PROFESSOR: Today we're going to talk about protein folding and its relation to linkage folding. We're going to look at a mechanical model of proteins. This is an example of a protein from lecture one. There's a ton out there in this place called the protein data bank, all freely available. It's really hard to get pictures like this. But you get some idea that there's a linkage embedded in here. You see various little spheres and edges. That's, of course, not reality. Those spheres are actually atoms and they're kind of amorphous blobs. The edges are chemical bonds. And those are connections. We don't know whether they're-- it's not really matter, but it's force.

This is a rather messy picture. This is what a protein folds into, some 3D shape. Most proteins fold consistently into one shape. We don't really know how that happens. We can't watch it happen. So the big challenge is to know how proteins fold. Given a protein, what does it fold into? That's the protein folding problem. Major unsolved problem in biology, biochemistry.

The protein design problem is I want to make a particular 3D shape so that it docks into something, binds to a virus, whatever, what protein should I synthesize in order for it to fold into that shape? That is potentially an easier question algorithmically and it's the really useful one from a drug design standpoint. Some new virus comes along, you design a drug to attack it and only it, you build it. Usually you would manufacture some synthetic DNA, you feed it into the cell, DNA goes to the RNA, goes to the mRNA, goes to the protein. You all remember biology 101, hopefully. We don't need to know much about it.

If you look at what's called the backbone of the protein, a protein's basically a chain and attached to the chain are various amino acids. Today I'm going to ignore the amino acids, which is a little crazy, and just think about the backbone chain. Backbone chain looks something like this. One of the challenges of video recording a class is I can only use copyright free or Creative Commons images. This one, I couldn't get one, so I had to draw it.

There's various measurements here in certain numbers of angstroms. Those are the chemical bonds. Various atoms here-- nitrogen, carbon, and so on, hydrogen. But basically, it's a chain that zigzags back and forth. You can also see the angles here. They're not quite all the same, but they're very similar. All the lengths in the angles are close. It zigzags-- this is really in three dimensions. I tried to draw the spheres so you could see the three dimensionality, but it's a little tricky. And then attached on the sides are the amino acids. I'm going to focus just on the backbone.

The way this thing is allowed to fold-- these lengths, as far as we know, are pretty static. They probably would jiggle a little bit. But you can think of them as edges. So you can think of this as a linkage. The catch is, also, the angles are fixed because the way this atom wants to bind to other things has very fixed angle patterns. If you ever played with a chemistry construction set, that's how they work. They have holes at just particular angles.

So if you think of like a robotic arm, normally-- like here, I have it two edge robotic arm, let's say. Normally, you have two degrees of freedom in three dimensions. You can change the angle and you can spin around this edge. Now, it's saying the angle is fixed-- for example, here it's, say, at 90 degrees. All I can do is spin. I'm not allowed to flex my muscle in this way. So that is the model. All of-- in this case, we have a tree-- all of the angles here are fixed. But you can still, for example, take this entire sub chain and spin it around this edge. That'll preserve all the angles and all the lengths. That's all you're allowed to do. You take an edge, you spin it-- spin one half of the edge relative to the other half.

These are called fixed angle linkages. And they have been studied quite a lot because of their connection to protein folding. So embedded in the term linkage, we assume that the edge lengths are fixed, and then we add the constraint that the angles are fixed. And the motivation is the backbone is something like-- the backbone of a protein is something like a fixed angle tree. Of course, it's not much of a tree. Most of it is a chain. There's just small objects hanging off, and if you add the amino acid there are bigger things hanging off, but still constant size. They'll have some cycles. They're not trees. But it's slightly more approximately-- I should draw wavier lines. It's a chain. Usually an open chain although occasionally a closed chain.

So we think a lot about fixed angle chain and sometimes about fixed angle trees. Now, fixed angle linkages are harder to think about than universal joints-- that's the usual kind of linkage. So we know 3D linkages are kind of tough. Nonetheless, we found lots of really interesting mathematical problems to solve here, and that is the topic of today.

At some level, we are thinking about the mechanics of protein folding. We're throwing away energy. We're throwing away the actuators in real life that make proteins fold. We're just imagining, given this mechanical model of how a protein might fold, what's possible. So in some sense, it's broader than reality. And the hope is you find an interesting algorithm for how to fold these protein chains. And maybe that's the algorithm that nature is implementing. That's the kind of general picture. We're not constrained by reality, and by how nature actually folds things.

So I'm going to talk today about four main problems here. The first one's called span. Second one's called flattening. Third one is flat state connectivity. And the fourth one is locked, our good friend locked chains. And of course there are locked chains because we're constraining linkages even more than before. So you could take knitting needles, it'll still be locked. So you add extra constraints. Makes it harder to fold. But there are actually some interesting positive results we can give of chains that are not locked in some sense.

And flat state connectivity is about the same kind of thing, where instead of worrying about getting from anywhere to anywhere, we just worry about getting from one flat state to another flat state. Flat means lying in a plane. Flattening is about is there such a configuration. And span is about given robotic arm-- like a more complicated one, like with multiple edges-- how far apart can the endpoints get, and how close can the endpoints get. The universal chain is not very exciting. Farthest it can get is when it's straight, and the least far it can get is when it's closed. You can always do that, I think. Well, no, I guess you can't always close it up. That's a little nontrivial.

But for fixed angle linkages, you can't straighten out because you have to preserve the angles. So it's kind of what is the straightest like configuration, given that the angles are fixed.

So let's start with span. So the span of a configuration is the distance between the endpoints. And in general, you'll find the max span and the min span. This search was begun by a guy named Mike Soss, who was a PhD student at McGill. And he proved that if you want to find, for example, a flat state that lives in two dimensions with the minimum or the maximum span, this is NP-hard. This is in his PhD thesis. Question?

AUDIENCE: If you have a linkage or [INAUDIBLE] chain that actually loops around, is there a span because there is no endpoint?

PROFESSOR: Oh, here I'm assuming open chain-- I should say that-- which most proteins are. I've been talking about trees and stuff. Here I mean chain, otherwise there aren't two end points to think about. Good. So here are his NP-hardness proofs. [INAUDIBLE] the problems are NP-complete. They're pretty simple. The problem here we're reducing from is partition. I give you a bunch of integers. I want to divide them into two halves of equal sum. And the top example is minimum flat span problem.

So you make an orthogonal chain where the horizontal edges are long and they're proportional to the integers you're given, the vertical edges are really tiny. And so what you'd like to do-- all you can do is sort of flip because you have to stay in the plane, you can flip one of the vertical edges, say, and make any of these edges go left or right. You get that freedom. So each integer, you get to choose. Do I go right by that amount and or do I go left by that amount? And if the amount you go left is equal to the amount you go right-- in other words, is it partitioned into two equal sums-- then those endpoints will be aligned, and then their distance will be very tiny. Otherwise, it will be quite large because the horizontal distances are all big. So it's kind of a very easy NP-hardness proof.

To maximize your flat span, instead of mapping your integers on to lengths, you map them on to angles-- return angles. I won't specify that too precisely. But again, if you make your total counterclockwise turn equal to your total clockwise turn, then the two end edges, which are super, super long, will be parallel. And to maximize the distance between the endpoints, you want them to be parallel. If you make them go some other angle, they're closer. Now, both of these proofs rely on the requirement that you want a flat configuration with minimum or maximum span. Now, there's a claim that flat configurations matter for proteins, so it's a natural constraint. But what about the general problem? What about, I have something in three dimensions, I want to maximize-- I have a fixed angle chain in 3D, maximize or minimize the span?

Both of those problems are open. Can you solve them in polynomial time? For 3D max span, so the non flat version just for maximization, there's been a lot of work. And there are two papers on the subject. One of them is by Nadia and Joe O'Rourke. Another one is by Borcea and Streinu. And I just want to quickly summarize that because there's a lot of stuff there. But essentially, they find what the structure of those spans look like. I have an early figure that's in our book before all this work was done.

The simple chain, this black guy at 1, 2 3, 4 bars open chain, and in that black three dimensional state, it maximizes the span, the green span there. And if you look from above, which is this picture-- of course, the end points look much closer in projection-- and the red configuration is the max span if you restrict to flat configurations. So here, of course, 3D buys you something. In general, it always will. An interesting thing is that this max span-- the green line-- passes through another vertex. It seems kind of weird. And in fact, there's a general theorem there sort of characterizing the structure of these chains. It's still not known whether we can solve this problem in polynomial time.

But for orthogonal chains, where all the angles are 90 degrees, we can solve that in linear time, I guess. And here's what it looks like. Suppose you have some orthogonal chain. Orthogonal chains are nice because you can draw them in the plane as a staircase. So there's a nice canonical configuration.

One way to think about how to find the max span configuration-- I'm just going to give a high level overview here, this won't be a complete algorithm-- is you triangulated that staircase in this sort of obvious way of connecting every endpoint to the one, two ahead. And think about this as like a body that's hinging around here because I can spin-- if I spin the left part of this chain around this edge, it's like hinging that triangle around that hinge. Same thing. You could think of these triangles as just being hinged together, like in rigid origami. It's the same class of motions.

And now you can-- what I'm going to do is compute a shortest path in this surface from here to here. Confusingly, this is called a geodesic shortest path although it's not really related to geodesics from polyhedral surfaces. But if I compute a shortest path, it's going to go like to this vertex and then probably to that vertex. But I'm constrained to stay inside the union of those triangles. I want to go from one endpoint to another.

Then I claim that-- OK, these two edges will stay planar, of course they form a triangle-- I claim these four edges will stay planar, and in the orthogonal case they'll stay zigzag. And then also these two guys will stay in their own plane. And then I claim that actually this wiggly line, which is not straight because it bends here and it bends here, the total length of that wiggly line is the max span.

And you achieve that by folding this planar part with respect to this planar part with respect to this planar part so that the wiggly lines become aligned and straight. And that's very hard to draw. But it can be done, and that's what you do in the orthogonal case and that gives you the answer in linear time with enough work. For non orthogonal though, it's open whether you can do this in polynomial time. Maybe it's NP-hard, actually. I don't know. That's all I want to say about span.

Next, we go to flattening. The first question about flattening, and the main one we'll talk about here until we get to flat state connectivity, is does a fixed angle chain have a flat state at all? Can you even draw it in the plane without crossing? So we're restricted here to have no self intersections. We want flat state, no self intersection.

Then there would be a question of given some configuration, can I actually continuously get to a flat state? But the simplest question is ignore getting there. Just, is there a flat state? And this problem is NP-hard. Again, Mike Soss and his advisor Godfried Toussaint.

It's a little more complicated, but it's basically the same idea as that very simple proof, which was just to map integers to a little zigzag staircase here. So the goal is to force x to end up being-- the two endpoints of the green curve-- to be aligned with each other. That will exist if and only if there is a partition of given integers. And there's all this infrastructure that's sort of-- there's little lock here and a key, and some structure on the left. Basically forces the picture to look like that.

So the first claim is that the black stuff is basically unique. I think there's one global reflection you can do that doesn't affect anything. But you try any of the other flips. Again, we're restricted to flat states here. So there's only sort of a bounded number of things you can do, a finite number of things you can do. You try all of them, they self intersect. So the black thing is basically forced, and it forces the endpoint-- this endpoint x-- from the black side to be aligned with this very narrow spike. And because the angles are preserved, that red guy's going to be vertical. It can't go down so it must go up.

And so only if this thing is aligned in the center, aligned with x-- in other words, this problem has a partition-- will this have a flat state. So it's not the most exciting example. This is only a weak NP-hardness proof. Lots of interesting questions still open here, like if all the links are the same, if they're all equal, then we don't know. Or if all the links are even polynomially bounded, this needs really, really long lengths verses really, really tiny links exponentially-- exponential in ratio. All these problems are open. And that's flattening. So we're going very quickly because there isn't-- well, partly because I'm more excited about this-- but there's more work in these two parts. So I'm going to focus on that.

Next topic is flat state connectivity. So the idea is to think about the configuration space of these fixed angle chains, let's say. And we kind of know that it's going to be disconnected because there are knitting needles, there are nasty things. So there's maybe various connective components. But let's say that we really care about flat states.

And the question is, are they connected to each other? So in other words, do all the flat states-- mark them with x's-- do they all appear? There's only finitely many. So configurations, there's this continuum that there are these messy blobs, semi algebraic sets. The flat states, those are discrete things. Because we have fixed angles, you can flip or not flip every edge. So [INAUDIBLE] most exponentially many of them, so finite. Are they all in one component? So I can get-- if I pick two of my favorite flat states, there's a path between them? Or are some of them in multiple components?

So in this case, we call it flat state disconnected. And if they're all like this, we call it flat state connected. And we'd just like to know which chains, which fixed angle trees, whatever, are flat state connected versus flat state disconnected. I would say, the big open problem here is every fixed angle chain, open chain flat state connected? That is still open. We have lots of results in that direction.

So the top four results are about open chains, but they have an extra constraint. For example, open chains that have a monotone configuration, like the staircase. Those are flat state connected. In fact, whenever the angles between the edges are either orthogonal or obtuse, then they're flat state connected. When the angles are acute, we're not really sure. If all the angles are equal and acute, then we can do it. But if they're different and acute, we don't know. Unless the edges are all unit length and the angles are in this funny range, then we can do it.

So there's all these special cases we can solve. The most relevant to proteins is actually obtuse chains, so we've solved sort of the main problem with this second result. But there's a natural theoretical question here is, are all open chains flat state connected or do we get disconnectivity? I will show you that-- I'll show the orthogonal case in a little bit.

We can do some stuff if you have multiple chains that are attached to some blob like a cell. Closed chains is a little bit-- for disconnected, we don't have very interesting examples, I would say. This is funny because locked examples are easy to come by but flat state disconnected examples are little trickier because flat is so constrained. So let me just show you these examples.

This is what we call a partially rigid fixed angle tree. So not only are the angles fixed, but also the black edges are not-- in fact, only the blue edges here are allowed to spin. Everything else is held rigid. So these arms are somehow forced to be in exactly that geometry. I can spin it around this edge, so spin it up into 3D, for example. These are two different flat states of the same linkage.

The only difference between these two-- I haven't rotated or anything-- is that I've taken each of these arms and flipped it around a blue axis. If I do all four of them, I would get this picture. But the claim is, you cannot do that without self intersection. The intuition is, when there aren't very-- oh, one other thing that makes it slightly more interesting. It's weird to say, well why did you force some of the edges to be rigid and not others? One way to force that is to use a general graph. If you add some extra edges to sort of brace this and all these angles are fixed, then this linkage will behave exactly like that one. So that at least is somewhat more natural, although what we really care about are chains, maybe trees. But we don't know whether there's a-- we also don't know whether all fixed angle trees are flat state connected. These are the worst examples we know.

Let me give you an idea of why it doesn't work. This is a little animation of just a couple of moves attempted, and it's just going cycle through that. And these are some static images of the same kind of thing. So the intuition is the following-- you have four arms. You have two sides to the plane, there's up and down. The four arms and two sides, at least two of them are going to have to go to the same side. The best you can do is two and two, or three and one. But in either case, you have two sides go to the-- two arms that go on the same side.

Now, it could be, like in this image, that they're opposite arms. So there's this arm here and there's this arm here. So they're connected by a 180 degree angle. And those guys, when they fold up, actually these edges will just hit each other dead on. So that's kind of obvious from a geometric standpoint. Maybe you call it cheating for them to hit dead on. You can twiddle the edge lengths so that they will properly intersect without dead on collision, without being degenerate basically. The alternative is that-- and this is a little harder to see geometrically, and that's why we drew that animation-- is that you have one arm and you have an adjacent arm connected by a 90 degree angle.

Now here, there's clearly some collision going on. And if you happen to fold it up 90 degrees like that and then fold the other guy, obviously you get stuck. But maybe you fold it a little bit and the other guy goes a little bit more and there could be some dance between those two degrees of freedom, those two arms, that somehow gets them both to pass over to the other side. It's obviously not possible. How do you prove it? Well, you could prove it with topology-- knot theory or link theory. So it's a very cute proof.

You start with-- so here's the full example, but I've highlighted the two arms in red that are going to move. And I imagine connecting the endpoints of each arm with these little blue ropes underneath the plane. They're both going on the same side. Let's say they somehow pass through each other on the top side. Then I'm free to connect stuff on the bottom, and I shouldn't collide with that. So if somehow, both of these guys flip over-- so arm on the left, A3 flips over. A3 stays where it is but now the arm is on the top, the north side instead of the south side.

And the other guy, from B to B3, used to go like this and now it goes like this. If that happens somehow, then these ropes could remain intact during that whole motion. On the top, you have two closed loops that are not interlocked. On the bottom, you have two closed loops that are interlocked. So there's no way to get from there to there without colliding somewhere. The blue stuff didn't move, so the red stuff must have collided. So even just topologically, you are screwed. That is their only negative example. Lots of interesting open questions here.

On the positive side, let me show you for orthogonal chains-- and the same algorithm works for obtuse chains, all the angles are obtuse-- how they are flat state connected. So in order to show it's flat state connected, I want to think about two flat states and show that I can fold from one to the other via some intermediate 3D stuff. Let's start with one of the flat states.

So it's orthogonal. So in two dimensions, all the edges will be horizontal or vertical. In 3D, they can kind of be in many, many different angles, many different dihedral triangles. In 2D, it's pretty simple. And all I need to do is sort of pick up that chain, and I'm going to try to pick it up into a staircase because there's only one staircase. If I can make it a staircase, I make flat configuration A a staircase, flat configuration B a staircase, and just FedEx in the middle. Once they're both staircases, I play one motion and the other one backwards, get from anywhere to anywhere.

So here's all you do you. You take the first edge and you just rotate it up to the red line A. And then you take the next edge and you take both of those edges, and you just rotate them like this, so you get that little 2-step staircase. Now I'd really like to pick up this edge, but I want to first get these two edges in a plane with that edge. So I rotate this flag over to the left, I get those two guys. And now they're in a plane with this, and I just lift that up. Then I'm going to flip, then rotate up. Flip, rotate, flip, rotate.

Here's some more examples. So if at this point, I have this staircase-- sorry, I guess originally I have from V3 to D up there. it's not in plane with this guy, so I just rotate it like that. I'm spinning around this edge. So now I have from B3 to #, and then I rotate it up along that green arc. And I get a bigger staircase above the chain and because everything's staying above, it will never penetrate the plane and will never hit anybody else. And I'm building a staircase by design. I always rotate this-- there's actually two ways I could be in plane-- but I always rotate it so that when I pick an edge up, it'll be in a staircase. So this is actually really easy.

And slight generalization is to obtuse chains, then instead of making a staircase, we make a monotone. Let me get this right. Yeah, sum z monotone state. So it goes monotone and z, out of the plane, that's enough to avoid collision, and you get a canonical configuration. Also, if you have acute angles but all the angles are equal, then there's a natural conical state, which is just like a compressed staircase. And that will work here, too. That takes more effort. That was in a separate paper. But big open question is, chains with arbitrary angles. We have no idea. It is very hard to do an operation like this. Wow, we are burning through this. This is fun.

So the next topic is about locked chains. Now as I said, you can take a knitting needles example, which has five edges. And that will still be locked if you force the angles to be fixed because it was locked without the angles being fixed. Now, it required a length ratio of 3:1, I think. This edge had to be longer than the sum of those three.

So let me put down some open problems. So you may recall in the case of universal chains-- universal joints, I should say-- the big open question was, can you lock a universal joint 3D chain with unit edge lengths? So, equilateral-- every edge is the same length. Is there a locked chain like the knitting needles when all the edge lengths are the same. And one of the motivations for that is in proteins, the edge lengths are all within like 50% of each other. So it's pretty natural, of course. We don't have universal joints with proteins. We have fixed angle joints.

So the big open problem for fixed angle joints-- I guess we'll do this in parts-- is there a locked 3D fixed angle chain that's equilateral? I'm going to add some conditions here. So that's the first natural question. Knitting needles doesn't suffice. We need a 3:1 length ratio, as far as we know. Turns out that question's not very interesting. I need to do slightly nonlinear editing here.

So you take your knitting needles example, and you just subdivide the edges into lots of little tiny bars. It doesn't have to be this extreme. You could not subdivide these edges at all, and make these guys subdivide them into like three or four parts. Because the angles are fixed, these guys act as a single [INAUDIBLE]. There's really no difference. Maybe you make a slight curve there and then they can bend a little bit, but really not much. So if you just say, oh, I want it to be unit length. I don't constrain what the angles are but I fix them, then it's trivial to come up with locked examples. So that's not very interesting.

What if I make it not only equilateral-- the lengths are the same-- if I make it equiangular. Because, again, in proteins, all the angles are similar. They're around 110, 108, something like that. They're all pretty close, I think within 10 to 20% of each other. Well, there's also a locked example. And just to show you how research was done back at the turn of the century, this is pre-web 2.0, pre-Ajax and all that fancy stuff. We used Ascii Art. Email was the tool of choice. I know it's hard to imagine a time-- 2002, so long ago. And I tracked this down. This is the original claim it looks-- we call this the crossed legs example because it's like two legs crossed around each other. And this is the first time we thought, oh, maybe it could be done, unit length. This is Stefan Langerman.

And here, for the first time ever-- this is not the first model, but this is the first photograph of any model I'm aware of-- this is the crossed legs example. This is made with a construction toy that used to be sold around here but is no longer in production. So they're pretty hard to get. It's straws-- nicely colored straws-- and the cool part are these connectors.

So the connectors force particular angles. In this case, every angle is 45 degrees. So this is equiangular and equilateral because all the straws, I'm told, are the same length. That's how they're sold. And you can do edge spins. Whoops, that's called cheating. It's not totally obvious that this is locked. The problem with the model is that the edges can bend. But if you treat it properly and only spin around the edges, then you're stuck.

Now, there is one thing you can do. See if I-- yeah, like this. So here, I'm almost in a plane. I've got the purple edge right against the pink one. Easier to see from that angle? I don't know. So here, this guy can come out and this guy can barely go on the edge. So actually, this doesn't quite work for equilateral. It works for one plus epsilon. That's why I added these little nubs at the end. So if they're all exactly equal length and you allow just abrasion of the endpoint, then this could go around like that and then you'd be unlocked.

But if you just add slightly-- either you change the angles to be not quite equal, so make this a little smaller, or you make the lengths a little bit longer at the ends-- then the claim is it's locked. We don't actually have a formal proof of this. We're just remembering, hey, we should probably write this up. I was talking to Stefan last night. So someday we'll prove that this is locked. But it certainly looks like it. So this isn't open yet. I mean, modulo the details of that proof. Equilateral and equiangular seems easy to lock with fixed angle chains. In fact, even easier, this example only has four edges. So even less than the knitting needles.

Fixed angles make for complicated motions, I guess. Make it hard to unlock things. So I need to add one more constraint, and the constraint is obtuse. So again, all of these properties are enjoyed by proteins. Protein backbones have all these properties. Even if you looked at fixed angle trees, is there something like this that's locked? And now, we don't know. And this seems quite tricky. I guess the intuition is that obtuse-- and usually we think about orthogonal, just cause it's easier to draw the pictures, but reality is more like 108 degrees-- the conjecture is obtuse fixed angle chains behave kind of like universal joints. And with universal joints, we don't know whether equilateral is enough. So it's tricky. Yeah, question.

AUDIENCE: In the previous example, you showed the ribbon thing--

PROFESSOR: The subdivided.

AUDIENCE: Subdivided into a bunch of little interconnecting pieces. What if you, instead, made your ribbon lengths basically a bunch of little unit obtuse angle connectors, and then when you hit the big terms it's just obtuse, obtuse, obtuse, obtuse.

PROFESSOR: Yeah, you can make this example be entirely obtuse. You can make every angle obtuse. Here, you could arc a little bit. Here, you could arc some more, but not too sharp. And because here, we actually know that this part can be made a string. We don't really care what it looks like. So you can make it fairly obtuse. It's just that these guys should not bend much. They have to be long no matter how you fold them.

So if you want equilateral and obtuse, that's also easy. But to make all the angles actually be equal, as far as we know you cannot take that knitting needles subdivided. Make all the lengths equal, and all the angles equal, and make them obtuse. That's open. But any two out of the three, it's easy. Of course, in reality they're not quite equilateral. They're not quite equiangular. But it's still open for those. If you have like a small range for the lengths and a small range for the angles, this is open. We pose it this way because it's the cleanest geometrically.

But the real question you care about is when these are fuzzy constraints. Obtuse is real, but these guys are fuzzier. So if you think about proteins, which fold very well in nature, there are a couple of reasons they might fold well. We know, as far as fixed angle chains go, it's actually quite easy to find locked examples. And, this is somewhat intuitive but bear with me, because they are locked examples in this configuration space, we believe these configuration spaces are really ugly nasty. So it would be very hard-- even if you know, oh I only need to fold something in my component-- if these guys are highly disconnected and flat states are all over the place, it's probably even within this connected component, it looks really ugly.

And so it's very hard to find a path from one state to another. Probably pieced based complete, although we don't know that for sure. But that's the intuition. Locked equals messy. When there are no locked configurations, like carpenter's rules, we get really nice algorithms. It's super easy to get from state A to state B. Now, if you're nature or you're designing nature, let's say, or you're building your own virtual world, Second Life, and you want to design proteins, you would like to design them in such a way that they fold easily because it happens all the time.

Every thing that is being acted on by our body, every living thing that we know has tons of little proteins that are doing all the work. They are folded into their shape and they do something. That's proteins plus RNA, but mostly proteins. So to understand life, we should understand proteins. Now, how to proteins fold so well when we know there are all these locked configurations? One possible answer is that proteins have extra structure, namely these three things, which somehow make it very easy to algorithmically go from A to B. Notice I'm not assuming anything about how proteins fold in terms of what is the mechanism that drives them because we don't really understand those mechanisms. There's hydrophobia, which we don't really know how it works.

So all these little forces that we don't fully understand. We understand lots of parts of the story, but not the whole story. And what's convenient about these kinds of problems is you don't need to assume anything about how it actually happens. All we're assuming is the mechanical behavior of the proteins, and how they could possibly fold. And the idea is if there's locked configurations, that's probably the wrong model because then everything's messy. Now there's also evolution coming into play and maybe some proteins are easy to fold, some proteins are hard to fold. That's an interesting question which should be experimented with. But let's hope that there's a model.

Things are mutating randomly. You really like everything to fold nicely. Maybe it's because you have all three of these properties, approximately, in real proteins. So general idea is that nature has some extra constraints that make protein folding easy. Just have to figure out what they are and why it makes them easy. Unfortunately, this is still an open problem. If this had an algorithm, that would be a natural candidate for what nature's doing using its mechanical-- or using it's energies and forces, and so on.

This would be a rather unsatisfactory ending if this was-- if the climax was an open problem. We have a theorem too. And this is what I'll cover in most detail. And it's a paper called producible protein chains. Protein chains just means fixed angle chains, open chains. And the idea is well, yeah, there are these constraints or there are these extra features. We don't know how to exploit them, so let's not even worry about them. Suppose they don't even exist. Maybe I'm going to assume obtuse, but none of the others.

There's another constraint in how proteins fold, or really how proteins are created. They're created by a machine, a molecular machine made up of a whole bunch of proteins and RNA, called the ribosome. You may have heard of. It translates messenger RNA into proteins. So there's some mRNA around here, maybe. Don't know exactly how this machine works. But there are actually very accurate three dimensional reconstructions of the ribosome.

With no copyright free images, you're going to have to-- there's a link on the slide that goes to the cool and 3D models of the ribosome, with a slice away. So you can see there's a tunnel down here, and the protein get sort of created here. The background gets created here. And it starts going through this tunnel. There's a bend in the tunnel around here, where it's conjectured and an amino acid gets attached. And then it goes out the tunnel and the protein starts spewing out here and presumably folding at the same time. We don't really know. So this is how proteins are created. The birds and bees, I guess, of proteins.

So what's interesting about this is it's not like a protein exists and then folds, which is how a lot of people might think about it at first glance. That's the natural way to model protein folding-- start with a protein, say, and just zigzag configuration. If it's obtuse, there's a nice zigzag monotone configuration. Then you see what is the best configuration I could fold into, for some notion of best. And that's sort of what this configuration space picture is about.

It's if I already have a protein, what configurations can I reach by motions? And that is interesting. That's important because you're still going to have to reach by a motion. But, it's actually more flexible than that because the protein could just be partially built. The rest of the protein hasn't been built. And it could start folding already. It might be easier to fold when you don't have the obstacles of your existing protein. So that's both a worry, but it's also a convenient structure because this ribosome is a giant obstacle. Bigger than most proteins. If your protein's really long, maybe it could go over here.

But most the time, it's going to stay on one side of this plane because locally, this thing is basically flat, if you look at the real 3D pictures, not the schematic. Now, this is good news for a geometer because there's this giant obstacle-- think of it as a half space-- which the protein cannot penetrate while it's being produced over here.

That's it. That half space constraint is enough to get really good algorithms for folding your chain. It's weird because we've made a problem both harder because the protein is only partially produced at any time and it can fold, which is part of it, but we've also made our life easier because there's this big obstacle.

AUDIENCE: [INAUDIBLE] makes sense why there's only obtuse angles there, right?

PROFESSOR: Yeah, right. Out of this, we're going to get that the angles and the protein are constrained. And in particular, for this angle-- it depends. I mean, in this picture because it's perpendicular here, yeah, the sharpest angle you could make is 90 degrees, more or less. That's a good point. So it's a convenient match between the chemistry, which also forces the angles to be obtuse, I guess. I don't know a ton of chemistry. But also, the ribosome just geometrically forces. We're going to use a property like that.

Our model is going to be a little bit more-- both more general and simpler. We're going to imagine that the ribosome is a cone. It's part of the upper come here. This is like a mirror image. And in reality, that cone is actually a plane and everything above the plane. But to be more general, we're going to allow some angle alpha here. It's also just easier to think about when alpha is smaller than 90, but everything I say will work when alpha equals 90 and that is sort of the reality case.

So the model is-- so the ribosome is a cone. We call this the half angle of the cone. From the vertical axis to the edge of the cone is alpha. So if you were going from one axis to the other, it would be 2 alpha. The model is, you start with one link of your chain, which is inside the cone. It's spews out through the apex. That's the exit of the tunnel. Here, we're allowing the tunnel to be actually quite free. Doesn't have to be perpendicular to the apex or the plane of the apex. So the edge comes out, and as soon as the endpoint of the chain reaches here, then a new link is created.

This is like a very simple model for how a chain can come out of a cone without worrying about what's happening inside the cone. Imagining everything's totally free. This is like you can allow self intersection in the cone, who knows what. But once you come outside the cone, you're not allowed to self intersect and you're not allowed to intersect the cone. Once you come out, you can't go back in. So that is a model of producing protein chains. And if you have a cone of angle alpha, we call this an alpha producible chain. For whatever reason, we often call it beta producible chain. Just change the variable.

So if you think of the ribosome as a cone with half angle beta, you can produce it like this. That is beta producible. Now this is a pretty powerful model because you only have to worry about it link by link. You don't have to worry about the rest of the chain until it spews outside of the cone. But it's restrictive in that you cannot penetrate the cone. All right.

One thing we can talk about is angles. So I'm going to write call a chain a less than or equal to alpha chain if all the turn angles are less than or equal to alpha. I don't know if I've used turn angles in this class. Probably. If I have two edges, the angle would be this. The turn angle would be this, the supplement. Yeah. I guess we used turn angles way back in origami land, Kawasaki's theorem and so on. It's just, if you're going straight, how much do you have to turn to get to the next edge.

So we'd like fairly obtuse things. So alpha is going to be small. There isn't a ton of turn. But in general, less than or equal to alpha chain for some alpha. Now there's a relation-- as Jason was mentioning, there's a relation between alpha and beta in the ribosome because you always exited orthogonally to the plane that was your cone. The sharpest angle you could get was 90 degree turn angle. Here, we're a little freer because this edge can wiggle around as long as it touches the apex.

So if you're up against the cone, you have to slide out into the complimentary cone-- that was the previous picture-- and as soon as you get there, you could create a new edge which is like this. So the sharpest angle you can get is actually twice beta. In general, we're going to have alpha over 2 is less than or equal to beta. That is-- you can get up to beta equals 2 alpha. Get that right. And also in the obtuse case, this is not too exciting. But it's true.

There's actually some problems here. When you have that full flexibility and you set alpha to two beta, not the other way around. I'm going to assume here that alpha equals beta. This will be convenient. And it's the interesting case because, in reality, the cone has a half angle of 90 degrees. So beta is 90. And the sharpest angle we're going to make was always obtuse. So saying that you have a less than or equal to 90 chain is just fine. But on the mathematical side, I think we saw the case when alpha is less than or equal to beta, but not when alpha over 2 is less than or equal to beta. That's a weaker constraint. So there is a range where it's not so easy.

Now, what do I claim about these chains other than their angles are not so sharp? I claim they're good algorithms for folding them. What could I possibly mean? There are still locked configurations. Is that true? Well, I mean presumably-- this is acute-- but you take the obtuse versions of this guy. Because I didn't constrain the edge lengths or anything, I just said that the angles are obtuse.

So I could just sort of round these corners, make it obtuse. You know, add lots of dots just at the corners that would be obtuse. And a chain like this will be producible. A chain with these angles and these edge lengths can be produced from a cone. But this configuration of this chain cannot be produced, I claim. I claim anything that can be produced is in one connected component. So while I can make a linkage that is locked and that there are bad configurations you can't get out of, the things you can actually make, you can always get out of.

So there's going to be the space of producible configurations. Maybe there's some stuff that's unproducible but still connected to it. I don't know. It doesn't matter too much. I won't worry about this stuff. There's other bad locked configurations that cannot reach here. But everything that's producible is in one connected component of the configuration space. That's property one. That's kind of nice. Also, all the flat states are going to be in here. This is actually pretty easy. I just need to prove that flat states are producible, which we'll worry about later.

So in particular, these guys are flat state connected. All the producible protein chains are flat state connected. That's interesting because we don't even know that all chains are flat state connected. But here-- I guess we know that obtuse chains are flat state connected, so maybe it's not so surprising. But what's important is not only are the flat states connected to each other and the producible states are connected to each other, but producible is connected to flat states. Everything is together here. I might have more properties. But that's already some good news. And there's algorithms to do all of this.

How do we prove it? Well, as usual we use the FedEx method. And in some sense, one of the challenges is what is the natural canonical state for protein chains? In fact, we're just going to assume that our chain is-- in reality, we're going to assume that it's an orthogonal chain-- an obtuse chain, I should say. But in general, for any less than or equal to alpha chain, for whatever alpha you like-- and it will be the half angle of the cone, so alpha equals beta-- we will define a canonical configuration. I think we called it the alpha CCC.

So it's going to be kind of like a helix. I think I have an example, an actual computed example. That's not the best picture because you can't see everything that's going on. But this is an actual canonical configuration of a particular chain. Let me tell you how it works in general. So in general, we have some chain, v1, v2-- sorry, starting at v0, v1, v2. I want to define a canonical-- and there's defined lengths between the two. And there's defined angles between every triple in sequence.

So I'm going to start with v0 somewhere. Doesn't really matter by translation. Say, the origin of space. And what I'm going to do is draw a cone whose apex is that v0. And the half angle here is going to be alpha over two, not alpha. This is a smaller cone than-- by a factor of two-- than the ribosome. That's important. So there's this vertical line. And to pick v1, I'm just going to use the right edge, which is-- let's say this is the x direction.

This is the z direction. Maximum x-coordinate. It's going to lie on the cone, maximum x-coordinate. That's v1. Now, v1-- let me redraw this picture a little lower. So there was v0, v1. Now I want to draw v2, and I want to draw it above. So what I'll do is draw a vertical column whose half angle here is alpha over two. I want to draw v2 on the cone here. Of course, the height of the cone is the length of the edge, not the height. You could think of the cone as infinite.

And then I just clip this to when it has the right length. So again, this might be a different height cone. I clip it to whatever the length v1, v2 is. I want it to be somewhere on this cone, but now I'm constrained to have the correct angle at v1. I can't just put it over here, because then the angle here would be 180. Presumably, I don't want to make a 180 degree angle. So in reality, what happens is that there's a cone which is-- whose axis is the edge v0, v1. So I extend v0 v1 out here, which, in this case, happens to lie here. And I make a cone like that. Let me draw it slightly more accurately.

In this case, the center axis of the cone would go right here, whatever the extension of v0, v1 was. And to have the right angle at v1, v2 must be on that cone. Conveniently, there are two intersection points between those two cones. I could choose either one of them to be v2. And I will choose the counterclockwise most one, which is this one. This is going to be v2. So I draw that edge. Now I repeat.

So for v2, I'm going to draw a vertical cone whose half angle-- here, the half angle is always alpha over two. Half angle is alpha over two. This cone had half angle, whatever the angle v0, v1, v2 was. Was that the half angle or angle? The half angle. I think this is right. OK. And then I take the intersection of those two cones, and that will give me where v3 is.

So I do the same thing for v2, for v3, and so on. I have a unique choice at every moment. Yeah, basically. And the only exception was at the beginning here, when I had a vertical line. These two cones could actually be equal, and then the intersection is the entire cone. And that case, I guess I'd choose the maximum x-coordinate one again. And then I have a canonical choice of everything along the way. It will always go up. And it sort of spinals around because of the counter clockwise most choice. And the result is a picture like this. Anything else I need to say here?

I claim this canonical configuration lies in an alpha over 2 half angle cone. That's true by construction. The challenge is, does the construction really work? So I start, obviously, with one cone, and I can think of this as actually an infinite cone that goes out to infinity here. And I claim the entire construction will lie inside that cone. And it's kind of obvious because v1-- I chose v2 to lie in the same cone, just translated up to start at v1 instead of starting at v0. Of course, this cone is contained in this bigger one.

And by induction, in fact, the entire rest of the chain will lie in this smaller cone. Therefore, it lies in the big one, also. OK. Fine. So by construction it will lie in alpha over two cone. The worry is that these two cones don't intersect. Here we have an angle. The half angle of the cone is whatever angle the angle is at v1. Sorry, this should not be the angle. This should be the turn angle. That angle is how much you turn from v0, v1.

Now, we know that-- we're assuming that-- the turn angles are all, at most, alpha. The turn angle cone could actually be twice as big as the vertical cone that we were always using. We always use a vertical cone, half angle alpha over two. But it's OK because we always keep these edges, like v0, v1, was on the edge of the cone. And so when we extend it, it lies on the edge of this vertical cone. So its angle-- in the most extreme case-- its half angle is alpha, which would look like this.

We'll go all the way over from the right side of the cone to the left side of the cone. In general, it's not going to be right and left, but it's going to be some side and the antipodal point. And because the double angle of the cone is alpha, it's still OK. You will intersect somewhere on the cone. This is a subtle detail, but it's really crucial because we start with a chain that has relatively large angles, alpha.

And we get it into-- we squeeze it into-- a cone that still has double-- twice its angle is alpha, but we kind of compress it into something of half angle alpha over two. You might think, oh, I'm just changing the definition and calling it half angle. Therefore, it gets to an alpha over two. But it's a little tricky to actually get it to fit in a vertical alpha over two cone. Once we have this, it's really easy to canonicalize a chain, a producible chain. So let me tell you how to do that.

So we're going use the FedEx method of taking some configuration and canonicalizing it, and then uncanonicalizing to something else. We're also going to use a new method, which I just came up with the term. Is called the momento method which is you play the movie in reverse. So, I guess, also the Merlin method. That's more complicated. So we have a movie here in mind, which is how was the chain produced?

So what I want to show is that if-- I want to start with a producible configuration and chain. Somehow, it got produced. So you had your cone and the thing starts spewing out and folding, and doing whatever. That's an animation, in some sense, of one edge coming out and stuff is folding at the same time. Then an edge is created, then another edge comes out, and so on. What I want to do is play that movie backwards. It's a pretty intuitive idea. I just want to start feeding the edges back into the cone, and just keep stuffing them in.

Now, what happens out here is easy because we know it doesn't penetrate the cone. That's the assumption. And we know whatever was created here could be uncreated, as long as you can afford to erase edges one by one. That's the tricky part. How do I erase an edge? And usually, I can't. But I don't have to erase any edge. Like, if I had to erase this one, that would be hard because some motion here might penetrate where that edge ought to have been. And erasing the edge will make it-- adding the edge makes it harder to fold. So I can't erase it.

But the edges I have to erase are the ones that have been fully inserted into the cone. So if I can somehow do something inside the cone, I would be OK. All this work, defining a canonical configuration, was about forcing a chain to stay inside a cone-- and not only an alpha cone, which is what we're going to have as the ribosome, but an alpha over two cone. This is smaller than my ribosome cone by factor of two. And I need that.

Why do I need that? Because this thing, I have no control over the outside chain. So the way that it approaches the cone could be as sharp as like this, where I have the first edge that is-- the current edge that is-- inside the cone. As far as the movie is concerned, there's only one edge. There's nothing up here. I want to put something up here, in a cone, naturally. But I don't get to control the first angle because that is controlled by this motion. Maybe it really needed to go sharp like that so that it could make a sharper angle or whatever.

So you might say, well, of course I can put it inside and alpha cone, which is the same as this alpha. Sorry-- bad picture. This is alpha. This would be a problem because I have this weird angle coming in, and now suddenly I have to bend back like that. Maybe the turn angle here is not so sharp as alpha. Maybe it's one degree. So I really can't force the rest of my chain to lie in this cone. But I claim I can force it to lie in this cone, with half angle alpha over two. This is a little more subtle. Oh, right. I wanted to mention that helices appear in nature. They appear in proteins, but they also appear in this crazy climber plant. Marty, have you seen this in Costa Rica?

AUDIENCE: Yeah.

PROFESSOR: Yeah. There's some really incredible wildlife in Costa Rica. I've never been there, but I've seen lots of pictures and this is spirals in practice. Also, proteins tend to form these things. They call them out alpha helices because they spin like an alpha, I guess. So it's kind of neat that the canonical configuration, which is totally geometrically motivated, also appears in biology. Not that kind of biology though.

So here is the picture. I have the big cone. That is my ribosome, and here I'm going to write beta for the-- beta for big, I guess. And then we know that we can canonicalize a chain, or there at least exists a canonical configuration of the chain, where everything lies inside a cone of half angle alpha over two. Now the problem is that cone, we want it to be at a funny angle. Let me draw a real picture.

Here's a ribosome. It has a big angle here. We'll call alpha. Now, I'm going to do the not extreme case, try to be a little more general. There's some edge that is currently entering, and we have no control over that edge or the rest of the chain. That's determined by the movie which we're trying to play backwards. What we have to control, and what we're free to control, is the rest of the chain because as far as the movie is concerned, that hasn't been created yet or it's already been destroyed, depending on whether you're playing forwards or backwards.

We need to say what happens to it. And what I want to happen is so that by the time this edge is inside, that edge plus the rest is in the canonical configuration. If I can achieve that, then as edges come in they become canonicalized, and then everything will be canonical and inside the cone and we're done. That's our goal. Canonicalization. So, what's the deal?

Well, in reality, there's some cone-- yeah, it can penetrate like that-- could penetrate the outside cone. This is getting messy. So if I extend this line, there's a cone of half angle here, which is equal to whatever the turn angle is at that vertex, which is, again, specified. We're not free to set it to whatever we want. We know the next edge must lie on this cone. What I do-- now, on the other hand, off to the side, I have in mind a vertical cone whose half angle is alpha over two. So it's quite small. And I know-- how do we do it-- initially, the first edge was along the maximum x direction, and then it spirals up from there. OK.

Here's what I'm going to do. There are sort of two situations. What I'd like to do is put a cone here that is vertical. Something like that. Just like I have here. The trouble is, the right side of this cone does not intersect the boundary of this cone. And need it to in order to form the right angle here. It's going to-- it might go through the middle of the cone like it does here. So then what I do is I take this canonical configuration and I rotate it so that-- this is hard to draw-- it'll be something like this. There's no hope of seeing this.

So it's on the surface of this cone. It's not on the far right edge. It's going to be some intermediate point. And that's exactly where it intersects this cone. It's just like the previous picture, just harder to see. I'm taking the intersection of these two cones. If I just rotate the picture, then it will lie in the intersection-- if they intersect. But they might not intersect. So let me go here and try it again.

So the easy case is when I can draw a vertical cone and it intersects the cone that I need to intersect. The harder case is-- maybe it's more extreme. Maybe it's a very tight angle here. So there's a very small turn angle. I have to intersect this cone. