in a hypercube H5 of 32 processors. That does not sound very impressive. Because there is a smaller hypercube which might be a candidate for embedding a 3 by for mesh. There is a 16 node hypercube, which is H4; H4 has sixteen nodes why do not we try to embed the 3 by 5 mesh in H4 which has 16 nodes. That would be more sensible if possible.

(Refer Slide Time: 24:42)



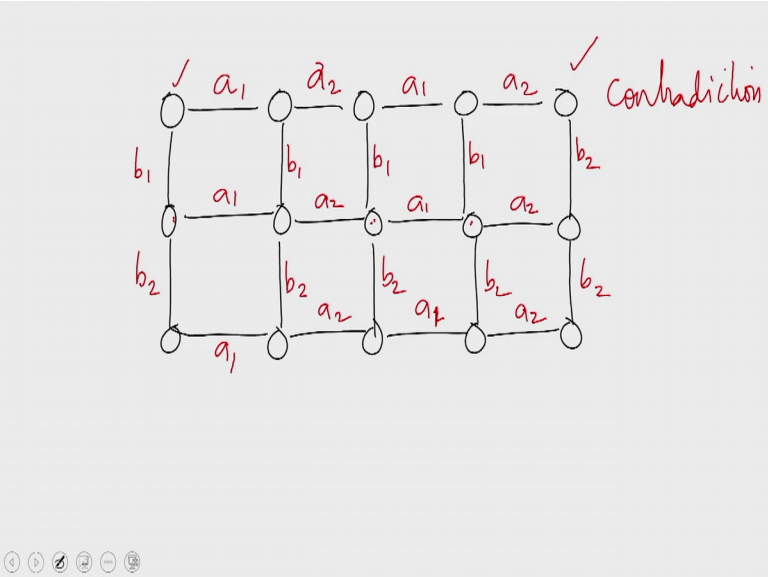
But then as it happens, a 3 by 5 mesh does not embed in H5 in H4 it embeds in H5, but we are going to claim that it will not embed in H4. So, let us prove the statement that a 3 by 5 mesh does not embed in H4. Assume the contrary assume that a 3 by 5 mesh embeds in H4. Now consider an embedding of the 3 by 5 mesh in H4 or in general any embedding of a mesh in H4, if you start at a vertex in a hypercube and then travel along some nodes in the hypercube and come back to the original vertex.

Then in the hypercube we would have preached every dimension twice. In every cycle of a hypercube we will have to trace every dimension exactly twice. That is because the cycle completes the loop and takes you back to where you began therefore, every dimension will have to be traced back and forth. If you have crossed a dimension you will have to cross the same dimension back to get back to the original point. Therefore, every dimension that you cross will have to be recrossed in the opposite direction.

So, every cycle of a hypercube has this property every dimension is traced an even number of times it traced exactly twice once in each direction. In particular if you consider a cycle of length 4, there you find that there are exactly 2 dimensions here. Because any dimension that is cross has to be recross, there can be only 2 dimensions in a cycle of size 4. So, if 1 dimension is of the 8th one and the 2nd one is the bth one, then you have to retrace these dimensions back, but then at any node in the Hypercube there is only one edge of a particular dimension.

So, if you consider this node here, it has only 1 edge of dimension b therefore, it is not possible to trace this edge trace another b dimensional edge from this vertex. Therefore, all you can do is to, take a dimension a edge from here and then complete the loop with the dimension b edge. Therefore, if you take any 4 cycle in a hypercube there you find that 2 dimensions alternate ab ab. This is the only way you can have a 4 cycle in a hypercube. Now we have let us come back embedding; the hypothesized embedding we assume that we have a 3 by 5 mesh that has been embedded in a 16 node hypercube.

(Refer Slide Time: 28:07)



So, let us consider the 3 by 5 mesh. So, as you can see a 3 by 5 mesh has several 4 cycles. So, if you manage to embed a 3 by 5 mesh in NH 4 these 4 cycles will correspond to some 4 cycles of the hypercube. Now in particular let me consider this edge this is of dimension this is a long dimension a 1 let us say, the marked edges along dimension a 1. That is we assume that this 3 by 5 meshes embedded in H4 let it be that this edges embedded along dimension a 1. Then the vertical edge incident with that vertex, let us say is along some dimension b 1. Now b 1 is not the same as a 1 because there cannot be 2 edges of the same dimension at any vertex.

So, a 1 is different from b 1 now in any 4 cycle the opposite edges are along the same dimension therefore, this edge will have to be of dimension a 1 and this edge will have to be of dimension b 1. Now the same property tells us that this edge is also a long dimension b 1; so is this and so is this. Similarly this edges along dimension a 1 as well.

