[SQUEAKING]

[RUSTLING]

[CLICKING]

ZACHARY ABEL: Good morning, good morning. Afternoon, I guess, technically. Welcome to lecture 15 of 6.1200-- 6.120. Again, I still don't know how to call it.

Today we are talking about relations and counting. And there's a bunch of scary stuff on the board. Ignore that. Wonderful. So first thing we're talking about is relations. And relations are a very useful tool, very useful term, generalizing the idea of a function. So let's see what we mean by that.

Let's see. So-- yes. So a relation R is a subset of A cross B. A and B are sets. They can be any sets you want. R is going to be a subset of pairs, where the first element is taken from A, the second element is taken from B. It might be all the pairs, none of the pairs, some subset of the pairs.

All we know is that we need all three of these pieces of data. A relation consists of, first of all, a domain A, which is any set, a codomain B, which is a set, and a subset R, subset of A cross B.

So you need to know what is your domain, what is your codomain, and which pairs are part of your relation. This generalizes the idea of a function because you can think of a function as you have your inputs, you have your outputs, and your function specifies which inputs map to which outputs by just encoding them in a pair like this.

Let me give a more concrete example. Here we go. So imagine we have these three sets. We have the set of students that are Luke, Geralt, Quentin, and Willow. And we have some classes that these magic students might take in a magic school, I guess-- chemistry, sports, literature, and 6.1200. Totally makes sense.

And we have some magic profs-- we have Galadriel, Zelda, Eda, and Doctor Strange. So we have some sets. We might specify a relation, let's call it the learning relation, L, which is going to be a subset of students across classes. So it's going to be all pairs where you have a student who's taking a class.

And in this case, it looks like we have these students taking these classes. It looks like Luke is taking chemistry. Luke is also taking literature. Geralt is taking chemistry and sports and literature, et cetera.

So we need all three of these pieces of information-- the domain, the codomain, and the set of pairs in order to specify this relation. Likewise, here's another relation-- which classes are being taught by which professors. So this is a subset of classes times professors.

And it's saying that this class chemistry is being taught by this professor, Eda. Sports is being taught by Strange, lit by Galadriel, and 6.1200 is being taught by Zach-- I mean Zelda.

[SOFT LAUGHTER]

Thank you. This might look pretty familiar. This is almost how we define directed graphs, right? You have some set of vertices or whatnot, and you have some edges between them. This is a very closely related concept.

And just like with graphs, that representation isn't the most useful way to represent it. It's kind of hard to see who's taking what and visualize and think about it. So let's draw it instead.

So we already said here we have the learning relation, L. We said Luke is taking chemistry. And I'm going to draw it with a directed arrow. Luke is also taking literature. Geralt is taking chemistry and sports. Geralt is taking also literature.

Then Quentin is taking sports and lit. Quentin is taking sports and lit. And Willow is taking lit. And apparently no one is taking 6.1200 at this magic school-- shame for them.

But this is a very clear representation in my mind. But again, it's representing the same information. And we can see that, in fact, this is exactly a directed graph where the vertices are, what is it, students union classes. Those are allowed vertices.

And it just has the special form where each one of our directed arrows always goes from a student to a class. But otherwise, it's just a directed graph. Does that make sense? Let's quickly draw the other one.

So we have chemistry being taught by Eda. We have sports being taught by Strange, lit by Galadriel, and 6.1200 by Zelda. So this is the teaching relation. Are we clear on what these examples are? Wonderful.

So for notation, there are lots of different notations that are used for relations because relations are used in lots of different ways. But here are some common notations. So if R is a relation from A to B, then we might write a relates to b to mean that the pair a comma b is one of the pairs in our relation, one of the directed edges in our graph, because, like I said, the relation is really about specifying which pairs relate and which pairs don't.

It's a yes/no question. Some of the pairs are yes. Some of the pairs are no. And so we might write a relates to b with this infix notation to say that a relates to b. We're saying the same thing, just with different notation.

And this might remind you of things like 3 is less than 7. A set S is a subset of a set T. This less-than relation, this subset equal relation, these are relations.

And this general notion of relation is a way to generalize concepts like this. And so this is why we sometimes have our infix notation like that, so we can think about whatever other relation we're talking about similar to how we think about these more familiar ones. All right.

We can also think of R as a predicate. We can say R of a comma b. So if R is a predicate that takes two arguments and spits out true or false, this is another way you can think of specifying a subset of pairs.

So R a, b is true when a, b is part of our relation, and it's false when it's not. And phrase-wise, we might say that a relates to b. Again, means the same thing-- means that a, b is in R.

At its core, we are just trying to give one bit of information for each pair. Either this pair relates, or this pair doesn't. This pair is in our set or it's not. This pair satisfies our predicate or it doesn't. All different ways of saying the same thing.

Does that make sense? Wonderful. And by the way, when we say "relation" here, specifically we're talking about binary relation. Yeah, that should say a binary relation.

But all the relations we're going to talk about today are binary. But it's the same relation as in relational database. You can imagine that this kind of thing is exactly how a database stores its information.

