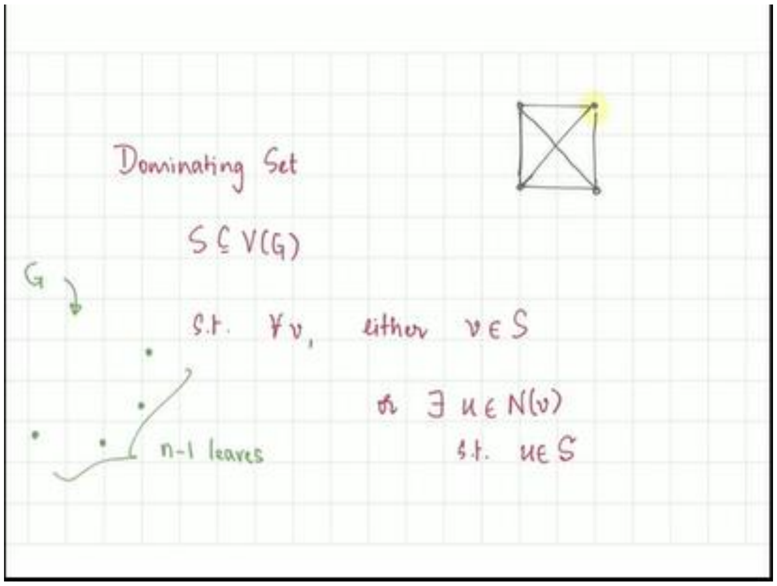
# Parameterized Algorithms

**Prof. Neeldhara Misra**
**Prof. Saket Saurabh**
**The Institute of Mathematical Science**
**Indian Institute of Technology, Gandhinagar**

## Lecture – 50
## Reductions – Problems as Hard as Clique (Dominating Set, Set Cover)

Welcome back, last time we stopped with showing that multi-coloured independent set is at least as hard as clique and I promise that I will show you an application of knowing that this problem is hard. And we will see how this multi-coloured format comes in handy when doing reductions. So, the example that we are going to use to illustrate this is going to be dominating set and, in this segment, I will talk about both dominating set and set cover.

**(Refer Slide Time: 00:41)**



So, let us get this started with the definition of dominating set. So, in a graph a dominating set is essentially a subset of vertices which is such that every vertex either belongs to this subset or at least has a neighbour in this subset. So, superficially when you look at this problem it may remind you of something like vertex cover. It does seem to have some similarities but actually it turns out that it is a very different problem and there are in fact a lot of contrasts with something like vertex cover.

Just as an example consider a complete graph on n vertices the smallest vertex cover has size n - 1 whereas the smallest dominating set has size just one as you can see from the example that is drawn here. In fact, any graph that has a global vertex which is a vertex that sees or is adjacent to every other vertex will have a dominating set of size one. Another property that sets dominating set apart from problems like vertex cover is the fact that the property of having the small dominating set is not a hereditary property.

Remember that when we were working with techniques like iterative compression it was useful that the property of having a small vertex cover is in fact hereditary. So, that when we discovered a sub graph which was a no instance, we were able to conclude correctly that the whole graph which is a super graph must also be a no instance. On the other hand, notice that this is not true for dominating set.

You could have a graph which has a small dominating set but it has a sub graph that does not have a small dominating set. See if you want to take a minute here and come up with an example that demonstrates this kind of behaviour. One hint would be to see how global vertices may play a role. Alright so, hopefully you had a chance to play with some examples here is one where this happens in a slightly dramatic way.

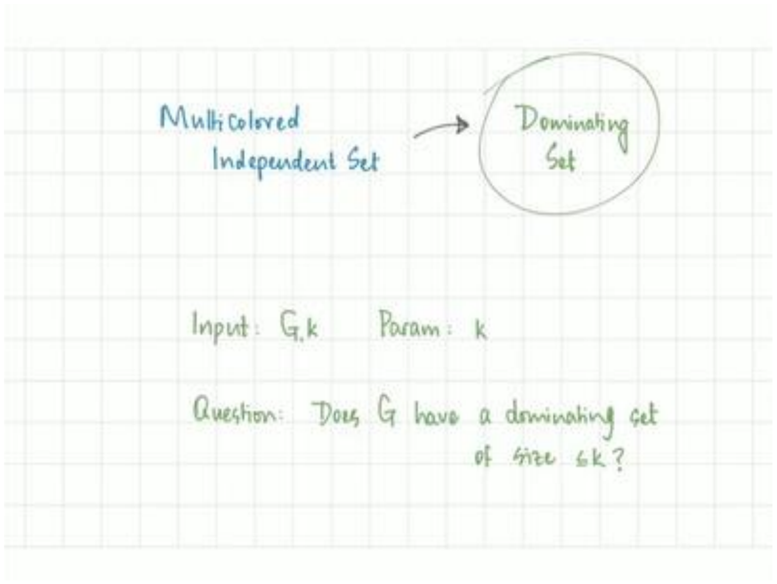
So, suppose you have a graph which is essentially a star with n - 1 leaves then notice that the graph itself has a dominating set of size one you could just pick the centre of the star. And that is a global vertex and a valid dominating set. But if you were to remove the centre of the star from the graph G then the graph that you were left with is just going to be an empty graph on n - 1 vertices. That is to see a graph without any edges.

And in such a graph the only valid dominating set would be one that picks every single vertex in the graph because there is really no opportunity for domination via edges. So, here is a graph which had a dominating set of size 1 to begin with but it has a sub graph whose optimal dominating set size is as large as n - 1. So, the gap between the optimal dominating sets of a graph and a sub graph could really be arbitrarily large.

And certainly, the size of an optimal dominating set of a graph is not a valid bound for the sizes of the dominating sets of the subgraphs. So, this is another point of contrast between dominating set and problems like vertex cover for example. So, dominating set happens to be a very, very fundamental problem, it is extremely well studied and it has a lot of applications as well. So, it is natural that we want to also look at it from the parameterized perspective.

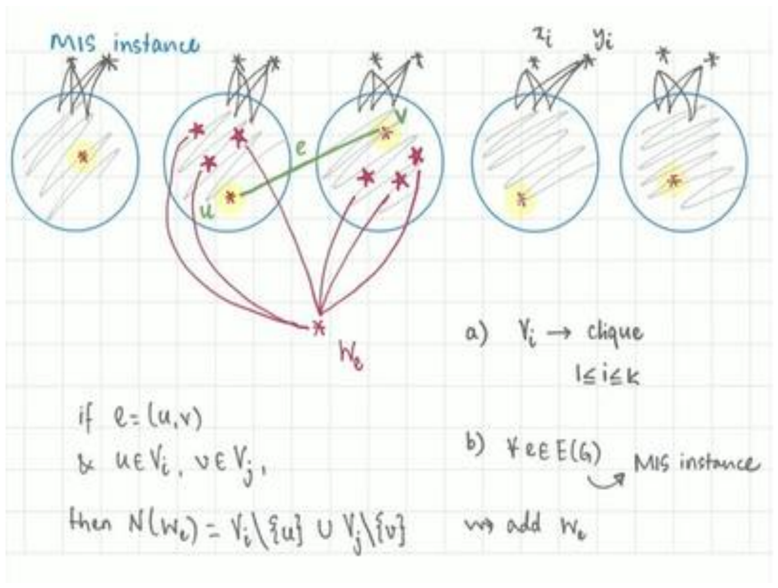
And we are going to start by working with the natural parameter which is the solution size and as I said because of the superficial similarities between dominating set and some other problems that we have seen. It may be tempting to conjecture that you can come up with FPT algorithms for this problem. But it turns out that dominating set also is as hard as clique and is therefore widely believed to be unlikely to be FPT.

**(Refer Slide Time: 04:27)**


Multicolored Independent Set
Dominating Set
Input: G, k
Param: k
Question: Does G have a dominating set of size ≤ k?

So, we are going to actually show this in the rest of our discussion here and we will do this by reducing multi-coloured independent set which we already know to be at least as hard as clique to the dominating set problem which is what we are considering right now. So, here is the computational version of the dominating set problem. The input is a graph and a budget k which is a positive number and k is going to be our parameter as well. And the question is if G has a dominating set of size at most k?

**(Refer Slide Time: 04:58)**


MIS instance
$x_i$ $y_i$
e
u
v
$w_e$
a) $V_i \rightarrow$ clique $1 \leq i \leq k$
b) $\forall e \in E(G)$ MIS instance
we add $w_e$
if $e = (u,v)$
& $u \in V_i$, $v \in V_j$,
then $N(w_e) = V_i \setminus \{u\} \cup V_j \setminus \{v\}$

So, we are going to start off as we always do for reductions with an instance of multi-coloured independent set. And let us just say that these are the colour classes here for convenience I am going to work with five colon classes you can just imagine that. In general, you will be of course working with v 1 through v k. Now we want to modify this graph and somehow turn it into an instance of dominating set.

