I'm Erik Demaine. You can call me Erik. This is the web page for the class. You should all go there if you haven't already, and sign up on this sheet if you want to be on the mailing list. Good.

So, maybe I'll tell you a little bit what this class about. Then I'll tell you about how the class works, and then we'll do more about what it's about. The idea of lecture 1 is to cover the entire class in one lecture. Obviously, I will omit a few of the details because we have a lot to cover, but I thought it would be fun to give you a picture what the whole class is, what the sort of content is, so you know whether you want to be here. And my chalk.

So this class is about geometry. It's about folding. It's about algorithms. In general, we are interested in the mathematics and algorithms behind folding things. And also unfolding things, because that turns out to be pretty interesting.

And the formal term for things is geometric objects. And you think of things like your arm folds. Pieces of paper fold. All sorts of things fold in the world. A lot of the-- if there's any sheet metal objects in this room, maybe some of these parts are folded out of sheet. I would guess so. So folding is everywhere, and this class is all about how that works mathematically.

And I'm a theoretical computer scientist. I do algorithms, so my slant is towards algorithms, which is getting computers to do all this for you. How many people here are computer scientists? How many people are-- you've got to pick one-- mathematicians? How many people are neither of the above? Wow, cool. It's like not a lot of mathematicians only, but a lot of computer scientists, a lot of everyone else. So I'll go around later maybe and find out what your background is.

You may be interested less in the mathematics and more in the applications, so let me tell you a little bit about those. I have a long list of applications, some of which have been realized, some of which are still in progress or just ideas.

Robotics is a big one, folding robotic arms. And I'll show later different kinds of transforming robots, like Transformers or Terminator 2-style, where you want one robot to take on many different forms, this idea of programmable matter, where you can program the geometry of your object as well as you could program the software. So next time when you buy the new iPhone, you download the hardware in addition to downloading software. That would be the crazy idea. That, of course, doesn't exist yet. But it exists in some simple forms.

Computer graphics. If you're making Toy Story 4, and you want to animate your characters from one position to another, you have some skeleton which is a foldable object, and you'd like to interpolate between two key frames that the animator draws automatically. That's a morphing problem, and it's a folding morphing problem.

We have mechanics. A lot of the early folding work is in like 17, 1800s, and is motivated by building mechanical linkages to do useful things. This is back before electrical computers, we had mechanical computers of sorts. Things like you could sign your name in triplicate by just signing once, and having a linkage that made many copies of it. I think Jefferson used such a linkage.

We have manufacturing, which is a pretty broad term-- things like sheet metal bending. Nano manufacturing, I think, is an exciting context, where you can-- we're really good at building flat nano-scale objects like CPUs. And if you could get them to fold up, then you could manufacture 3D nano-scale objects.

I have optics here, Jason. Sure. George [INAUDIBLE] group does some optical devices through folding here at MIT.

Medical is a big one. Imagine folding a stent really small, so it can fit through small blood vessels until it gets to where you want in your body and expands-- do non-intrusive heart surgery. Drug delivery is another one. You have some object you want to deliver, your cancer drug, so it follows through your body until it detects the cancer, and then it releases the drug, for example. Something we just started looking at.

Aero-astro. You want to deploy something out in outer space. You've got to get it there first within the space shuttle. So how do you fold it down small so it fits in your space shuttle, and then expands when it gets there?

Biology. Big one. I'm interested in protein folding. Proteins are the building block of all life forms we know, and we'd like to know how they fold. I'll show you some pictures of them later. We'll talk more about it, obviously, in the rest of the class. And the goal is to use this mathematics of folding to try to figure out what biology might be doing in real life, though that's still obviously very difficult.

Sculpture, sort of an obvious one. Designing cooler origami is possible, thanks to mathematics and algorithms. Hopefully Jason agrees. Jason Ku is, I guess, the top origami designer currently at MIT. That's a safe bet. You could make some broader claim. And we'll get him to give a guest lecture on the more artistic side of origami, and I'll talk about the mathematics-- some of which is currently used in origami design today, either implicitly or explicitly, some of which is yet to be used, but hopefully will make cooler sculpture. You could also imagine building interactive sculpture.

Interactive buildings' architecture is another big one. How many people here are architects? Cool. A bunch. So I think this is really exciting and underexploited at the moment, reconfigurable buildings. Hoberman is one example of somebody exploring this, getting the building to fold from one shape to another, or getting your shades to fold from one shape to another, all sorts of things.

That's all I have. Maybe there are more. You're welcome to tell me. But this is somehow why you might care about the mathematics of folding things.

And what else do I have? Yeah, let me tell you a little bit about the field. It's, in some sense, old. There are some problems in this world of geometric folding that go back four or five centuries. And some of those problems are still unsolved, but a lot of the action in the field has been in the last 12 years or so. And it's been really exciting. A lot of theorems have been proved. In fact, a lot of theorems have been proved in the context of previous iterations of this class. And so I always have to update

things, because we keep getting new results. And the idea of this class is to cover the bleeding edge, whatever the frontiers of what's known, and also to push that edge further.

So there is an open problem session, which is optional. But if you want to come solve new problems that haven't been solved before, every week probably we'll work on a problem related to what I cover in class. So you know what's known, and then we try to prove what's unknown. That's the idea. And that's worked pretty well in the past.

Let's dive a little bit into the sort of mathematical structures. So say geometric objects. There are three main things that we typically think about, linkages, pieces of paper, and polyhedra.

A linkage is something like your arm or a robotic arm. You have one-dimensional straight links like bones, and hinges that connect them together. And you'd like to know how that thing can fold. So this is like in the graphics world or mechanical linkage world.

The typical way we think about linkages is that they have rigid bars. As opposed to a string, which is really floppy, here you have these rigid, like maybe metal, parts, and you can only fold at the hinges. That's typical thinking on linkages. Makes it interesting. And sometimes we also require no crossing.

I'm trying to take a physical thing you have intuition for, and write some of the mathematical constraints. No crossing means you can't intersect yourself. Sometimes that's important. Sometimes it's not. In a lot of the mechanical linkage world you can have crossing bars, but just because they're in different planes in real life, if you're making a two-dimensional linkage.

Paper. This is my drawing of a piece of paper. The rules are you are not allowed to stretch the material, and you're not allowed to tear or cut. So all you can do is fold. And so intuitively this is saying you can't make paper any longer. You can't really make it shorter. The only thing you can do is change its 3D geometry. But if you

look really closely, the geometry is the same. It has a fixed intrinsic geometry. All the distances on the surface are staying the same. And usually in modern origami, you're not allowed to cut the paper, because that makes things too easy, basically. And I guess also no crossing here. It's a requirement. This is a pretty informal description of paper. You can formalize it. We will do that at some point. But it's pretty intuitive. You've all done it before.

Polyhedron is something more three-dimensional. So here we had one-dimensional things we were folding. Two-dimensional things we were folding. A three-dimensional thing, or at least a two-dimensional surface in three dimensions, a typical thing you might want to do with a polyhedron is build it. And often you want to build something out of flat material, something like-- you've probably made a cube by building a cross, and then folding it up into that thing. That's the folding problem.

The unfolding problem is, where do I cut along this surface in order to make one of these nice unfoldings? I think there's one more cut here in the back. If I did it right, that will unfold into the cross. So if I have some complicated 3D shape, what 2D shape do I cut out in order to bend it into that 3D shape? That's the unfolding problem.

There are actually lots of different kinds of polyhedron folding problems, but here it is you want to cut the surface. You want one piece-- it's a little harder to assemble multiple pieces-- and you'd like no overlap in that unfolding. Because if, when you unfold, it overlaps, then you can't actually make it out of one piece of sheet material.

So these are the sorts of things we study. And I think the plan in the class is I will start with paper, because it's kind of the most fun, origami design, computational origami design. And then we'll talk about linkages and polyhedra. And I'll probably jump around from week to week, just to keep it exciting, because they're all interesting. You might each have your favorite one. And so that we can talk about all of them at once.

So what kind of mathematical questions would you ask about these structures? Well, there are two main types I like to characterize. On the one hand, we have

what I call foldability questions, where you're given some existing structure like a linkage, or a piece of paper with some creases on it maybe, and you'd like to know, how does that thing fold?

So I give you some structure, and you'd like to know, does it fold? Maybe just does it fold at all? Or, in some particular way, can you make it fold into something interesting? Or some notion of interesting? I'm going to be generic here, because there's a lot of different questions obviously.

And this is in contrast to a design question, where you're given-- what you start with is a goal. Like I want to make a butterfly. And then the question is, how do I fold a piece of paper to make that butterfly?

So here you're given maybe a crease pattern and given some structure. You want to see how it folds. In design, you're given the goal and you want to figure out what you should build, what folding structure should I make that will achieve this goal?

So what shapes? Shape is one thing you might want. But in general, you have some property you want to achieve. Can you fold it? And of course if you can, how do you do it? So this tends-- in general you could think of this as a more mathematical question, this as a more algorithmic question. But usually actually both of them are addressed with algorithms to some extent.

Of course design is somehow cooler, but often we need to understand foldability before we can solve design. But not always. It depends.

And just to give you like-- I mean if you looked at the entire set of results, the questions and answers, that we consider, and you filter them down, there would be three kinds of things that we prove in this class. These are the results. All the results in three bullets.

It could be either you get universality. This is the coolest kind of result, and it's surprisingly common. Everything can be folded, for some notion of everything. That's always a challenge. And you get an algorithm to do it.

So if you say I want to fold a butterfly, you put the butterfly into the algorithm, out come your design for folding a butterfly. That's the ideal picture when you get a universality result.

The next best thing you could hope for is a sort of decision result, which is that there's a fast algorithm that will tell you whether something is foldable. So we say it decides foldability. So you give it a butterfly. It says, oh, that's impossible. That's not true. But you could imagine that. Or maybe you give it a butterfly, it says yes. You give it the skyline of New York, and it says no, you can't fold that. That's not true. But if it's not the case that everything is foldable, the next best thing is you could at least distinguish which things are foldable from which things are not.

And then, of course, the worst thing that could happen, in some sense, is you get a hardness results which says, there's no good way to even distinguish foldable things from unfoldable things. So I'm going to say computationally intractable to mean generically that there's no good algorithm to solve a problem. We're going to be more formal about that in the future. Because different people have different backgrounds, you may know about competition complexity. Many of you don't. That's cool. We're going to talk about all that here.

All right. That's the class in a nutshell. Totally generic. Any questions so far? Probably not. I thought now I would tell you a little bit about the class structure. The main part of the class are these lectures, and attendance is mandatory for classes, because you're not going to learn it unless you come here. You can have exceptions. Just email me and watch the videos later. That's one of the fun features of the videos, although the videos are mainly to reach the world for fun.

There will be a few problem sets in this class, not too many. And the other main thing that you turn in is the project. So sort of a project-focused class. You will hand in some write up. You will give some presentation in the last few lectures of class, if you're taking the class for credit. If you're listening, you don't have to do this. You can, if you want. You can do the problem sets. Some of them will be fun. Hopefully all of them will be fun. None of them will bee too painful.

But for people who are taking the class for credit, you have to do a project and presentation. The project could be about tons of things. You could build a sculpture. You could come up with a cool virtual design of something amazing that somehow relates to folding. It doesn't have to be direct. If you're a coder, you could implement some algorithm we talk about, or make a beautiful image or animation or applet or something.

If you're more theory, you could solve an open problem. Obviously that's a big win. But even trying to solve an open problem is fine. You can talk about how you failed to solve it, in the unhappy case that happens. But if you want to solve an open problem, I would encourage you to come to the open problem session, so we can all solve it together. Then you can have a joint project on what we do, but it's totally unpredictable, of course.

Even posing an open problem in this field is pretty interesting. I know a lot of the hard open problems. I would like to find more of them, more tractable ones. And so if you have some idea, especially related to some application field that you know a lot about, it'd be cool to try to extract, what is the mathematical problem in a lot of these fields?

Or you can write a survey. That's a typical project. You should avoid overlap with-- I don't actually have it here, but the textbook for this class-- you can imagine it being here-- is *Geometric Folding Algorithms.* Here it is. Thank you, Jason. That's good. I ran out of copies. Now I have one. No, I'll give it back. So this is by me and Joe O'Rourke at Smith College. Any questions about class structure? Yeah.

AUDIENCE: Can projects be individual? [INAUDIBLE]

PROFESSOR: Projects can be with groups. I forget whether the website has a limit on the number of people. I don't think so. But at some point you'll have to do a project proposal. I should mention that also. And then I will vet your project group. But I think anything's fine. Just when you have more people, you're expected to do a little more, naturally.

Other questions? Yeah. I love collaboration. I think I've never written a paper without a co-author. It wasn't a survey paper. So collaboration's good. All right. Doing well.

So the next thing I want to do is actually dive into actual content. This was totally generic. And to me, it's useful. But maybe to you, it's less useful. This is sort of an organization. As we go through, everything you'll see will fit almost always into one of these three categories. The question will fit into one of these two categories, and the answer will fit into one of these three categories. Now let's see what some of those actual problems, questions and answers, are. That's the fun part. There's so many cool results here, and so many cool open problems. I thought I'd tell you a bunch of the big open problems, too.

So in today's class, I'm going to go in order. We're going to start with linkages, and then paper, and then polyhedra. And I'm going to start with linkages allowing intersection. So magically, bars of the linkage can overlap each other. And then later I'll talk about linkages, which is most interesting in two dimensions, because you can actually build them in three dimensions. And then we'll go to linkages that do not have intersection.

So an early motivation here where a lot of this linkage folding came from originally-- this is the 17, 1800s-- is converting linear motion into circular motion. Actually, it's really-- yeah, get it right. Linear motion to circular motion.

I don't know what order to cover these things. I'll show you a slide. My rules are essentially no words on any slide. This is just for pretty pictures, and I'll write stuff on the board. There's this fun book called *How to Draw a Straight Line* by Kempe in 1877, which is all about this problem. How do I turn a circular crank and make a straight line come out as a result?

And the motivation for this problem is steam engines. You have a steam piston, which is moving something up and down along a straight line. And maybe you're building a locomotive train, and you want to turn a wheel in a circle, because wheels are round. So how do you convert this linear motion into a circular motion? That is

How to Draw a Straight Line, affectionately titled.

But you can see that one of the earliest linkages for this is called the Watt parallel motion. And we have a little animation here. So the idea is this vertex in the top left is pinned down. This one in the bottom right is pinned down. And then, I think if I move this around the circle, the green guy moves along that figure eight. And there's a limit to how far it can go.

So and if you draw it right, the figure eight is almost a straight line. So that was mathematics back in the day. No, no one thought that that was perfect, but it was pretty good. And that actually led Watt-- you may have heard of Watt. He's a unit. And he was very proud of this invention. And he made tons of innovations in the steam engine world, and this was his favorite. And, I mean, it changed things.

But later on-- I mean this is like 100 years later, 1864 versus 1764. Exactly 100 years later. Peaucellier a French guy in the army, I think, came up with this linkage. So again, you have two pinned vertices here. This one's moving around a circle. So you just turn the crank, and look. This guy moves along that red line. Perfect. Very cool. Again it has a limit how far you can go. But it's pretty awesome. I mean you could play with this forever. I won't bore you.

There's this other guy you may have heard of, Kelvin, Lord Kelvin. Another unit. There's a story about him playing with one. He wouldn't want to give it up, because he was having too much fun just pushing it back and forth. It's like, wow, a straight line out of a circle.

So that's pretty cool. Making straight lines out of circles is pretty neat. In fact, you could think, well, what else could I make? I can make a straight line. That's kind of nice. But could I make, I don't know, some other curve? I mean a straight line is a special kind of curve.

Maybe I could make this curve. That's a nice curve. In general, there's a universality result which says there's a linkage to sign your name. That's the cute phrasing. So it's a two-dimensional linkage. You turn one circular crank, and it signs your name.

The mathematical version is that you trace a piecewise polynomial curve. Or if you're an architect or graphics person, call this a spline. Pick your favorite word. You can make it all with one linkage. It's pretty crazy, and not super practical. Building this would probably require thousands of bars. But hey, who's counting? At least you can get a universality result.

This is actually something that goes back, in particular, to the first time this class was taught six years ago. We found some better ways to do, so it's known that it could be done. And I'll talk about those ways later on. Cool.

Open problem. Not everything is known. There's a lot of open problems here, but one of them is, what if you forbid crossings? So these linkages, like the ones I've been showing here, like the Peaucellier linkage, there's crossing bars. And yeah, you could do that, but what if you forbid crossings? It'd be really cool to sign your name with a linkage that doesn't even cross itself. That's totally open. Getting any kind of result, positive result, maybe not everything, but at least getting some interesting things. Even drawing a straight line, I think, is open. Could be a fun problem to work on. All right. That's one.

Let's go over here. Let's see, is that the end of linkages, is crossing? No. Next question you might ask is-- so that's sort of a design question. I was given the goal, which was to make a straight line or sign my name. And I wanted to find the linkage that did it.

The other question, the foldability question, one of them is rigidity, which is, does a linkage fold at all? I should say a given linkage. So for example, I'll give you three examples. Test you out here. These are rigid bars connected by hinges. OK, rigid or flexible? Rigid, I agree. Rigid or flexible. Rigid. Any other answers? Flexible, correct. Both are correct.

In two dimensions, OK, it depends where you live? In two dimensions, this is rigid no matter what, in any dimension. But in three dimensions, this is flexible and in two dimensions, it's rigid. In three dimensions, you can pick this guy up and it spins around a circle, out of the board here. Or you could pick this guy up. It spins around

a circle. In two dimensions, though, it's rigid because it's really just two triangles. Triangles are rigid in two dimensions. This is a tetrahedron. Tetrahedra are rigid in three dimensions. Rigid or flexible? Everyone agrees. In both dimensions it is flexible.

OK, pretty intuitive for four vertices. But you can ask the mathematical question and give you a linkage. Is it rigid in 2D? Is it rigid in 3D? And there are many versions of this question. But the short version, a short answer, let's say, is that distinguishing rigid from flexible two-dimensional linkages is easy. There's a good algorithm to do it. It's very powerful, very useful. In 3D, we have no idea. Very open. Tough. Very tough problem. A lot of people have been thinking about that for decades. So that's rigidity. I'm just going to touch on lots of topics very briefly.

All right. Next we go on to, I guess, one and a half. This is linkages forbidding intersection. And this is more interesting when you're talking about 3D linkages like my arm. I really don't want it to penetrate other bars. It's not possible. And the first question you might ask in this world, which I guess is a foldability question, sort of a reconfiguration question, let's say I want to fold my robotic arm from one configuration that I know-- call it configuration A-- to some other configuration, B.

When is that possible? Can I go from A to B? Can I go from some other A prime to some other B prime? Sometimes yes, sometimes no. It depends. Sometimes it's not foldable at all, even when you allow intersections. So this is a pretty open-ended question. In general, it's computationally intractable. If I give you a linkage and two configurations, to decide whether you can go from A to B is, for the complexity theorists, piece space complete. Talk about what that means later. Really, really hard is the short version.

But a lot of the times you can think about special linkages. There are a lot of interesting special cases. In particular, we like to think about chains like I drew before. Also polygons. That's a little messy, but imagine those don't self-intersect. So this is what I call an open chain, and this is a closed chain. In general, these are chains.

And the other thing I might like to think about, in particular because proteins look kind of like this, are trees. Trees are just linkages without any cycles in them. So those are nice and simple. Here I have no cycles, no cycles, one cycle. Easier to think about. And sometimes you actually get a universality result that these linkages can fold from any configuration to any configuration. And that's especially cool. Let me tell you about them.

Where do I want to go? So it depends again what dimension you live in. I'm very flexible in this class. You can live in any dimension you want to, and even fictional ones. And you can think about chains and trees, let's say. You could go more general, but this is where most things have been studied.

And the answer is, for chains in 2D you get a universality result. You can fold from anything into anything. For trees in 2D, you don't. There are some trees you can't get from one configuration to another. Which ones? We don't know. But at least you don't get a universality result. In 3D chains you do not get a universality result. And so also for trees, because that's even harder. And for 4D, everything's easy. Also in 5D, any higher dimension.

Because the intuition here, at least for this column, is you think about tying knots. I have a one-dimensional linkage here. Think of it as a one-dimensional cord. It's kind of a kinky cord. It has kinks-- not the other kind of kinky.

And in two dimensions, if you draw a non-self-intersecting loop, it's never knotted. You can't draw a knot in two dimensions. You can drawn a knot in three dimensions, and you cannot draw a knot in four dimensions or higher. You may not know that result, but it's true. So it matches, but things are little tricky, even trees. If this was a piece of string, you'd be able to always fold this piece of a tree-shaped piece of string into anything you wanted. But it's a little more complicated.

Let me show you a locked tree, I think, is next. Yeah. For a long time, these were the only known locked trees. These are configurations of tree linkages that cannot reach some other configuration. In fact, they can barely move at all. It's less obvious for some of them. But say, in this top left one, you have these little sort of petals

tucked into their armpits, I guess, and you can't get any of those arms open unless you had a lot of room to open it. And in order to make room, you'd have to squeeze all the others really tight. And if you draw this example tight enough, also none of the arms can get compressed very much. And so it's locked. And this is one of the first examples actually discovered in 1998, and publication took a while.

There are a few others, which you see here. This one's kind of crazy because it has only one vertex with three incident bars. Everybody else is like a chain. So it's like three chains joined together at that point, and still it is locked.

And for a long time, these were the examples we would always carry around. These are the ones that appear in the textbook. But I thought it would be neat to see, well, are there any simpler examples? And last time this class was offered three years ago, we found what is believed to be the smallest locked tree in existence. It has 1, 2, 3, 4, 5, 6, 7, 8 bars, if I counted right. And it looks curved here, just to make it easier to see. But, in fact, you could straighten these out and it's still locked. If you squeeze these little regions down, they'd be very tight. So that was with a bunch of students from this class. Cool. So that gives you some idea of this answer.

Maybe I'll draw you a picture for chains, because it's really simple. OK, imagine tying a knot, but don't actually close the loops. And make these end lengths really, really long. We call this the knitting needles example, because the intuition was you have two long knitting needles, and then a very short cord connecting them in a knot.

Mathematically, this is not a knot, because if it were string, you could untie it, no problem. But because of these really long bars, you can't untie it. So that's why 3D is hard. Or one example of why 3D is hard. It's pretty much our only example. It is the smallest example, and we can prove things about it.

But there's a pretty fascinating open question here, I would say, which is, characterize these bad examples. Which 3D chains and which 2D trees have locked configurations? And all that means is that there are two configurations, A and B, for which you cannot get from A to B. So this is an example. Those locked trees are an

example. It'd be really fascinating if you could do this. I would guess that this is a hard problem, but I don't know. It's hard to know whether it's hard.

It'd be nice to understand 3D chains in particular, because they relate-- and 3D trees. Oh, sorry. I have more animations. I forgot. Let me show you some pretty pictures for this result. This was actually my PhD thesis, way back in 2001. And this is a more modern algorithm for solving this problem.

I give you some complicated polygon. First thing you want to know is, can you unfold it into a nice convex shape? Once you get there, you could refolded into some other shape by playing one of these motions backwards. So that's how you unfold some teeth.

Here's one of those tree examples, but doubled. For a while, people thought that might still be locked, but it's not. Can do this crazy fivefold rotationally symmetric motion to unfold that thing. We're zooming in and out, so it looks like things are getting bigger and smaller. But in fact, each of these bars is staying the same length, and they're never crossing each other.

Here's a much more complicated example. This is the first algorithm that could handle examples of this size. It's pretty fast. That's like it's going to come back. You can do it. It's like a spider. Spooky. I think it has 500 vertices, and it probably took a couple minutes to compute. I'm not computing it live here. There's an applet on the web if you want to pick your favorite polygon and run this algorithm on it. It doesn't make quite movies like in this style, but it will show you how it unfolds. We call this the tentacle.

Yeah, so we'll talk about this algorithm, how it works. There's a couple other algorithms for solving this problem. It's pretty cool, and you could imagine using this for planning the motion of a robotic arm in two dimensions. But in three dimensions, things are a lot harder. We lack good algorithms. I would like to study four dimensions actually. There's some pretty neat questions here, but I'm not supposed to talk about that. It's not on my list. I've got to move quickly, but we're doing all right on time.

What I wanted to show you was a protein. This is a particular enzyme protein called hexokinase, whatever. Embarrassing myself. It's a particularly complicated one. It is one of a few I could find a nice free image of. But you can see closely, if you look closely here, it's a linkage. And this one actually has lots of cycles. But the backbone of a protein is like a tree, and so it fits into this kind of world. Unfortunately it fits in this world of three-dimensional trees, which are really hard to fold. Or three-dimensional chains, if you really just look at the backbone. They're a little bit more complicated than the sorts of linkages we're talking about here, but it makes it even harder to fold things like this.

But there's something special about proteins that makes them fold really well. That enzyme over there is in every living organism we have ever tested for the existence of that thing, which I assume is everything. And yet it's folding. It's produced by this machine-- the Ribosome, you probably know about it-- in a sort of straight state, and then it folds into this shape pretty reliably, less than a second, usually like within nanoseconds. So it's really hard to watch what's happening. It's very hot and jiggly, so it's a little hard to see what's actually going on in the real thing.

But somehow, these kinds of barriers to foldability don't happen. Maybe that's because evolution found the right things, or maybe it's because protein chains don't really look like this. They don't have these super long bars and super short bars. If you've ever played with a chemistry set, all of the bars are within a factor of like 1.5 of each other. So there's lots of cool mathematical questions that come out of protein folding, and we will talk about the ones I know. I'd love to find more. But the ones I know I will talk about. And that's linkages.

Let me move on to paper, unless there are questions. All right. Let's go over here. So we've seen some universality results, some hardness results. I didn't go into them. Folding these linkages. Let's do that for paper. I think first up I have foldability. And then we'll talk about design.

So there are lots of questions in both. But for foldability, the sort of first question people like to ask is, which crease patterns fold flat? So a crease pattern is just a

graph drawn on a piece of paper. So you have some collection of lines, and I think that will fold flat. But I'm not going to try to draw something more complicated, but you could imagine doing it. I don't know. That probably is not flat foldable. It might be. It's close.

But that's the sort of question. If you take some origami and unfold it, what kind of patterns do you get? Some flat origami, something that folds into two dimensions in the end. There's some really nice structure here. If I drew it right, this angle plus this angle should equal 180 degrees, for example. And that's true everywhere. And there's all sorts of cool properties. But unfortunately, this is really hard.

This is NP hard problem. You've probably at least heard of P versus NP. Again, I'll define it later. But it means probably there is no good algorithm to solve this problem. It's really hard to figure out which crease patterns fold flat. It's kind of annoying.

One good news is that if you just have an example like the original thing I drew, which is like this-- so it has one vertex and a bunch of creases emanating out, that picture we understand. So it's easy for a single vertex. That may seem kind of trivial, but it's actually really useful because it lets you understand the local behavior around one vertex. If you check that for every vertex, you don't know that the whole thing folds, but at least you know it mostly folds, at least locally. And you can't tell whether it globally folds correctly, because that's NP hard.

There are so many questions here like, what about two vertices? No one's studied that. I think it's polynomial, but well, it's certainly polynomial, but I think you could do it in linear time. Anyway, there's lots of open questions there I haven't even listed here.

One of the bigger open questions is a particular kind of crease pattern, which you may have encountered in real life refolding your roadmaps. The saying goes, the easiest way to refold your road map is differently. So suppose you have your road map, and each of these creases is marked. I'm not going to mark all them because

it's a little messy with black and white chalk. But some of them are marked mountain, some of them are marked valley, meaning you fold-- this is valley. This is mountain. So they just have a relative orientation to each other.

Suppose you look carefully at your map. You can recover which way it was folded last, hopefully correctly. And then you want to find, does that thing fold flat? Sometimes it does, sometimes it doesn't. For two by n maps, we don't know whether we can even detect this with an efficient algorithm. Can I decide whether this thing folds flat using those creases and those orientations? For two by n, it's open. For one by n, it's easy. Obviously for bigger, it's also open. Two by n is the smallest. This is a really annoying problem. Worked on it many times. It's very difficult.

All right, so that's a quick overview of foldability. It's hard, but there are a lot of interesting special cases where we might be able to solve it.

Let's move on to design, which is probably where most of the action is in the origami world, in the mathematical origami world. So and there are tons of results, and I had a much longer list initially, but I trimmed just for brevity down to a few things. The central question-- at least it's been considered so far-- in computational origami design is, what shapes can you make? You could imagine other properties than just shape, if you want stability in your folding, lots of practical things-- not much thickness, not too many creases, whatever. But shape has sort of been the fun centerpiece.

And there's an early result that says universally, you can make anything you want. And in the mathematical world, you might try to make a polygon. Try to make a silhouette or something. You could make a polyhedron in 3D. Maybe you want to wrap a box, so you want to fold a box, or you want to fold a 3D model of a dragon. That's all possible.

And for fun, you can even make any two-color pattern on either of those things. So I have a tiny example here, which is-- yeah, it's still alive-- a four by four checkerboard. And it's made from one square paper. It's white on one side and red

on the other. So you get some idea of making two-color patterns. This one's pretty easy to fold. It even folds itself almost. That's great.

Usually when I give a class like to high school students or something, I say, could you refold this for me? And they're like, oh, no! But it actually does it itself. It's great. Like magic. So you could try that at home. Just take a square paper. It works.

So this is great. Super general. This is something actually my dad and I proved with Joe Mitchell back in '99, I think. I don't remember exactly. So beginning of modern computational origami design. But the way that we do this, while algorithmic, is completely impractical. We'd never want to use the foldings that are designed by this algorithm, unless you're starting out with a big ticker tape of paper. Then it's a great method. But if you're starting with a square, the first thing it does is fold it down to a tiny little narrow strip, and then wraps the shape.

And we'll see how that's done. It's interesting to do it mathematically, but it's not how you want to do it practically. Good news is-- that used to be sort of the end of the story. But now you can even do it pretty practically-- practically is a relative term-- using something called Origamizer. And if you've seen the poster of this class, you've seen-- and if I'm lucky, it's even the next slide, yes it is-- this rabbit-- bunny, I should say.

This is the Stanford bunny. The original Stanford bunny is in the top right. That's a classic model. Everybody in computer graphics does something with that bunny. It has a zillion triangles. I don't know offhand how many. It's been simplified here to make it feasible to fold to this mesh of triangles. So the input to the algorithm was this mesh of triangles. The output of the algorithm-- and it's a real program, you can go and download it right now for free-- is this crease pattern.

Now it doesn't look like a square or paper, but if you start with a square of paper, you just fold away the excess stuff. You'll get to a piece of paper like this. And then you just fold along all those little creases there, and eight hours later you have this bunny. This is a real photograph of a folded bunny by Tomohiro Tachi, who designed this thing. He came up with the original algorithm and computer program.

In the last few years, we've been proving this algorithm actually always works. We still don't know how to prove that it's efficient, or even what efficient should mean. But in practice it's super good. If you look closely, the white parts are the parts of the paper that you need to use in order to make this. They are the triangles of the surface. These grey regions are the excess. And it's pretty small. I don't know, maybe 50% of the area is used in a useful way here, which is a lot. Whereas this method over here would use like a one millionth of a percent of the paper or something. A very tiny amount.

The Origamizer is super efficient. We don't know how to prove that it's super efficient, but at least we can prove that it works. It will make any 3D polyhedron you want. And in practice, it seems really good at it.

Now it's a very different style from typical origami, so this has not yet hit the sculpture world, let's say. But I think it opens the door to a lot of new possibilities. Traditionally though-- traditionally before this Origamizer or in practice today, a much more commonly used approach to computational origami design is something called the tree method. I'm just going to call it TreeMaker here. TreeMaker's the name of the program that implements the tree method, and it's made by Robert Lang.

And this is sort of making stick figures, which sounds a little silly. But TreeMaker or the tree method is powerful for if I want to make what's called an origami base, that has a lot of limbs in different places of different lengths-- maybe you want to make something like that. I don't know. The typical origami base might look something like this. Maybe you want to make a lizard. It has a head, some forearms, some really big hind legs, a little tail, and some body.

You can abstract this into a tree and then say, well can I fold a piece of paper into some shape whose projection is that tree, with all the right edge links? And the answer is yes. I mean, you can make anything, of course.

But what TreeMaker tries to do is find the most efficient way to make that thing.

Unfortunately, finding the most efficient way-- like the smallest square paper, so the most paper usage possible to make something with that projection-- finding that is NP complete. We just proved that this year with Robert Lang and Sandor Fekete.

But there's a theory. There's a hard problem you have to solve, something related to disk packing. If you solve it, you find the best way. In practice, use some heuristics which are really good. And usually you find the optimal way to make a given stick figure. But for super complicated things, it may not be perfect.

Now a lot of people use this method for designing origami. I have a slide of cool origami, just to show you. Not all of this uses the tree method. These three designs by Brian Chan, who was an MIT grad student, graduated last year-- these are all designed partly using the tree method. I'm sure they're more complicated than that. But to get the initial structure of the arms in the right place, the legs in the right place, the head, body segments, fingers-- some of these guys have fingers-- all that stuff is done using tree theory, almost certainly not with TreeMaker, the program.

Most origami designers do it by hand, in their head, or drawing pictures with a drawing program, to design out the base. Then they fold by hand everything. But still, the math is in there.

These guys are not yet mathematically analyzed, let's say. They're experiments by [? Garn ?] and Joel Cooper. [? Garn ?] is a computer scientist slash mathematician, but he doesn't understand these yet, and sort of he's trying to figure it out. So there's a lot of interesting questions, and you get some really cool sculpture as a result, out of this world.

So this is something we've also been working on with my dad and Robert Lang, proving that this actually works. It's been around for a long time, but it turns out to be really complicated to prove that it always works. But it looks like it does, so stay tuned. Some time we will publish that paper. It's still in process.

All right. I have some more fun things. This is just a random selection. There's a ton of work here. One of my favorites-- this was my first result in this field-- is the folding

cut problem, theorem, whatever.

So you start with a rectangle of paper, and you fold it flat. Now you take your scissors-- I know this is blasphemy for most origamists. But I'm not going to make a lot of cuts. I'm just going to make one complete straight cut. I get two pieces in this case, and then I unfold the pieces and see what I get. In this case I get a little swan. And the general theorem is you can make any polygon you want. In fact, you can make one cut and make any collection of polygons you want.

So like on my website, you can make the MIT logo with all the little rectangles with one complete straight cut. And the outside shape will have holes exactly where you want them. And you can download this, too, if you want to impress all your friends, especially recommended for, like, kids. It's a good magic trick. In fact, Harry Houdini used to do this trick in the '20s before he was an escape artist. He did general magic. Not this trick. He made stars. And we thought about it, and we proved you could make anything. So that's fun. Another universality result. It's what got us started.

And it relates to things like the tree theory, in fact, because what you're really solving here is, how do I take a polygon and-- if I fold this guy back up-- how do I make all the edges of that polygon lie along-- this guy's not very happy, get rid of that-- lie along a straight line. That actually turns out to be important for origami design. A lot of these things-- an Origamizer-- it's all over the place. And so this structure is helpful for solving sort of pure origami problems, not with cutting.

So what else do I have? Curved creases. You may not know this, because most origami is straight creases. But if you take a piece of paper, you can put a curved crease into it. It's a little hard to do by hand, but there you go. Curved crease.

Is that allowed? What can you make like that? It's not too well studied. Jeannine Mosely has done some of the curved crease stuff. Early stuff was done by David Huffman, you may know from Huffman codes in every cell phone, whatever. And by Ron Resch. That's wrong queuing. All right, I'm going to show that later.

This are some of the things that we've made, this is my dad and I, out of curved creases. This is kind of a crazy-- it's like a circular piece of paper, but it goes around twice like a ramp. So you make two circles and then join the ends, and you pleat along concentric circles, alternating mountain and valley. And you change a few different parameters, and you get these three different shapes. These are in the permanent collection at MOMA in New York.

So you can make some really cool sculpture out of this. This is a more recent piece we did. And this is taking just a regular circular band of paper with concentric circular creases, and then taking three of them and joining them together at a few points. And then you get some cool structures.

These are interesting to us especially because the paper actually does fold itself. You put in these circular creases. You just squeeze a little, and it will make these shapes automatically. It's called self-folding origami.

And in that spirit, I will show you a different kind of self-folding origami. I think I need to click this button. Which is approach to making paper that folds itself. This is a self-folding sheet. It's made of rigid panels connected by little rubber hinges, and there's little muscles, you could say, that are folding those creases shut. By turning on some electrical signal, that thing folds itself into a boat.

That same sheet, you can send it a different electrical signal and it will just fold along these two creases. And then there's little magnets to hold it into place, so you can turn off that electricity now. No power. Then you send it a third signal, and it will fold those three creases, and you get a paper airplane. No origamist required.

So the idea here, this is the programmable matter vision, where you could download hardware in the same way you could download software. You can make one sheet that has lots of panels in it, enough creases to make anything you want. And then you just push a button and some of the creases turn on. Then push a button and some other creases turn on, and it folds into some complicated origami all by itself. You could imagine making this building scale. You could imagine making it nano scale, some scale where an origamist can't do it, or you don't have an origamist to

do it for you. This would be the self-refolding, transforming robot thing.

To do it, though, we needed to prove new theory. And this is the world of, call it, universal hinge patterns. And this is the topic of a master's thesis just finished last week, I think, by [? Viva Vadia ?] where-- and maybe we'll get him to talk about it at some point-- it's one crease pattern. In that case we were using a crease pattern called box pleating, where you take a square grid, and then you put alternating diagonals in those squares. Get my alternation going here. So that's the so-called box pleating pattern.

We proved that that pattern, if you make it big enough, can fold any 3D shape that's made out of little cubes. I think, yeah, so in fact if you take a sort of n by n sheet, you could make about n cubes. And in the worst case, that's the best you can do.

So that's cool, because that tells you here's one robot you build. You just make this sheet with those creases, those rubber creases with muscles on each of those edges. And if you can turn those edges on and off programmably, you could make anything you want up to some resolution.

If you want to make a bunny, you can sort of build that bunny our of little cubes in simulation, and then fold that cubefied, voxillized form of the bunny. And then you want to make something else, you unfold it and refold it all automatically.

AUDIENCE: This is like [INAUDIBLE] no width of the material?

PROFESSOR: Yeah, there's a lot of open questions here. This result assumes 0 thickness. In fact, every result I've talked about assumes 0 thickness. I think there's one theorem in the literature that looks at thickness of paper. That's a great open question, and you really see it in something like that video. That just appeared at PNAS, and the video's online. Because in the airplane, you have to fold through multiple layers. And that's tricky. We get around that right now by having the creases be a little bit stretchy, some rubber. But there's a limit, and it's related to the thickness. So yes, some complicated shapes are not going to work this way.

I think one thing that would work really well-- we have another result with Jason Ku,

which is folding a maze. This is sort of like folding a shallow terrain in this way, so you could make a rat maze out of a square-- just tracks by 50%, and then you have arbitrary undulation in the middle, which is pretty cool.

So some of these things are going to be practical. Some are not. So there's still interesting theory questions to ask about. I think thickness is a really good one. Other questions? Yeah.

AUDIENCE: [INAUDIBLE] I thought I heard something about how you can't fold a sheet of paper more than like--

PROFESSOR: Yeah, so that's the one thing. The question is, you can't fold a piece of paper in half 8 times, 7 times, whatever the number is everyone quotes. The answer was solved by someone who was a high school student a few years ago. What's her name? Britney Gallivan.

And so this is the one paper that's about thickness, and she analyzed-- the issue is when you fold-- if you have a folding that is 1,000 layers thick, and you want to fold in the middle, you can't just like cut here and move it over here and then be folded. Paper has to actually turn that corner. And the thicker that thing is, the more length of paper it takes to turn that corner. And you can compute how much that is if you assume sort of a circular trajectory, and it matches reality.

So she computed how thin does a piece of paper need to be, or how long does a piece of paper have to be-- it's that ratio-- in order to fold it in half 12 times. So she took a piece of paper that was, I think, three-quarters of a mile long, and folded it in half 12 times. And it's like this big monstrosity when it's done. And it follows the theory exactly. So that seems to be the right model, but that's the only paper that analyzes paper folding in that model, because it's so much easier to think about zero thickness. But that is what we should do. So that particular question we understand, but everything else is open. I'll show you some pictures of that next time. Other questions? All right.

That is paper, and then I was going to show you a little bit about polyhedra and

hinged dissections. All right. Let's go. So I mentioned there's a bunch of different kinds of polyhedron folding and unfolding problems. Probably the coolest one-- because it's the oldest problem in this entire field-- is unfolding convex polyhedron by cutting along the edges. We call this edge unfolding convex polyhedra.

At the beginning of class, I showed you a cube which you could unfold into a cross. I want that. So I was only cutting along edges of the cube. That's this edge unfolding. The cube is a convex shape. It doesn't have any dents. And I got to unfold it with one piece without overlap.

Is that always possible? We have no idea. Can every convex polyhedron be made in that way? We don't know. This problem goes back to Albrecht Durer, this guy on the left. This is his self-portrait. In 1525, he wrote this book called *The Painter's Manual* in German. And he tried this out for a whole bunch of polyhedra. This is the so-called snub cube, and unfolded in his book. And he has page after page after page of these unfoldings. And they're all edge unfoldings. Most of them are correct. There's a couple small errors, probably just transcription errors.

And he did this because he wanted to understand these 3D shapes. He had to build them. How do you build them? So he didn't pose this mathematical question. He was not a mathematician. But it's sort of implicit there, and people have been thinking about it for at least decades if not more. At least since '65, I think, was the first formal posing. And it's really hard. A lot of people have worked on it. We'll talk about it.

There are some interesting things that are known. I mean the depressing thing is we have no algorithm which we think will work. Every algorithm we've tried has a counter example. Every counter example we've tried has an algorithm for which it works, but they don't match up, so it's frustrating.

If you allow non-convex polyhedra, then the answer is no. So that's something I did with a bunch of people back in the day. This was actually done in '98, then it took forever for the journal paper to appear. This is a polyhedron. All the faces are triangles. So it's sort of topologically convex. And then if I made the spike a little

shorter, it would be a convex polyhedron. But this particular embedding-- these are just two views. If you cross your eyes, it's three dimensions. No, I'm just kidding. It's not designed for that. Just two views of the same thing.

And if you cut along the edges any way you want, it will overlap itself or be multiple pieces. And we'll prove that at some point. It's not too hard. Yeah, so good. That's edge unfolding.

But there's another kind of unfolding called general unfolding. This seems a lot more interesting to me. Edges are sort of artificial. What if you let me cut anywhere on the surface, not just at the edges? Well, then it turns out, for convex polyhedra, you can do it. It's always possible. I don't have pictures of that here, but even for some non-convex polyhedra you could do it.

For example, this polyhedron, if you let me cut not just at the edges but anywhere on the surface, you can do this crazy thing. These are the spikes, and you cut out a tiny little sliver out to the side, and that lets you attach one of those spikes from one of these witch's hats to the wrong side. Like you see, this blue guy, the blue spike-- because this edge ends up being glued over here, we can attach the spike with the little sliver that gets opened up here, over there, and avoid overlap. Even if these spikes are super tall, does this go out to infinity and overlap. That's one example.

Can you do it for every non-convex polyhedron? We don't know. I would love to answer that question. The one thing we know or the big thing that we know is so-called orthogonal polyhedra. This is always possible. This the same kind of thing I was talking about. You take any 3D shape made of little cubes-- so all the faces are horizontal, vertical, or the other way-- then it's also always possible.

The current method takes an exponential number of cuts. It's really impractical. I think we could make it practical. I think we could generalize it to everything. But those are all open questions.

One more picture I wanted to show you is, so here we really want to keep the shape connected. You have to have a positive area of connection between everything. If

you let that connection go down to a single point-- sort of cheating, this is called vertex unfolding-- and then it's possible for any surface made out of triangles. So it could be non-convex, anything. These happen to be convex polyhedra. It's a little hard to see that they're three-dimensional, but they are.

You cut them up into little faces. Here I'm only cutting along edges. And I string them out on a line here. This thing will not overlap, but I'm kind of cheating because the connections are only at single points. But hey, we'll take a positive result if we can get it. A lot of these problems are really hard.

OK, the next thing I want to talk about is the reverse direction folding. So we know-- this is a video I made back when I was a grad student, a long time ago. So we know you can take a cube, and there's a whole bunch of different ways to just cut along its edges and unfold it into one piece. And the particular unfolding I want to consider is the cross unfolding, because that's our favorite.

So you take this cross unfolding, and ideally don't stutter. And you say, well what if I gave you that polygon? What 3D shapes could it fold into? One thing is the cube, but it can also make this flat doubly covered quadrilateral. So I'm taking this as a piece of paper, but I can set the creases however I want. I can fold it into this five-sided, reflectionally symmetric polyhedron. All of these things are convex, and they all have perfect coverage, unlike an origami where I'm allowed to have multiple layers. Here, one layer everywhere. Like sheet metal bending.

So here, this pocket fits perfectly into that tab. I make a tetrahedron. And the last thing you can make with something called edge-to-edge [INAUDIBLE]-- I'll talk about this more. There's an algorithm here which, given a polygon, will list all of the things it can fold into, in a reasonable amount of time.

This is an octahedron. Those are the five things you can make by edge-to-edge [INAUDIBLE] of the cross. Let me stop that. So that's also on the web. You can watch it. In fact, all the lecture notes I use, my handwritten notes, are on the class web page. The slides that I'm showing are on the class web page. And when there's videos, there's links to the videos, so it's all there for your perusal if you miss

something.

The last thing I wanted to talk about, because I've already done one, two and three, there should always be one more thing. Fourth thing. And this is kind of funny, because I don't really know where to put it. I'm still figuring it out. We have these debates, because it's so easy when you're thinking about linkages. Oh, that's one dimensional. Paper, that's two-dimensional. Polyhedra, well locally it's two-dimensional, but it's three-dimensional. It's different. Different questions.

Hinged dissections span so many dimensions, I don't know where to put it. So four, hinged dissections. This didn't used to be a topic in this class, and it's not a topic in our textbook, because the result was proved like last year or the year before out of this class. Three years ago.

And the theorem is you take any finite set of polygons, of the same area, they can be folded from one chain of polygons without collision. I'm going to say this first, and but let me show you a simple example. Very simple. This is going to be a little boring. Let me show you more interesting example.

Let's take an equilateral triangle. I'm going to cut it into two triangles, and then I'm going to hinge them together here. So this is now a hinged chain of polygons. If I open it up a little bit, it looks like this. It looks a lot like those vertex unfoldings I was showing before.

And you can fold this thing into one other shape. It's going to look like this. Not a very good drawing, but you get the idea. Something like that. So this is what we call a hinged dissection from this equilateral triangle into this other triangle. And it's been an open question for about 100 years. Can you always do this for polygons of the same area? Turns out yes, not only for two polygons-- you can take seven polygons of equal area and there's one of these chains of blocks that can fold into all of them.

And I think I have a little teaser picture of how the construction goes. It's really crazy. You have this chain of blocks. It's changed in a particular way and you say,

well I really want that green piece-- so think of it like this. You just have three triangles or three pieces that you can fold. You say, well I really like that green piece to be up top. Turns out you can make all these crazy cuts. And if you fold it this way, you get the green piece on the bottom. If you fold it this way, you get the green piece on the top.

It's one hinged dissection that can move the pieces around, and basically pretend as if the hinges weren't there, and make anything you want. So it's a bit complicated, but it's really cool theory, something that we worked on for like 10 years before finally solving.

And it's practical. I mean this particular construction isn't practical, but this is another way to build transformers. Also to build sculpture. This is a sculpture by Laurie Palmer that we collaborated on. It's an interactive sculpture. You pick up gloves, and this is one chain of blocks. And the theorem is, this particular chain of blocks-- about 1,000 blocks here-- can make any shape made out of about 250 cubes. It's slightly skewed, but so there's one hinged dissection that can fold into exponentially many different shapes-- again, anything made out of little cubes.

And this is the sculpture version. We also have various prototypes of a robot that's a long chain of little cubes or cube-like shapes, that can fold itself into any 3D shape you want. It's, again, a different kind of transformer, more inspired by proteins, the way nature does it by long chains, compared to the sheet-folding, origami-inspired approach. But it's all pretty exciting and fun. Any questions? Jean?

AUDIENCE: This is a 3D version you were talking about.

PROFESSOR: Yeah, I pulled a fast one. That picture's 3D. This is about 2D. For 3D, you need the same area. You need one other condition just for them to have a dissection. It's impossible to go from a regular tetrahedron to a cube. This was one of Hilbert's problems from 1900. And volumetrically it's impossible to go from regular tetrahedron to a cube, because the angles are wrong. It's an algebraic thing. And if you make things out of little cubes, it's fine. You get that. There's no problem. But in general, there's some tricky issues. It's easier to state for 2D, which is why I did, but

it works in 3D, too, with a little extra condition. And we'll talk about that in the future. Other questions? All right, that's the end of lecture one.