In this case, we're only talking about databases with two columns. So learning has a column for student and a column for class. And there's a row every time a student is taking a class. That's all this is. It's a database.

Yes. So I said earlier that-- here, let's go back up here-- that relations are a generalization of functions. And let's see what we mean by that.

So definition-- relation R from A to B is a function precisely when every element on the left side relates to, at most, one element on the right side. So if everything in the left set appears on the left side of at most one pair, it's the source of at most one arrow.

So another way to say that is every a in A has at most one arrow out. All right, so these two examples, is this learning relation a function? Who can tell me why?

AUDIENCE: One of the students has multiple arrows going out.

ZACHARY ABEL: Yeah, we have a student with multiple arrows going out. That's precisely what we need to avoid. Its function says each node on the left is supposed to have at most one arrow going out, and this fails that.

Is this a function? Yeah. Every class has at most one arrow out, so it's a function. If I deleted this arrow, so now we only have these three arrows and no one's teaching 6.1200, is this still a function? Yeah, it's fine-- at most one arrow, not exactly one arrow. So that's still fine even if that arrow is missing. But I'm going to put it back because that's our example.

Let's see. When R is a function, we'll write R of a equals b to mean-- means that a, b is in the relation. And so R of a is the element that a points to if it exists.

R of a might be undefined if the function doesn't have any arrow out at this point. But if there's only one, we can just call it R of a. This is the usual function notation that we're used to.

Let's make that even more precise. Let's look at the function f from R to R. By the way, when it's a function, we can use this more specific notation, f is a function from R to R. We use the arrow instead of f is a subset of R cross R.

So let's define the function where f of x is 1 over x squared, the kind of function you might see in a precalculus class or something. Everyone agrees this is a function? Well, I mean, kind of.

How is it a function by our definition of relation? We're supposed to specify a domain-- check-- a codomain-- check-- and a set of pairs. So what are the set of pairs?

In this case, the set of pairs is going to be x comma y in R cross R such that y equals 1 over x squared. It's just the set of pairs that satisfy our relation. Another way to write this-- x comma 1 over x squared, where x is a real number without 0 because we're not allowed to divide by 0.

All right, so f is a function. Here's a way to write it in terms of the pairs like we've said. And it's totally fine that f of 0 isn't defined because, again, you can't divide by 0. f says each input has at most one output. And it just so happens that the input 0 has no outputs, which is fine.

Sometimes you do want everything to have-- every input to have an output. So we have a different term for that. We'll say that a relation is total if and only if every a in A has at least one arrow out.

In other words, for every a on the left side, there's at least one b on the right side that it relates to. Everything on the left has some arrow going out, possibly multiple. That's what we call total.

Are our examples here total? Is this one total. Yeah, everything on the left has one arrow out-- has at least one arrow out. Here, everything on the left has at least one arrow out. Excellent. Is this function total?

AUDIENCE: No.

ZACHARY ABEL: No? Why not?

AUDIENCE: 0 has no arrow out.

ZACHARY ABEL: 0 has no arrow out. f of 0 doesn't exist. There is nothing that 0 relates to on the right. OK, let's give a different example.

Let's define g from the set of R without 0 to the set of R without 0-- this is supposed to be a set-- where we're going to define g of x is 1 over x squared. All right, same function, right? Same set of pairs. Is this function g total?

I saw exactly two nods and a bunch of blank stares. That is all right. Someone who's-- oh, a couple of more nods. Someone who's nodding, can you tell me why g is total? Yes.

AUDIENCE: It's total because, like in the previous example, 0 [INAUDIBLE] arrow out because 0 isn't an option. [INAUDIBLE]

ZACHARY ABEL: Right. So they were saying, since 0 is no longer an option as an input, all of the inputs that we're allowed to choose do have a valid output assigned to them. So this function g is total.

This function f is not total, even though it's exactly the same set of pairs for both f and g. It's exactly the same set of real comma real, or x comma 1 over x squared, for all the nonzero values of x. All we've done is change the domain and codomain.

And this goes to show that you need all three parts of that definition of relation. You need to know what set you're mapping from, what set you're mapping to, and what all of your related pairs are. All three of those pieces of information are important to determine things like whether it's a function, whether it's total. Does that make sense? Awesome.

Now there's-- oh, by the way-- oh, also, these often go together. So a total function means it has-- so total means everything on the left has at least one arrow out. Function means everything on the left has at most one arrow out. So if you put those together, total function has exactly one arrow out for each a in A.

And secretly, when mathematicians say "function," they usually mean total function. Usually, the fact that some inputs don't have outputs, that's supposed to be a special case. And the default when we say function is that everything on the left has a definite output on the right.

So when mathematicians say "function," we usually mean total function. When we're not assuming that a function is total, mathematicians will often say "partial function." So partial function means a function that isn't necessarily total. It might be total.

Confusingly, a total function is an example of a partial function. Partial does not mean not total. Partial does not mean that it's definitely missing inputs. Partial just means that you're not assuming there aren't missing inputs. So by our definitions that we've set out today, partial function just means function. And total function means function that has nothing missing, no inputs missing.