In such a way that the dominating set instance has a dominating set of a certain size if and only if this original instance that we are working with here has a multi-coloured independent set. So, somehow you want your independent set to manifest as a dominating set in an appropriately modified graph. And also, the other way around you want the dominating set to give you some hint or signal about an appropriate independent set.

In this case what you would want is a multi-coloured independent set. So, at this point I could just go ahead and describe the construction for you and then we could argue the equivalence. But I want to see if we can build this out a little bit by just playing around and thinking it through. If you remember when we; were reducing from independent set on regular graphs to partial vertex cover.

We were kind of lucky in the sense that was a very simple reduction we could just work with the same graph. And the independent set itself became a partial vertex cover. So, let us see if we by chance get a little bit lucky like that here as well. So, let us start off with a multi-coloured independent set what we know about any multi-coloured independent set is that it picks a different vertex from every colour class.

And the entire vertex set is essentially partitioned into these colour classes. So, let me go ahead and mark some multi-coloured independent set in the input instance. Now let me see if I can pitch this to you as a dominating set in the same instance. Now it seems like this is not at all clear as to why this should be a dominating set. In fact, it is quite easy to come up with examples where you could pick up a multi-coloured independent set.

And it would in fact not be a dominating set of the same graph at all. So, we need some way of turning this independent set into a dominating set. Perhaps at least that is one way of thinking about it there could be any number of other ways of going about this but let me see if we could use this as a starting point and then try and take it from there. So, instead of focusing on the independent; set aspect of this subset of vertices.

Let us focus on the fact that it is a multi-coloured set which is to say that this is a subset of vertices which picks exactly one vertex from each part of the graph. And this is going to be a feature of any solution in the multi-coloured independent set instance. So, can we design our constructed graph in such a way that any subset of vertices that has this property actually ends up dominating all the vertices.

Notice that this will not really give us exactly what we want. What we want is not that any multi-coloured set should manifest as a dominating set. We only want the multi-coloured independent sets to become dominating sets. But we will come back and worry about that in the next phase but first let me try and see if I can get any set of k vertices that come one from each part to actually behave like a dominating set.

So, this is of course a bit more than what we want and we will have to come back and readjust a little bit. But take a minute here and see if you can come up with a way to modify this graph. So, that; any subset of k vertices that are colourful end up dominating all the vertices that are there in the graph. Think about the fact that whenever you have global vertices, they dominate a whole lot of vertices at once.

So, see if you can use that structure to make this happen. So, if you thought about it a bit you might also end up at this sort of a construction where we say that we will turn every part into a clique. So, what we will do is essentially add all possible edges within these parts which will ensure that every vertex within every part is global within that part. So, no matter which way you choose a multi-coloured set.

These vertices will end up dominating every vertex in their part and because the graph is essentially a partition into these parts every vertex will end up getting dominated. Because it belongs to some part and you would have chosen a vertex from that part. So, far we have this simple construction which simply says that take your instance of multi-coloured independent set. And for any pair of vertices that belong to the same part in the multi-coloured independent set instance.

Let us just go ahead and add an edge. What this does is? It makes sure that the graph induced by any part of the MIS instance is in fact a clique. And therefore, any colourful subset that you pick is actually a dominating set. So, in some sense this actually gives you the forward direction because if you do have a multi-coloured independent set then you in fact also have a dominating set of size k but really this is not a very meaningful reduction so far.

Because not just multi-coloured independent sets but any multi-coloured set will manifest as a dominating set. So, the dominating set is really not distinguishing between yes and no instances of multi-coloured independent sets. But it is a baby step hopefully in the right direction. So, now let us think about how we can in some sense fix the reverse direction. We have over indulged in dominating sets and we want to get rid of some of them.

In particular if you have a multi-coloured subset that is not an independent set then we do not want it to manifest as a dominating set here. We want to say that okay look this subset is problematic it is not a dominating set. We need to add some sort of a gadget or some additional structure in this graph that can encode this information about independent sets. So, what does a subset of vertices which is not an independent set look like in the original MIS instance?

Well, it looks like a subset of vertices where some pair of vertices has an edge between them. So, let us consider a situation where we have an edge between one of the highlighted vertices here. So, for example let us say that we have an edge between the second vertex and the third vertex here and hopefully that is visible. So, let me call this edge e and now what I want to do is say that okay look if you chose this particular subset of vertices.

And in particular if you choose these two vertices together in fact let us also go ahead and label these endpoints here. So, let us say e has endpoints u and v. So, we want to say that any set which tries to pick u and v together cannot be a valid dominating set. And we want to say that that happens because something gets missed out. If you choose to pick u and v together. By the way I should say that this feels like a counter-intuitive goal.