Now at this node we have 3 edges to are along dimensions b1 and a1 the 3rd one will have to be along a third dimension. So, this let us say is the long dimension b2, I will use b is for the vertical dimensions.

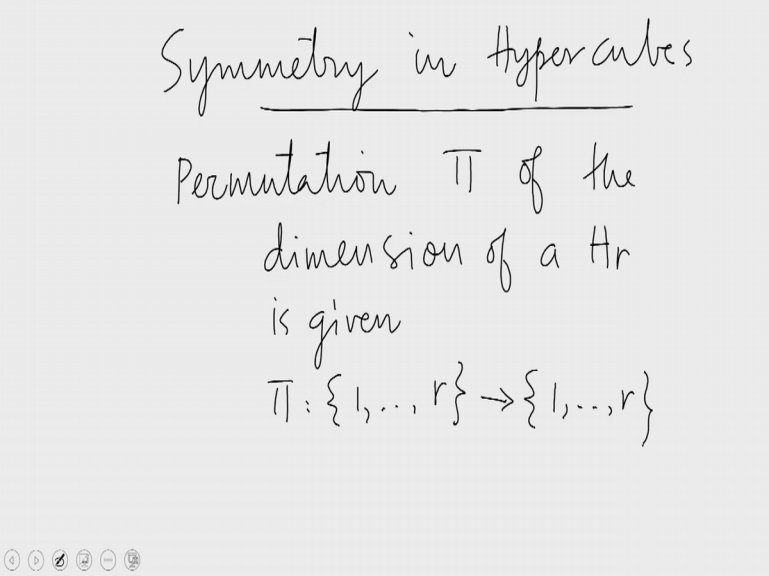
So, this is just along dimension b 2, let us say in that case these edges are also along dimension b2. Because opposite edges of a 4 cycle are along the same dimensions. Now let us look at this edge; this dimension will have to be different from both a1 and b1. And it will also have to be different from b2, because whatever dimension you assigned to this edge the same will be assigned to the edge here which is adjacent to dimensions a 1, b 1 and b 2 therefore, this will have to be a 4th dimension. Since this is a horizontal dimension a dimension assigned to a horizontal edge of the mesh let me mark it a 2; that tells us that these edges are also of dimension a 2. Now remember we are attempting to embed the 3 by 5 mesh in a 4 dimensional hypercube.

So, there are only 4 distinct dimensions and we have already used up all for listing dimensions a 1, b 1, a 2 and b 2, therefore, if the embedding has to be possible we will have to fill the remaining unmarked edges also with these 4 labels. Now coming to this node we find that it has a2, b1 and b 2 already assigned to the adjacent incident edges therefore, the only free dimension is a 1. So, this edge will have to be along dimension a 1. If we do not want to use a fifth dimension therefore, these 2 are also along dimension a 1, opposite edges are along the same dimension in any 4 cycle. By the same argument when we consider this node we find that a 1, b 1 and b 2 are already taken therefore, the unlabeled edge here will have to take a 2 therefore, this edge also will have to take a 2, the bottom edge also will have to take a 2.

Now every edge is labeled it would seem that an embedding has gone through, but it hasn't, that is because when i consider the top row I find that along the top row. The dimensions alternate a 1, a 2, a 1, a 2. Now this has been embedded in a hypercube according to our hypothesis. But in that case in a hypercube, when we start at any vertex and use exactly 2 dimensions a 1 and a 2 and use them in this order when you go from the first vertex to the second vertex you cross in dimension a 1 then you cross in dimension a 2. Then you use dimension a 1 you are crossing back in dimension a 1 and then you cross back in dimension a 2 in a hypercube you would be getting back to the original vertex. When you cross in dimension a 1 you are flipping the a 1 bit of the binary representation of the label of a node.