From now on in this class, whenever we use the word "function," we're going to try to specify whether we mean a total function or whether we're allowed to have some missing inputs because usually when we say "function," we're going to mean total function. And so let's decide from now on in this class, at least when we give problem statements to you, that the word "function" on its own is hereby ambiguous.

Whenever we say "function," we should really be telling you, do we mean total function? Or do we mean function that may or may not be total? So if you see it on a problem set or test and it's not qualified like that, please ask. We want to clarify that for you. We're not trying to be tricky with edge cases of this definition. If you see "function" on its own without qualifiers, please ask. Yes?

AUDIENCE: On top, [INAUDIBLE] like-- it says that the relationship is total if every a in A has at least one. On the bottom, it's total if it has exactly one arrow.

ZACHARY ABEL: So a relation is total if everything on the left has at least one arrow out. A relation is a total function--

AUDIENCE: Oh.

ZACHARY ABEL: --if its total and also a function, which means that everything on the left has exactly one arrow out. Great question. All right.

Yeah, sorry that partial function does not mean function that isn't total, even though that's what it sounds like it should be. Partial function just means function that may or may not be total. Naming things is hard. Or rather, using bad names that someone else has already defined for us is hard. I would change it if I could.

Anyway-- (SING-SONGY) do, do, do, do. Great. We have similar terminology for arrows in instead of arrows out. So far, we've looked at total and function, which are talking about how many arrows are coming out of the left side. We can also ask how many arrows are coming into the right side.

So we'll say that R, which is a relation between A and B, is injective if and only if every element on the right side has at most one arrow in. And we'll call it surjective if and only if every b in B has at least one arrow in.

So injective means you never hit the same thing twice. Surjective means you hit everything possibly more than once. Back to these examples, is L injective? In other words, does everything on the right side have at most one arrow in? No. Thumbs down. Thank you.

For example, chemistry has multiple arrows in. That alone is enough to show that it's not injective. Is it surjective? Also no. Another thumbs down. Thank you.

It's not surjective because 6.1200 doesn't have any arrows in. Surjective means you're supposed to hit everything on the right, and we don't. So this is neither surjective nor injective, whereas this is both.

Everything on the right has at least one arrow in-- so it's subjective-- and at most one arrow in. So it's injective. Now, one of the reasons we're defining all of these similar confusing-sounding terms-- apologies for that-- is that we have some theorems that these terms are enough to help us relate the sizes of various sets.

So we have our first theorem of the day. If A and B are finite sets and R, a relation from A to B, is a total injection-- so if we have finite sets and a total injection from the first to the second-- then the size of A is at most the size of B.

If we can find a total injection from one set to another set, then the set on the left is smaller or at most the same size. And let's draw a picture to see why. So total injection means that everything on the left has at least one arrow out. Total means that.

And injection means we never hit the same thing on the right side twice. So there's always an arrow out on the left side. And there's at most one arrow out on the right side.

And so all this is saying is that the number of elements on the left is at most the number of arrows because everything on the left has at least one arrow out. And the number of elements on the right side is at least the number of arrows because everything on the right has at most one arrow out. So that's why this is true.

Size of A is less or equal to the number of arrows, which is less or equal to the size of B. So often, when you want to show that one set is smaller than another set, you just define a total injection between them, and that's your proof.

Similarly, on the other side of things, theorem 2-- let's see-- if, again, A and B are finite sets and R is a surjective function, then the size of A is greater or equal to the size of B. And I claim that's just exactly this picture but in the opposite direction.

Instead of total meaning everything on the left has an arrow out, we have surjective, which means everything on the right has an arrow in. And instead of injection, which means everything on the right, has at most one out-- or sorry, at most one in, we have function, which says everything on the left has at most one arrow out.

So we've just swapped the roles of in and out. That's really all that happened to translate this theorem to that theorem. And by our powers combined, definition-- if R is injective, surjective, total, and a function, if it has all four of these properties, then we say that R is a bijection.

So that's the definition of bijection. Bijection means you have all four of these nice properties. To summarize, everything on the left has exactly one arrow out. Everything on the right has exactly one arrow in. And theorem-- if R from A to B is a bijection, then the size of A equals the size of B.

All we're doing is applying both of these theorems, and we get both inequalities. So the sets have the same size. All right, so bijection is the main tool that we're going to be using in the second half of this lecture to compare the sizes of sets. Questions about that? Wonderful.

So next, I want to talk about a couple special kinds of relations. So first of all, I never said that the sets A and B have to be disjoint or even distinct. So let's look at the special case where A equals B. So R is a subset of A cross A. We call this a binary relation on A.

And this, I claim, is basically exactly how we define directed graphs. You have a set of nodes, which in this case is A and serves as both our domain and codomain. And you have a set of directed edges between them, and that's it.

That's how we define directed graph. So a relation on a is basically identical to a directed graph whose vertices are A, which means that all of the directed graph examples we've already talked about, like who follows whom on Twitter, like reachability in Google Maps, all of those digraphs are also relations. They're relations on the set of Twitter users or locations in Google Maps.

Another good example that I really like of a relation on a set is the relation of who likes whom, not like romantically or anything, just as a person, who likes whom. Fix a moment in time. Everyone measures who they like among all the other people in the world. Bam, you have your relation.

And the reason I like it is that it has tons and tons of chaotic structure. For example, if I like you, that doesn't mean that you like me back necessarily, especially this week when the problem set's a bit long. Apologies. If I like you and you like them, that doesn't mean that I like them. Not always, right?

That first example, I like you means you like me, that's what we're going to call symmetry in a little bit. That second example, "if I like you, you like them, that means I like them," we call that transitivity. So this graph is neither symmetric nor transitive.

It's also not reflexive. There are days when I don't like myself, so I wouldn't have a self-loop in this graph. Now, I don't mean that in like a "cry for help" kind of way. But you know, some days you're a little down on yourself. It can happen. It's OK. If it happens too much, you have support. Please seek it out.

[SOFT LAUGHTER]

So funny story. I gave this same example a couple years ago. I got the same half-hearted chuckle that we got just now. And then someone went home and wrote about it on "MIT Confessions."

[LAUGHTER]

I don't remember exactly what they said, but they maybe left off some of the context. And then, for the next couple days, I was getting multiple Facebook messages from multiple friends, 'Zach, are you OK?' 'I know we haven't caught up in a while, but I'm here to chat. And I'd love to talk to if you're available. I hope you're doing well.' And I just want to say I felt loved.

[LAUGHTER]

So thank you to whoever did that. But maybe let's not have a repeat.

[LAUGHTER]

Anyway, back to graphs. So one of the main reasons we want to look at relations on a set is because we already have lots of examples, and especially lots of examples that we don't necessarily think of as graphs.

For example, A equals B is a relation that we use all the time. A is congruent to B mod 10 is a relation on the set of natural numbers, of integers that we use all the time and don't necessarily think of as a graph. But it's definitely a relation. It's also a graph because that's secretly the same thing. But the way you think about it can change if you want to.

Likewise, A less equal B. We have set A is a subset of set B. We have A divides B. Lots of examples of relations that we want to be able to talk about.

Another good example is reachability. So if G is a digraph. And if V and W are nodes of our graph, then we can define the walk relation. Sorry, that was out of order.

If G is a digraph, we can define the walk relation on the set of vertices of our graph, where we define that V relates to W under this walk relation that we're going to call G prime-- or sorry, G star if and only if there exists a walk from V to W. That's a relation we can define on a digraph, and it's a really useful one.

One thing I want to warn you about, we started with a graph. We defined a relation, G star. We know that relations on this set of vertices can be thought of as a graph on that set of vertices. We're not necessarily talking about the same graph.

This relation, G star, is something we constructed from this digraph G. But the graph we constructed isn't necessarily the same as the graph G that we started with. So it can be a little confusing thinking of, oh, that's a graph, and that's also a graph that was constructed from this other.

It's fun to think about. But just a warning, even though we're talking about two different graphs on the same set of vertices, they might actually be two different graphs. Cool.

So there are two special kinds of relations on a set that are especially useful because we have lots of examples. And it's useful to be able to generalize those examples. The first one I want to talk about is called an equivalence relation.

And some of the examples I wrote down, for example a equals b or a is congruent to b mod 10, these are somehow capturing this idea of sameness. a and b are the same. a and b have the same remainder, mod 10.

If we're talking about connectivity in an undirected graph, we say that two nodes are connected with each other, meaning there's a walk from one to the other, which means that they belong to the same connected component. There's another idea of sameness there.

So an equivalence relation is an idea that generalizes this idea of sameness. And so let's see if we can do that. What does it mean to somehow encapsulate that the two things are somehow equivalent to each other, even if they're not identical?

And it turns out there are three easy-- by "easy," I mean possibly confusing, but I apologize-- three easily stated properties that a relation might have. And if it has all three, then I claim it captures everything it needs to capture about this idea of sameness.

Everything I've just said is extremely vague. Let's make it precise. So let's say that R is a relation on A. We're going to say R is reflexive. This means that for all a in A, a relates to itself.

Reflexive means everything relates to itself. Everything should be the same as itself if sameness means anything. So this is definitely something we would want to have.

In terms of a digraph, it means that every vertex has a self-loop. All right, next one, R is symmetric if and only if for all a and b in our set, a relates to b if and only if b relates to a. So symmetry is saying that order doesn't matter. If I'm your friend, then you're my friend-- so Facebook, not Twitter.

Cool. In terms of digraph, remember a relation is always a digraph. This means that every time we have an edge in one direction, we also have the edge in the other direction. All right, so symmetric means order doesn't matter.

And finally, R is transitive if and only if for all a, b, and c in A, if a relates to b and b relates to c, then a relates to c. So if this is the same as that, and that's the same as this other thing, then this should be the same as this other thing-- another thing that we might want to be true if we're trying to capture what we mean by sameness.

In terms of graphs, it means that any time we have a walk of length 2, then there was already a directed edge from the start to the end. If you can get from here to there and from there to this other place, then you can get straight from here to this other place.

And both of these examples that I mentioned-- all three of these examples-- same connected component. So these circled relations, I claim, all have all three of these properties. Every number is congruent to itself mod 10.

If two numbers are congruent to each other mod 10, then they're congruent the other way mod 10. And if a and b are congruent and b and c are congruent, then a and c are congruent mod 10. And you can check that all of those examples have this property. And in fact, I claim that these three properties are enough.

Unfortunately, I have to put the definition very far away from-- or sorry, the theorem very far away from that definition. We'll make it work. Bye, eraser

OK, definition-- R is an equivalence relation-- relation-- if and only if R is all three of those things-- symmetric-- sorry, reflexive, symmetric, and transitive. And we have a theorem that backs up my claim that says this is all we need for sameness.

Theorem-- if R, which is a relation on A, is an equivalence relation-- if R is an equivalence relation, then R partitions A into some collection of subsets, A1 union A2 union A3 union, and so on. And this is going to be a disjoint union such that every a in A belongs to exactly one of our subsets.

That's what it means to be a partition, that we're breaking A into smaller pieces, where every element of A is in exactly one of our pieces such that a relates to b if and only if a and b are in the same subset.

So any time we have an equivalence relation that satisfies these three axioms, then that exactly forms a partition which tells us precisely what our relation is. Our relation just says you're related if and only if you're in the same subset.

For example, if we're talking about congruence mod 10, the parts of our partition are, well, we have the set 0, 10, minus 10, 20, minus 20. This is one part of a partition. Everything in the set is congruent to everything else in the set and not congruent to anything outside the set.

Then, we also have 1 and 11 and negative 9 and 21 and negative 19. Here's another part of our partition. All of these numbers are congruent to each other and congruent to none of the other numbers.

And in total, there are 10 of these parts. So anytime we have an equivalence relation, we can partition our set into subsets. That exactly tells us what our relation is. Yes?

AUDIENCE: What's the union with the dot?

ZACHARY ABEL: Yeah, so the union with the dot here, the dot is there to emphasize that we're taking the union of sets that are disjoint, that no element is in multiple of these sets. I know I wrote it out here in words. But I like to emphasize it in the formula as well.

When we're talking about connectivity in an undirected graph, being connected to each other, a pair of vertices are connected if and only if they're in the same connected component. And so the connected components form the pieces of our partition.

So we've seen multiple examples of this. This is just making this more general and more abstract. Any time we satisfy all three of those axioms, we get this.

All right. so the other kind of subset I want to talk about-- actually, before that, we have a couple more examples. We had this example of a less equal b, a divides b, set A is a subset of set B. These are other common relations that we work with. And these are not equivalence relations.

We know that 2 divides 10. But 10 doesn't divide 2. So it's not symmetric like we wanted with equivalence relations. But in many ways, a divides b behaves a lot like a less equal b or "a" subset of b. There's some commonality there. And that commonality we're going to define as what are called weak partial orders.

So weak partial order. Our goal is going to be, behaves like a less or equal b, A is a subset of B, a divides b, et cetera. So let's see if we can capture what these all have in common and prove something about them.

So again-- OK, so let's see. R is a weak partial order-- weak partial order-- if and only if R is-- are these all reflexive? Is a number always less or equal to itself? Does it divide itself? Is a set a subset of itself? Yes. So these are all reflexive.

So let's add that to our desired definition for weak partial order. We already said it's not symmetric. So we're probably going to have to replace that somehow. Are these transitive? If a less equal b and b less equal c, is a less equal c? Yeah, same for subset. Same for divides. So let's put that.

So this is looking very similar to the definition of equivalence relation. We're going to make one tweak. Instead of requiring it to be symmetric, we're going to require it to be antisymmetric, which does not mean not symmetric. It's a different concept. Let's define that.

So R is antisymmetric means for all a and b, if a relates to b and b relates to a, then a equals b. So the only way that you relate in both directions is if you're comparing yourself to yourself. That is certainly true for less equal.

If we know that a is less equal b and a is greater or equal b, then a equals b. That's true for a subset, as well. If we have two sets and each of them is a subset of the other, then they're exactly the same set.

And so this is going to be our axiom. If r is reflexive, antisymmetric like that, and transitive, we're going to define that to be a weak partial order. And that's going to be our notion of behaving like these familiar examples.

And so why is a weak partial order useful? What can we prove about them? What can we do with them? Well, just like equivalence relations had a different characterization in terms of partitions of a set, we can prove that equivalence relations-- sorry, prove that weak partial orders are related to DAGs.

Claim-- yes. If G is a digraph, then the walk relation G star is a weak partial order if and only if G is a DAG. So the point of weak partial order is that we can put things in somewhat increasing order of size.

If a is less than b-- less equal b, less equal c, less equal d, less equal e, what we want to avoid is coming back around and e is less equal a because, as it turns out, that's going to break some of these axioms. And so all we're saying is that satisfying these three axioms is equivalent to being the walk relation on a DAG. And DAGs were defined by not having directed cycles.

All right, we're not going to prove this, but it's true. You can come ask me about it later if you'd like. Last definition for now-- once again, I apologize. There are lots of definitions today.

But I want to bring our attention to this word "partial." A weak partial order, just like we had partial function that might be missing values, a weak partial order might be missing comparisons. For example, the set 2, 1 and the set 2, 7, is one a subset of the other? No, in neither direction is that true.

So here are two sets, and neither is related to the other one under this subset relation. So this is a pair that cannot be compared. So if R is a weak partial order, we'll say that a and b are comparable if and only if the relation goes in at least one of the two directions, a relates to b or b relates to a. Otherwise, they're incomparable.

So these two sets, 2, 1 and 2, 7, are incomparable under this subset equal relation. All right? And now, finally, a weak partial order is called a total order if and only if every pair is comparable.

So whenever you have two elements, you can always put them in order. One of them is less or equal to the other one. So that's certainly true for the less equal relation. It's not true for the subset relation. And it's not true for the divides relation.

2 and 3 are incomparable under the divides relation. But if every pair can be compared, then it's called a total order. And if you have a total order, then really all you have is just an entire line of them organized by size. A total order is the most structure you can have. It's just everything is ordered, and there's no ambiguity.

All right, so that is everything I wanted to say on the relations side of things today. Now let's move on to the other part of today's lecture, which is counting. And I don't mean counting like 1, 2, 3, 4, 5. I assume all of us are capable of counting at least to 300 or so.

No, but by "counting," I mean counting the size a set. If you want to know-- like, here's a comically large deck of playing cards. How many different ways are there to shuffle them?

How many different orderings are there for the 52 cards in the deck? Here's one of those orderings. Here's a different one, et cetera. How many orderings are there for this deck of cards? Does anyone know? Yes?

AUDIENCE: 52 factorial.

ZACHARY ABEL: 52 factorial, absolutely. So the number of orders for a deck of cards is 52 factorial. And we'll see why later in this course, or later in this lecture in fact.

How many trees are there whose vertices are exactly the set 1, 2, 3, up to n? Does anyone know? I'd honestly be surprised. It's not an easy problem. But this is the kind of problem that we're going to try to be able to answer and we're going to look at techniques for trying to answer.

In this case, the number of trees on that set of vertices is n to the n minus 2, which I love. I find that kind of magical. There are lots of cool ways to prove it. We're not going to prove it together, but come ask me if you're interested.

But these are the kinds of questions we're going to try to ask. Here's a set of things. How many things are there? And to get us a little more in the mood for counting, I have a short story I want to tell you, the parable of the two shepherds. No one calls it that. That's just me.

So this first shepherd-- this first shepherd, every morning, has to let the sheep out to graze every evening, bring the sheep back in to protect them from wolves, I guess. I don't know what shepherds do. But unfortunately, the shepherd does not know how to count grade-school style, 1, 2, 3. The shepherd doesn't have words for numbers above 3.

So how do they make sure that all of their 50-something sheep go out in the morning and come back in the evening if they can't count the sheep? And a classic answer to this, which may or may not have historical merit, but all mathematicians who have studied counting have heard this story at least once, the classic example is every time you let a sheep out of the pen, you put a pebble in your pocket. Another sheep, another pebble. Another sheep, another pebble.

At the end of the day, when you're bringing the sheep back in, every time a sheep comes in, you take a pebble out of your pocket. And when all the sheep have come in, that's the same moment that all the pebbles are out of your pocket. So when your pocket is empty, you know there are no more sheep. And vice versa, if there are pebbles left in your pocket, there are still sheep out there, and you have to go find them.

If your pocket is empty and there are more sheep coming in, well, something weird happened and you probably stole from your neighbor. But the point is, for these next couple of lectures in our counting unit, we count with correspondences. We count with bijections, which we defined on one of these boards.

If we can set up a one-to-one correspondence, everything in this set corresponds to everything in this set and everyone has a partner, no one has two partners, no one is missing a partner, then there are the same number of things on both sides. What that shepherd is doing is basically making sure that there's always a correspondence between sheep that are out there and pebbles in their pocket.

They don't have to keep track of what that correspondence is. And in fact, it might change. If you initially put a pebble in your pocket for sheep Deborah. you're not keeping track of which pebble corresponds to sheep Deborah. So when Deborah comes back in, you might remove the pebble that was assigned to Sandy.

I don't know why I'm picking these names. But you don't care what that correspondence is, as long as there is a correspondence between pebbles in your pocket and sheep that are still grazing. So the moral here is we count with correspondences.

The second shepherd-- this second shepherd knows how to count. That's totally fine, not a problem. We have language. But the shepherd has an overeager apprentice.

This apprentice runs in one evening and says, master, I brought all 40 sheep into the pen. And the shepherd looks confused. What do you mean? We only have 38 sheep. Don't worry, master, I rounded them up.