A more natural thing would be to say something like if you do not pick some subset of vertices then you miss out on dominating something. But now we are saying that if you do pick these two vertices then you miss out on dominating something. So, how would something like that work? Well, one way to think about this philosophically would be to say that what if choosing these two vertices came at the expense of not choosing something else.

Then when you do not choose something else then that is why you miss out on dominating something. So, how do we say that choosing these vertices implies that you are not allowed to choose something else? A natural thing to say is that for example if I commit to choosing the vertex u from the second part then I cannot pick any other vertex from the second part. Suppose that was true and let us say it was also true that if I choose to pick v from the third part.

Then I am not allowed to pick anything else from the third part. So, right now there is nothing that ensures this kind of structure on the dominating set. We will come back in a minute and talk about how to force dominating sets to behave in this way that they are only allowed to pick one vertex from each part right now. Our dominating set of size k could be all over the place and there is really no reason to believe that it would be a colourful set.

So, in fact remember that although we are starting from multi-coloured independent set we are reducing to normal dominating set. So, the dominating set solution is not obliged to you know respect the structure that you had in the MIS instance. It can do anything but because we are constructing this reduced instance, we can play around with it and perhaps add some gadgets to force the dominating set to have the kind of structure that we wanted to have.

So, we will come back to that in a minute but for now imagine that somehow dominating sets in this graph were also constrained to be colourful in the sense that they were supposed to be spread out across these parts. And you were not allowed to pick more than one vertex from any part. If your dominating set has the structure then picking the vertex u comes at the expense of not picking any other vertex in the part that u belongs to.

And picking the vertex v likewise also comes at the expense of not picking any of the other vertices in the part that v belongs to. Now this gives us the following idea. Perhaps we can introduce a vertex let me call it w sub e. This vertex has been introduced to remember the presence of the edge e and what do I want to connect this vertex to. Well, I want to connect it to every vertex in the part that u belongs to except for u and every vertex in the part that v belongs to except for v.

With this sort of a neighbourhood w sub e will not get dominated if you choose to pick u and v from the paths that you and v belong to. So, now we have just added the edges incident on w sub e and let me just recap the construction that we have described so far. So, to begin with we said that we will take every part and turn it into a clique and further what we are saying now is that we will take every edge in the multi-coloured independent set instance.

And we are going to add a vertex corresponding to that edge which is essentially adjacent to everything in the pair of parts that the edge has its end points in except for the endpoints of the edge itself. Notice that I am implicitly assuming here that every edge has its endpoints in two distinct parts in the multi-coloured independent set instance. And this is an assumption I can make without loss of generality.

Because if somebody gives you a MIS instance where the parts have edges inside them. You can simply ignore these edges because you anyway know that structurally a solution to a multi-coloured independent set instance will never pick two vertices from the same part. So, as we have hinted to before while talking about multi-coloured clique as well. What goes on inside these parts does not really matter and here what I am going to assume is that each part was independent to begin with.

And of course, in our construction we are going to turn them into cliques but when I am in the second point talking about edges in the multi-coloured independent set instance. All of these edges can be assumed to be edges that go across distinct parts. So, hopefully that makes sense. And so, this is what we have so far and this is a good time to do a bit of a health check-up. Let us see how we are doing in terms of our proof of equivalence.

And this will be an opportunity to figure out if we are done and if not, what more do, we need to do. So, first let us tackle the forward direction which is the question of whether a multi-coloured independent set remains a dominating set as it did before. Notice that now apart from the vertices of G we have these m new vertices to worry about. So, let us start with a multi-coloured independent set and ask ourselves if every vertex in the constructed instance gets dominated by this multi-coloured independent set.

I claim that every vertex does indeed get dominated. So, the original vertices get dominated for the same reason as before because of the clique that we introduced in the first step. So, every part has a representative vertex from the multi-coloured independent set. And that is going to take care of all the vertices of G that come from that part. And since every vertex of G belongs to some part all of these vertices are easily accounted for.

Now let us turn to the vertices that we have introduced. We have introduced one vertex for every edge and let us look at what happens if one of these vertices is not dominated by a multi-coloured independent set. In particular just for convenience because it is already on the slide let us say that this vertex our friend here w sub e is not dominated by a multi-coloured independent set. Well, what does that mean?

It means that none of w e's neighbours got chosen by the multi-coloured independent set. And notice that the only way that this can happen is if the multi-coloured independent set chose the vertex u from the second part here and the vertex v from the third part. Any other choice of vertices from the second and third part would ensure that w sub e gets dominated. So, the only way to MIS w sub e is to actually pick both of its endpoints.