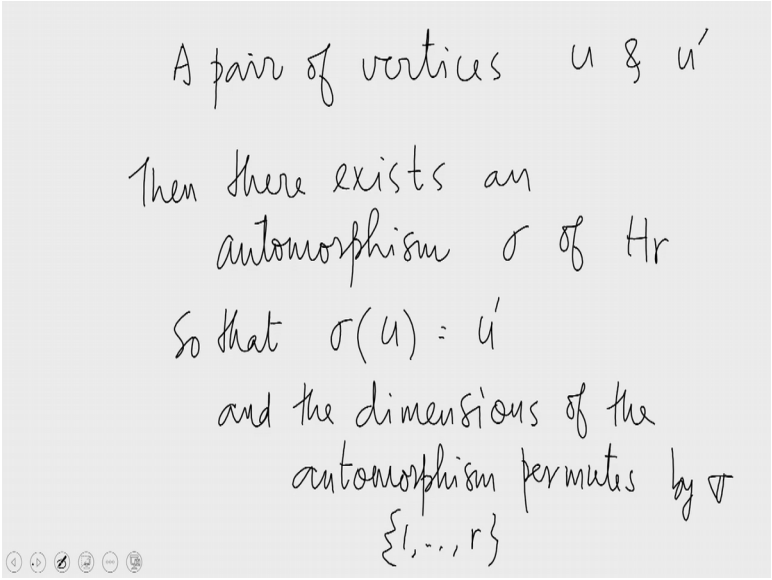
So, you flip the a 1th bit, then the a 2th bit, then again the a 1th bit and then again the a 2th bit, you will get the original representation back. So, you have to be back in the original vertex, which means this vertex and this vertex are both embedded onto the same vertex of the hypercube, which is not possible. In an embedding of a graph, into another graph we have to map every vertex of the first graph into a unique vertex of the second graph. Here we find that 2 vertices of the first graph are mapped on to the same vertex of the second graph, the mapping has to be 1 to 1, which is not the case here therefore, such an embedding is not possible. So, this is the contradiction that we have to write. So, it is not possible to embed a 3 by 5 dimensional mesh in a 4 dimensional hypercube. So, what we have seen is that a 3 by 5 mesh cannot be embedded in a 4 dimensional hypercube. Now let us see some other embedding properties of the hypercube,

(Refer Slide Time: 33:58)



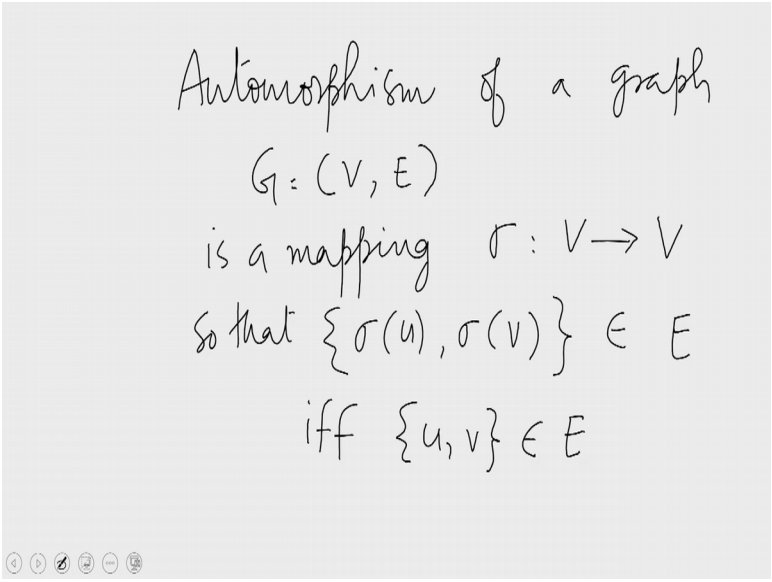
But first let us see some symmetric properties of the hypercube architecture. Let us say we are given a permutation of the vertices, a permutation pi of the dimensions not the vertices, the dimensions of a hypercube of r dimensions is given. So, pi is a function that maps its 1 to 2 to 1 to r. So, it is a permutation, so, it is a 1 to 1 mapping.

(Refer Slide Time: 35:03)



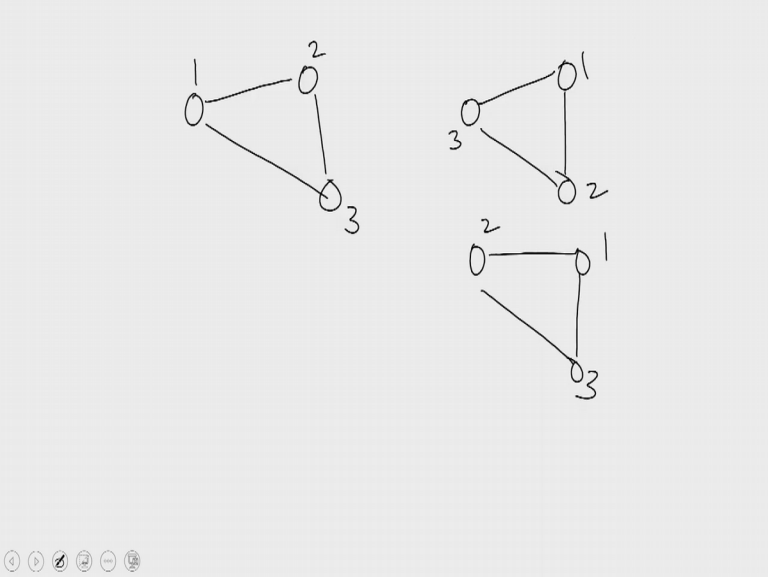
So, this permutation is given to us and in addition to this let us say we are also given a pair of vertices, u and u prime then there exists an automosphism sigma of Hr. So, that sigma of u is equal to sigma of u is equal to u prime and the dimensions of the automosphism, permutes by pi the set 1 to r that is the original dimensions of the hypercube Hr are permuted using pi, in the automosphism. So, this is an interesting property of a hypercube.

(Refer Slide Time: 36:33)



So, to explain this, let me first define what an automosphism this? An automosphism of a graph G is a mapping sigma from V to V. So, it is a renaming of the vertices of G, so, that for 2 vertices u and sigma of u and sigma of V are adjacent in the automosphism are adjacent in E, adjacent in graph G, if and only if u and V are adjacent graph E. So, what it means is that, if you rename the vertices of G using the function sigma then what you get is exactly the same graph.

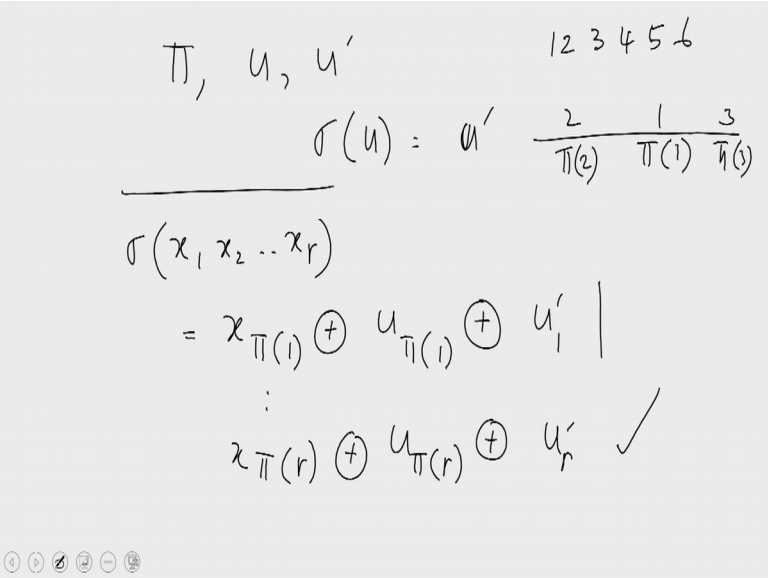
(Refer Slide Time: 37:56)



For example, the trivial example would be a 3 cycle, suppose these are the original names. If you rename the vertices in any order we will get exactly the same graph, the same adjacency relation will be obtained these are all the same graph. So, you can make this claim trivially for every single click, every single complete graph and for some of the other graphs this claim would be quite non trivial. If some of the edges are missing then making this claim would be certainly non trivial.

So, what we have claiming here is that for an r dimensional hyper cube when we are given a permutation pi and a pair of vertices u and u prime then there x is a renaming of the vertices of the hypercube in such a way that the dimensions are permuted according to pi and all the adjacencies of the hypercube are maintained. Let us even under the renaming with the permutation; with the permutation pi of the dimensions realized, the graph would remain a hypercube. The adjacencies of the hypercube would remain the same. So, this can be shown like this.

(Refer Slide Time: 39:20)



So, pi is the permutation given to us, u plus u prime are the vertices given. So, what we want is this we want the vertices of the hypercube to be renamed. So, that u gets mapped to u prime, we want sigma of u to become u prime. And the other vertices must be renamed accordingly so that, 2 vertices that are adjacent in the hypercube will be adjacent even under the renaming. So, what we are going to do is this? We will effect and renaming of the sort, let us say for a node x 1 through x r, sigma is defined as the vertical bar denotes the contactination operation, we use the exclusive or operator here. So, a vertex that is labeled x 1 through x r in the original hypercube will be relabeled in this fashion. The new label will be obtained in this fashion, the first bit of the new label will be the xr of these 3 bits.

You take the pi 1th bit of x the pi 1th bit of u and the first bit of u prime take the x r of these 3 that will give us the first bit of the new name of x 1 through xr. So, how does this achieve what we want? According to the permutation pi the first dimension is going to occupy the pi 1th position and so on. When we are given a permutation pi of 1, 2, 3, 4, 5, 6, these are the original positions, original dimensions and if the permutation pi is in this sort is of the sort then the pi 1 position will be occupied by 1 the pi 2th position will be occupied by 2 and so on. So, what we find is that, when the renaming is affected in this manner then the desired properties are achieved. So, how exactly this will work we shall see in the next lecture. So, that is it from this lecture hope to see you in the next.

Thank you.