[SOFT LAUGHTER]

Thank you. No moral. I just like bad puns.

[SOFT LAUGHTER]

I heard one groan. Thank you very much. That warms my heart. And now that we're in a great mood, let's talk about counting.

[LAUGHTER]

All right, so let's actually talk about counting. Here. First of all, why do we care? Why are these tools that we need in a computer science class? Why do we care about counting things?

Well, often when we're analyzing the runtime of an algorithm, we're doing some steps some number of times. And we need to be able to count how many times we do that step in order to bound the runtime of the algorithm and prove that it runs fast.

It's also useful for things like probability. Probability is built very heavily on top of counting. And all of our counting techniques will transfer over to probability techniques, as well. And that'll be useful after quiz 2, which is coming up next week by the way.

But let's learn-- where my chalk is. Found it. Let's learn some techniques for counting. The first one I want to mention is called the product rule, which says that for finite sets A and B, the size of A cross B equals the size of A times the size of B.

Remember, A cross B is the set of all possible pairs where the first element is in A and the second element is in B. So the number of ways to choose an element from A and then choose an element from B is the size of A times the size of B. And this generalizes, as well.

If we have n different sets and we're looking at ordered sequences where the first element comes from A1, the second element comes from A2, the third element from A3, and all the way down, the number of such sequences is just the size of A1 times the size of A2, all the way to the size of An.

So if we have to choose an element of A1 and then choose an element of A2 and then choose an element of A3 and all the way down, the number of ways we can make that choice in its entirety is size of A1 times size of A2 times size of A3 all the way down.

For example, the number of length n binary sequences-- so a binary sequence is a sequence of zeros and ones, just zeros and ones. And we're asking, how many length n binary sequences are there? Well, the set of length n binary sequences is 0, 1 times 0, 1 times 0, 1 n times.

This is precisely the set of binary sequences. This is what we mean by binary sequence. First element is a 0 or a 1. Second element is a 0 or a 1, all the way down.

And the generalized product rule-- or sorry, the general version of the product rule says that this set, the size of this set is just the size of 0, 1 times the size of 0, 1 all the way down, which is 2 times 2 times 2 times 2 n times. So there are 2 to the n binary sequences of length n. Does that make sense? Wonderful.

Next, let's look at the bijection rule. If there exists a bijection from A to B, then A and B have the same size. This is what that first shepherd was doing. If there exists a bijection between two sets, the sets have the same size because we're just pairing them up one to one. No one is repeated. No one is excluded.

And as an example of this, how many subsets of 1 through n are there? So if we're looking at all of the subsets of 1 through n of any length-- could be length 0, where we take none of the elements; length n, where we take all the elements; or any length in between, where we take some of the elements and not others-- how many subsets are there?

Well, let's use the bijection rule. I claim set of subsets of 1 through n is in bijection with Bn. And I'm going to define this. Bn is binary sequences of length n.

So the thing we counted in the last example, I claim that the set of subsets of 1 through n is in bijection with the set of those length n binary sequences. And to prove that, we can just define our function and verify that it's a bijection.

So given a binary sequence b1, b2, bn, where each of these is 0 or 1, I'm going to map that to a subset of 1 through n, i in 1 through n, such that bi equals 1. We send it to the subset of indices that have value 1.

So you can think of this as a set-- as a sequence of binary decisions. Do we keep element 1 or throw it away? Do we keep element 2 or throw it away? All the way down. For example, 0, 1; 0; 0, 1; 1, if n is 6, then this maps to the subset 2, 5, and 6 because those are the indices that have ones.

All right, and we're not going to do it now. But you can verify that this gives a bijection between the set Bn and the set of subsets of 1 through n. Claim-- this is a bijection. You'll see more examples in your homework of what details we would need to check to verify more carefully that this really is a bijection. But intuitively, they're encoding the same information.

Each number from 1 through n is either included or excluded. That's a binary decision. The i-th bit is either 1 or 0. That's a binary decision. So we're just corresponding those two things.

All right, next. We have the product rule, the bijection rule. Now we have the sum rule. Sum rule says that if A and B are disjoint, so they have no elements in common, then the size of A union B is the size of A plus the size of B.

So if I have five cards in my right hand and three cards in my left hand, and I don't have any cards in both hands, then I'm holding eight cards-- 5 plus 3. And more generally, if I'm taking the size of a disjoint union of n things, if all of these Ai's are disjoint with each other, then this is the size of A1 plus the size of A2 plus the size of An. And let's see that in action.

OK, so all of us who have not been using password managers all our lives have frequently been told, choose another password that has these specific requirements that has this length. It must use a special character. It has to do all this other stuff, which just makes it harder for me to remember it. Anyway, I'm going to play that role now.

How many passwords have length between 6 and 8 inclusive, start with a capital letter, and the rest are capital or lowercase letters and/or digits. The digits aren't capital or lowercase. They're just digits.

So how many passwords have length between 6 and 8, start with a capital letter, and then finish with whatever letters or digits you want? And with this, it's useful to break it up into cases based on the length of your password.

I claim that the set W-- I called it W in the notes for some reason, W for "password"-- I claim that W is the disjoint union of the set of passwords of length 6 that satisfy the other properties, set of passwords of length 7, and set of passwords of length 8. So if we can count each of these three subsets, then we add them up and get the size of W. And let's see if we can do that.

So what is the size of W6? How many passwords satisfy this property? Well, I claim the set W6 itself as a set is, well, first of all, the set of capital letters for the first entry times the set of letters or digits to the fifth for the other five entries.

This is exactly what this set is. The first entry has to be a capital letter. The other five characters can be anything from the set. So we have this product of six sets. And that brings us back to the regular product rule.

So by the product rule, the size of W6 is, well, how many capital letters are there? I think that's 26. And then how many letters and digits are there? 26, 26, and 10, so that's 50-- 62 to the fifth. We just use the regular product rule to get this.

And we can do the same thing for W7 and W8. W7 is going to be 26 times 62, this time to the sixth because there are six extra digits after the-- extra characters after the first one.

And the size of W8 is 26 times 62 to the seventh. So by the sum rule, since W is the disjoint union of these three different sets, the size of W is the size of W6 plus the size of W7 plus the size of W8, which is 26 times 62 to the fifth, plus 26 times 62 to the sixth, plus 26 times 62 to the seventh.

So we use the sum rule to break it up into three separate cases, and then use the product rule for each of those cases. Did that make sense? Wonderful. All right.

So this is a common strategy here. Notice that we're using the sum rule when what we mean is "or." You can choose a password of length six or a password of length seven or a password of length eight. So "or" means sum.

For product rule, you have to choose a capital letter and then choose one of these characters and then choose one of these characters and then choose one of these characters. I forget how many times I've said it, but five times for this part. We multiply when we mean "and."

Said differently, if I have 10 shirts and 6 pants in my closet, if I want to choose an article of clothing, I can choose a shirt or a pair of pants. And there are 16 options-- 10 plus 6. "Or" means add.

On the other hand, if I'm trying to put together an outfit for the day, I have to choose a shirt and a pair of pants. "And" means multiply. It's 10 times 6 options, so I have 60 outfits. Maybe not all of those look good. But I wear the same pants every day anyway, so it doesn't matter.

Does this make sense? Generally speaking, "or" means add, "and" means times. OK. The last example I want to talk about today is called the generalized product rule.

And example first-- in fact, an example we mentioned earlier. How many different permutations are there for a set of 52 cards? How many different orderings are there for a set of 52 cards? Let's talk through this together.

Here are the cards. I can't hold them all at once with my notes and my chalk. So bye, bye, cards. So how many permutations of the 52 cards? Well, let's think about it. Let's choose the first card first.

Which card is going to go at the beginning of our list? Well, how many options do we have? We have the whole deck to play with. OK, I think I do want to hold the cards.

We have the whole deck to play with. We can choose any one of the cards as our first card. Yeah, maybe I choose this one. I'm doing a magic trick to myself. I don't know what card this is.

So the 9 of spades is going to be the first card in our ordering. So this was the 9 of spades. And we had 52 options for it, yeah? Great. That goes at the beginning of our order. What's the next card we pick? Well how many options do we have left? Just shout it out.

AUDIENCE: 51.

ZACHARY ABEL: 51, yeah. We have 51 options. Maybe we choose this one, without dropping all the others-- 8 of clubs. Great, that's the second card in our order, 8 of clubs. Now let's all watch as I do this 50 more times. No. But we get the idea.

The next card, the third card in my ordering-- well, I've used two cards already. I have to choose one of the remaining 50 cards, then 49, then 48-- then 2, then 1. This is the number of choices I have at each stage.

So it feels like, by the product rule, this answer should be 52 times 51 times 50 times 49, and so on-- 52 factorial. The weird thing, though, is that this isn't the product rule. Product rule says you have some set of options at the beginning, which is true. We have the set of 52 cards at the beginning.

But then we're supposed to have the same set of options for the next entry every time. But the set of options we have in the second round depends on which card we picked in the first round. There are always 51 options because we've always excluded one card from the first round. But the set of options changes depending on what we picked in the first round.

So this isn't straight-up product rule. The set of permutations is not just a set of 52 cards times a set of 51 cards times set of 50 cards, even though the number of options at each stage is always the same. And that's what the generalized product rule does for us. It basically says, if the number of options at every step of the process is always consistent, even if this particular set of those options might change, then the number of ways to make your choices is just the product of the number of choices at each stage.

So this is the generalized product rule. If A is a set of length k sequences where there are n1 choices for the first entry, for a1, and two choices for a2, no matter what a1 was chosen, et cetera, exactly-- sorry, exactly ni choices for ai, for the i-th entry in our sequence, no matter what a1 through ai minus 1 were chosen, and all the way down to nk at the end-- so if earlier choices don't change the number of options we have in later steps-- then the size of A is n1 times n2 up to nk.

Sound good? Cool. We will get more practice with that next time. And I will see you then. Thank you.