And now hopefully this is beginning to already sound funny to you how can an independent set pick both the end points of an edge. So, that would be your contradiction. So, if the vertex w sub e is not dominated by a multi-coloured independent set, then this subset of vertices is not even an independent set and that would be a contradiction. Therefore, any multi-coloured independent set still continues to dominate every vertex that we have introduced in the constructor instance.

Now let us take a look at the reverse direction which is what we wanted to fix and which is why we introduced all of these vertices in the first place. Let me make a simplifying assumption. So, suppose somebody gives us a dominating set that also picks one vertex from each part in the graph G. So, the graph h is all of these parts plus these m additional vertices. Let me say that we are given a dominating set which happens to have this nice structure.

I will say that this is not without loss of generality there is really no reason to expect any dominating set in this graph to actually have this structure. Once again let me emphasize that we are not reducing to some multi-coloured dominating set when we come to the dominating set instance. The dominating set instance has no knowledge about this special partition of some part of the graph that it is working with.

So, it is really free to pick any subset of k vertices as long as they get the job done and there is no good reason for now to expect that these k vertices will be nice and spread out. You could easily come up with examples where all the vertices in some part get dominated by some subset of vertices that do not belong to that part. So, you do not necessarily have to use the clique to actually dominate.

It is one way of doing it, it works in the forward direction but in the reverse direction you could have much more chaos when you are considering an arbitrary dominating set. Having said that, let us just make this as an explicit simplifying assumption and then we will come back and worry about how to justify this assumption. So, if the dominating set was indeed nice and spread out and it picked one vertex from each part then I claim that it must in fact also be an independent set.

So, as before, let us try to argue this by contradiction. So, suppose somebody gives you a structured dominating set that picks exactly one vertex from each of the parts of the MIS instance. And suppose that this subset of vertices is not an independent set back in the graph G. So, what that means is that somehow our dominating set has managed to choose a pair of adjacent vertices and for convenience let me say that the adjacent pair that our dominating set has chosen is the pair u v connected by the edge e.

This is just so that you can see the scenario on your screen here and remember what we are targeting is some sort of a contradiction. We want to say that this dominating set actually fails to be a dominating set; there is some vertex that it does not successfully dominate. But notice that we have such a vertex literally by construction this is our moment of truth, it was an anticipation of this situation that we introduced explicitly the vertex w sub e.

And notice that any dominating set that picks exactly one vertex from each part must have necessarily not chosen any of the neighbours of w sub e as soon as it commits to picking the vertices u and v from the parts that you and v belong to. Therefore, whenever the dominating set chooses to pick an adjacent pair there is going to be a vertex that it will necessarily fail to dominate provided it has the structure that it picks exactly one vertex from each part.

This structure is important because if the structure was not there then it is conceivable that the dominating set may have picked some other vertices from these parts which take care of the vertex w sub e. But as such we have assumed that the dominating set is nice and spread out and it picks exactly one vertex from each part in which case, we do get the contradiction that we want. So, now let us come back to this assumption that we are making about the structure of the dominating set.

So, remember we have already discussed that this assumption is not without loss of generality, your dominating set does not have really any reason to behave in this particular wave. But we can modify this construction further to force the dominating set to have this property. So, first let me focus on trying to ensure that any dominating set of this constructed instance must pick at least one vertex from each part.

This will eventually force any dominating set to pick exactly one vertex from each part because this is overall budget of k and there are k paths. So, if you are looking for a dominating set of size at most k and you have somehow ensured that the dominating set cannot avoid any part. Then it follows that any search dominating set must in fact pick exactly one vertex from each part because really there is no wiggle room here.

So, let us just focus on ensuring that any dominating set must pick at least one vertex from each part. How do we do this? Well, what we could do is introduce some dummy vertices in these parts which do not get dominated by any vertices except for vertices in that part itself. So, let us make this idea a little more concrete by saying that we will introduce a vertex for every part that is a neighbour of every vertex in that part and nothing else.

So, this is in some sense a locally global vertex. So, for path i you are introducing a new vertex let us call it x i which is adjacent to every vertex in v i. That is what we are doing here and notice that this almost does it what I want to be able to say is that if you missed picking vertices from v i then that is not on because then you would have missed dominating x i. As such this statement is mildly untrue because you could avoid all of v i and still dominate x i by just picking the vertex x i itself.

Normally you would not do this because when you do this then you miss out on opportunities for dominating vertices corresponding to edges and so on. But then the justification just becomes longer. So, to make our lives easier what we can do is you know because it does not really cost us anything. Let us introduce another global vertex that I will call y i and y is again simply adjacent to all of v i and it is not adjacent to x i so, it does not get dominated by x i.

And similarly, x i also does not get dominated by y i and once again we are going to do this for every part. And now notice that if you have a dominating set that does not pick anything from the part v i then it is forced to pick both x i and y i because if you did not pick x i and y i then whatever vertex you did not pick would actually not be dominated. Because it is entire closed neighbourhood is simply omitted from the dominating set.

So, in some sense the dominating set has to pay two units, it has to pick two vertices somehow associated with the part v i which means its total budget is now down to k – 2. And now it has to somehow dominate everything else from k -1 remaining parts using a budget of just k - 2 and you can convince yourself that is really not feasible at all. Because certainly every part requires at least one vertex in the dominating set because of if nothing else the x size.

So, the x i's are vertices whose close neighbourhoods consist of x i union v i and at least this cannot be omitted which means that you would basically be stuck if you tried to pick both x i and y i for any part. And because you cannot do that within the constraints of a budget of k, it must be the case that you are compelled to pick a vertex from v i in any dominating set of size k. So, although the semantics of the problem do not require the dominating set to be colourful or to have any special structure by just introducing these 2k additional vertices.

We have ensured that any dominating set of size at most k must in fact have what we were calling a colourful structure in the context of the graph G. So, any dominating set of size at most k will end up picking exactly one vertex from each of the v i's. Now that we have this structure, we can legitimately conclude that this dominating set in the constructed graph actually corresponds to an independent set in the original graph.

Because from here; everything that we have argued before holds. However, we do have to revisit the forward direction every time we make a change in our construction, we have to make sure that both of the directions go through. But the forward direction actually remains pretty much intact because the independent set already has this multi-coloured structure. So, it automatically ends up dominating these 2k extra vertices that we have introduced.

And we have already argued that any multi-coloured independent set would also dominate the m extra vertices that we introduced corresponding to the edges and of course the vertices from g get dominated quite easily because of the clique structure that we added. So, that completes the description of the construction that we have here. Let me just quickly recap everything that we have discussed.

You start with an instance of multi-coloured independent set; you start with that graph and you take every part of the multi-coloured independent set instance and turn it into a clique. That is the first thing that we did. And then for every edge in the MIS instance which we assume always cuts across two distinct paths. We said that we will introduce a vertex that represents that edge and we made this vertex adjacent to the union of these two paths except for the end points of the h.

The endpoints of the; edge that the vertex is representing and this was the key step that allowed us to encode information about the independent set as we moved into our dominating set. We also introduced our two vertices for each part which played the role of ensuring that any dominating set that we pick has a certain structure. So, for every part we introduced vertices x i and y i which were adjacent to every vertex in v i.

But they had no other neighbours in the entire graph and they were not even adjacent to each other for example. So, the rules that these three steps in the construction played are hopefully intuitively clear. The reason we made each part of clique is so that our independent set in particular in fact any colourful set would end up comfortably dominating all the original vertices. Then we introduced these vertices representing the edges to ensure that any dominating set that picks a pair of adjacent vertices from G ends up failing to be a dominating set.

That is how a dominating set ends up being an independent set but for this to actually work we needed the dominating set to have a special structure. So, to force any dominating set to have this structure we introduced these two locally global vertices corresponding to each part. So, that is essentially the entire construction and we have already talked about the equivalence of the instances that we started with and the instance that we generate.

And notice that this whole construction can be carried out in polynomial time and once again we have k goes to k. So, the other two properties that we want from parameterized reductions also hold quite clearly. So, that completes our discussion of why dominating set is as hard as multi-coloured independent set and in turn this implies that dominating set is in fact as hard as clique because of the fact that we can chain up parameterized reductions.

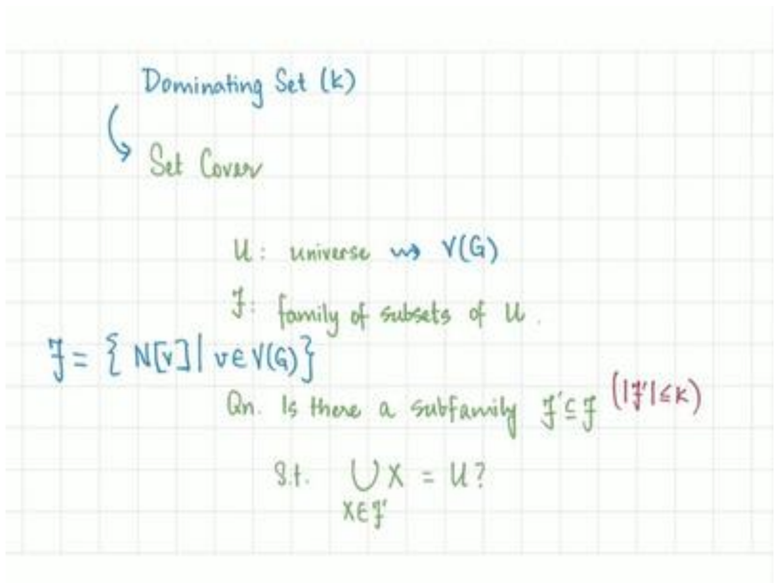
And we already know that multi-coloured independent set is at least as hard as clique. Now before we move on and talk about the next problem that I want to talk about let me just make an informal remark here. So, whenever you read reductions, you will see these constructions employing interesting tricks to make certain things happen. So, for instance here you would have noticed that we introduced these global vertices to force dominating sets to have a certain very specialized structure whenever they do exist.

Of course, if you are coming in from a no instance there would be no dominating sets to speak of but whenever you are working with the yes instances then any valid dominating set will automatically end up looking nice just because of these vertices that we introduced. So, whenever you read up on a reduction and you see a cool gadget or you see a nice trick like this. Just make a mental note of it.

Because you never know when a similar idea will be useful for something that you are working on so, it just turns out that the toolkit for reductions is not nearly as systematic or well classified as the toolkit for algorithms. So, I think the best way to develop a facility with reductions is to simply read up on a lot of them keep track of interesting ideas that come up in constructions. And try to develop your own intuition for why certain things were done the way that they were done.

Try to break it down and always think about how would you come up with something like this yourself.

**(Refer Slide Time: 33:09)**


Dominating Set (k)
Set Cover
U: universe ⇝ V(G)
F: family of subsets of U.
F = { N[v] | v ∈ V(G) }
Qn. Is there a subfamily F' ⊆ F (|F'| ≤ k)
s.t. ⋃_{X ∈ F'} X = U?

So, with that said let us talk about the next problem that I want to present to you here. And this is the set cover problem. Here the input is a universe which you can think of as a fixed and finite set and what we are given is a collection of subsets over this universe u and the question we are interested in is if there is a small subfamily that covers the entire universe. So, in other words can we find a subfamily consisting of at most k sets from the given family.

Such that when you take the union of the sets belonging just to the subfamily you end up recovering the entire universe. We do not have any additional constraints here we do not require the sets in the subfamily to be disjoined or anything like that. This would be the case if you were working with a problem like exact cover. But we are just looking at the classic variant where we have no additional constraints.

So, we want to consider set cover parameterized by again the solution size the standard parameter here and I want to show you that solving set cover with respect to the standard parameter is once again at least as hard as clique. And we are going to do this by reducing our most freshly minted heart problem which is dominating set to set cover. And again, for dominating set we will also continue working with the standard parameter.

So, the reduction from dominating set to set cover is actually reasonably natural and it kind of follows from thinking about what a dominating set solution is really doing. So, think about modelling the instance of dominating set in the following way. Let us say that the universe corresponds to the vertex set of the graph G. So, we have an element in the universe corresponding to every vertex in G.

Now think about what the sets should be so that a subfamily corresponding to a set cover somehow corresponds to a dominating set in the graph G. Take a minute here and think through this and come back once you are ready. So, I hope you had a moment to pause and think about this a little bit. So, as we were saying earlier, let us set up our universe so that we have one element in the universe corresponding to every vertex in the graph G.

So, the universe is essentially the vertex set of G and what is really crucial is how we define the family. So, just reverse engineering a bit let us think ahead and appreciate that we would want a subfamily corresponding to a set cover to tell us something about a dominating set back in G. So, we want the sets in the subfamily to correspond to elements of the dominating set which means that somehow, we would like the sets in the family to correspond also to vertices.

Because if we had a set in the family correspond naturally to a vertex then the collection of sets in the subfamily would naturally correspond to a subset of vertices. And then what we want to do is ensure that these chosen vertices actually perform as a dominating set when we go back to G. So, well it is clear so far that somehow, we want the elements of the family to also correspond to vertices.

But of course, if we just introduced singleton vertices then that is no good because then the only way to cover the universe would be to pick everything and that does not really tell us anything about the structure of the graph it does not relate to dominating sets in any meaningful way. So, let us think about what it means for a vertex to get into the dominating set. What is the work that it is doing for us?

Well, intuitively speaking a vertex in the dominating set ends up dominating itself as well as all of its neighbours. So, whenever I include some vertex in my dominating set, I do not have to worry about its neighbours anymore they were already taken care of. They may end up getting taken care of again by other vertices as well. But at least I know for sure that they have been taken care of at least once.

On the other hand, in a set cover we are trying to ensure that the sets that we pick in our family cover certain portions of the set cover. So, what if we; could ensure that the set corresponding to a vertex v ends up covering that part of the universe that is actually dominated by the vertex v back in the graph G. So, in other words I want to define the following family of sets. So, let us say that for every vertex in the graph G we introduce a subset corresponding to its closed neighbourhood in G.

So, that is going to be our family and intuitively it is really reflecting the work that is being done by these vertices if they get recruited inside the dominating set and by defining the family this way, they are going to do the analogous work in the set cover instance as well. So, let us formally try to establish the equivalence. So, suppose we start off with a yes instance of dominating set so G does have a dominating set of size k.

Then I want to say that this instance of set cover also has a set cover of the same size in fact. So, let us go ahead and pick out the sets corresponding to the vertices that belong to the dominating set and I claim that if you take the union of all of these k sets then you have covered the universe. Suppose not so suppose there is some element in the universe that is left out it is not covered by all of these sets.

Then notice that the corresponding vertex is also not dominated back in the graph G because if it was dominated consider which vertex is dominating it. Suppose it is the vertex u then this leftover vertex belongs to the closed neighbourhood of u but the closed neighbourhood of u was a subset that was chosen in the set cover. So, it could not have been that this element corresponding to this vertex was left out.

So, hopefully it is clear that if you do have a dominating set then picking the subsets that correspond to the vertices in the dominating set will form a set cover in our instance of set cover. In the other direction suppose you start off with a yes instance of set covers so, suppose this family has a sub collection of k sets which are such that they cover the entire universe. Notice that each set in our family is defined based on some vertex.

So, you could think of the vertex associated with any set in our family and let us say that we look at the subfamily corresponding to the set cover then that naturally leads us to a subset of vertices. These are the vertices that are represented by these sets. The vertices those were responsible for the creation of these sets in the first place. So, there are these k vertices and I am going to propose this as a dominating set of size k.

Suppose this is not a dominating set, what this means is that some vertex in the graph is not dominated by this subset of vertices. But now let us think about the element corresponding to this vertex that is left out that is not getting dominated. Notice that this element cannot possibly be covered by the claimed set cover because if it was covered then again it belongs to the closed neighbourhood of some vertex that we picked in the dominating set.

So, really this is a fairly transparent connection between these problems and I hope that the proof of equivalence is clear and it should also be clear that the construction works in polynomial time. Notice that all we have to do to build the set cover instance is enumerate all the closed neighbourhoods of all the vertices in G. So, this is certainly a polynomial amount of work that needs to be done.

And also, the parameter transforms safely because it goes from k to k. So, we have yet another valid parameterized reduction from dominating set to set cover. Showing that; set cover is at least as hard as dominating set and in turn at least as hard as clique. So, that brings us to the end of the first module for this week and at this point we have seen a bunch of examples of parameterized reductions starting with the assumption that clique is a hard problem.

And we have seen based on this alone that clique is also hard when restricted to regular graphs and so is independent set. We have also seen the hardness of a popular variant of clique which we called multi-coloured clique. And also multi-coloured independent set and we used the hardness of independent set on regular graphs to establish the hardness of partial vertex cover parameterized by solution size.

We also use the hardness of multi-coloured independent set to establish the hardness of dominating set and finally we use the hardness of dominating set to establish the hardness of set cover. So, I hope you enjoyed this little collection of examples of reductions. And what is coming up next is a brief look at the terminology that we associate with parameterized classes in particular we will talk a little bit about the w hierarchy.

We will not get into a lot of technical detail but we will just learn enough terminology to be able to recognize what is being said when you read papers that talk about parameterized intractability. And after that we will come back and look at a couple of more examples of reductions in particular, I want to show you at least one reduction that is a valid FPT reduction but would actually not count as a valid polynomial time reduction.

Because it actually leverages the full power of the definition of a parameterized reduction and it takes FPT time and in a way that is not polynomial time. So, it would not work in the classical setting but it is an interesting example of a parameterized reduction. And I want to show you one other example that involves working with structural parameters. So, we will be looking at a problem called list colouring parameterized by treewidth.

So, that is what is coming up in the rest of the discussions that we are going to have this week. So, please stay tuned and I will see you there.