PROFESSOR: We've briefly looked at graph isomorphism in the context of digraphs. And it comes up in even more fundamental way really for simple graphs where the definition is a bit simpler. So let's just look at this graph abstraction idea and how isomorphism connects with it.

This is an example of two different ways of drawing the same graph. That is here's a 257, and there's 257. It's connected directly to 122, as here. And also 257 is connected to 99, as here. And if you check, it's exactly the same six vertices and exactly the same eight edges. But they're just drawn differently.

So we don't want to confuse a drawing of a graph, like these two, with the graph itself. The graph itself consists of just the set of nodes and the set of edges. And if you extracted that from these two diagrams, you would get the same set of nodes and the same set of edges.

So same graph, different layouts. But here's a case where it's really the same layout. You can see these two pictures, if you ignore the labels, are exactly the same with the two grays and the two grays and the red and the red. The difference now is that I've renamed the vertices.

So we've assigned different labels to those vertices. And the connection between the two graphs now, this graph with vertices which are integers and this graph with vertices that are the names of people, is that they are isomorphic. And what isomorphism means is that all that matters between two graphs are their connections. And so graphs with the same connections among the same number of vertices are said to be isomorphic.

To say it more precisely, two graphs are isomorphic when there's an edge preserving matching between their vertices. Matching meaning byjection junction between their vertices. And edge preserving means that where there is an edge on one side there's an edge between the corresponding vertices on the other side. Let's look at an example.

Here are two graphs. And I claim that they are isomorphic. On the left, we've got a bunch of animals, dog, pig, cow, cat. And on the right we have a bunch of animal foods, hey, corn, beef, tuna. And it's a hint on how we're going to do the matching.

So I'm going to tell you that the dog vertex on the left corresponds to the beef vertex on the right. So I'm defining a function, a byjection, from the vertices on the left in blue to the vertices on the right in red. And f of dog is beef.

Likewise, f of cat, cats eat tuna. I'm going to map cat to tuna. And continuing for the remaining two vertices, I'm going to map cow to hay, which is what they eat, and pig to corn, which is frequently what's fed to pigs. OK, so this is a byjection. I mean, it's a perfect correspondence between the four vertices on the left and the four vertices on the right. But I have to check now that the edges are preserved.

What does that mean? Well, let's do an example. There's an edge on the left between dog and pig. That means that there should be an edge on the right between where they go to. So there ought to be an edge between beef and corn, because that's where dog and pig go. And indeed, there's an edge there. So that part's good.

And you can check the others. The other thing that we have to check on the left is since the edge preserving is an if and only if, there's an edge on the right if and only if there's an edge on the left, that's the same as saying there's no edge on the left if and only if there's no edge on the right. So let's check non-edges on the left.

There's no edge between cow and pig. And indeed, cow goes to hay, and pig goes to corn. And sure enough, there is no edge on the right between hay and corn. And you can check the remaining cases. These two graphs are isomorphic. And that function f is in fact the edge preserving byjection.

So stating it again, an isomorphism between two graphs G1 and G2 is a byjection between the vertices V1 of G1 and the vertices V2 of G2 with the property that there's an edge uv in G1, an E1 edge, if and only if f of u f of v is an edge in the second graph in E2. And it's an if and only if that's edge preserving. So if there's an edge here, there's an edge there. If there's no edge on the left, there's no edge on the right. And that's a definition that's worth remembering.

It's basically the same as the digraph case. Except in the diagram case, the edges have a direction. So it would be an edge from u to v if and only if there is an edge from f of u to f of v. But since we don't have to worry about direction in the simple case, the definition gets slightly simpler.

What about non-isomorphism? How do you show that two graphs are not isomorphic? I can show you the two graphs are isomorphic by simply telling you what the byjection between their vertices is. And then it becomes a simple matter of checking whether the edges that should be there are there are not.

How do you figure out the two graphs are not isomorphic and that there isn't any byjection that edge preserves edges? Well, for a start, these both have four vertices, so it's perfect. There are lots of byjections between the four vertices on the left and the four vertices on the right. Why isn't there an edge preserving one?

Well, if you look at the graph on the left, it's actually got two vertices of degree 2 marked in red here. There's a degree 2 vertex. There's a degree 2 vertex. And on the right, every vertex is degree 3, if you check.

Now one of the things that properties of isomorphism is that the edges that come out of the red, these two edges, have to correspond to two edges that come out of wherever it's mapped to. So a degree 2 vertex can only map to a degree 2 vertex. There aren't any. That's a proof that there can't be an isomorphism between the two graphs.

So in general, the idea is that we're looking at properties that are preserved by isomorphism. This is almost like a state machine invariant kind of idea. So a property is preserved by isomorphism.

Means that if two graphs-- if graph one has the property and graph one is isomorphic to graph two, then graph two has the property. And clearly if there's a property that's preserved by isomorphism and one graph has it and the other graph doesn't have it, that's a proof that they can't be isomorphic.

So what are some of these properties that are preserved by isomorphism? Well, the number of nodes. Clearly there's got to be a byjection, so they have to have the same number of nodes. They have to have the same number of edges for similar reasons. Because the edges are preserved. An edge on one side corresponds to an edge on the other side.

Others things that matter is we've just made this argument that the degrees are preserved as a consequence of the preserving of the edges. And all sorts of other structural properties are going to be preserved by isomorphism, like for example, the existence of circular paths, and distances between vertices, and things like that. Those will all be properties that are preserved by isomorphism.

So that gives you a hook on trying to figure out whether or not two graphs are or are not isomorphic. But in general, there will be, if you've got a graph with a few 100 or 1,000 vertices, there are an awful lot of potential byjections between them to check. And the question is, how

do you do it? It's a huge search that can't really be effectively done exhaustively.

So what you look for is properties that are preserved by isomorphisms that give you a guide. So for example, if the graph on the left happens to have a degree 4 vertex and that degree 4 vertex is adjacent to a degree 3 vertex, then the adjacency of a degree 4 and a degree 3 is a typical property that's preserved by isomorphism.

So you know for sure that if there's going to be a byjection between the first graph and the second graph, this pair of adjacent vertices of degree 4 and degree 3 can only map to another pair of adjacent vertices in the second graph that also have degrees 4 and 3. So that will cut down enormously the number of places that this given vertex can map to in the other graph. And it gives you some structure to use to try to narrow down the search for the number of isomorphisms, and where the isomorphism is, and whether or not it exists.

So having a degree 4 adjacent to a degree 3, for example, is a typical property that's preserved under isomorphism. But even so, if I give you two very large graphs, and these are actually extracted graphs from some communication network, an image of them, it's very hard to tell whether or not they're isomorphic. Well, you could guess, because of course, we took the same picture and copied it twice.

But if there was some subtle difference between these two, like I erased one edge somewhere in the middle of that mess, how would you figure out that the two graphs were not isomorphic in that case? And the answer is that like these NP complete problems, there is no known procedure to check whether or not two graphs are isomorphic that is guaranteed to be efficient and to run in polynomial time.

On the other hand, there are technical reasons, there are technical properties, that says that graph isomorphism is not one of these NP complete problems, unless [? peoples ?] NP or something like that. And so that's one distinguishing characteristic of this problem. The important one is that, as a matter of fact, in practice there are some really good isomorphism programs around that will in many cases figure out, given two graphs, whether or not they are isomorphic in time that's approximately the size of the two graphs.

So pragmatically, graph isomorphism seems to be a manageable problem. Although theoretically you can't be sure that these efficient procedures that work most of the time are going to work always. Well, known procedures in fact blow up exponentially on some example or another.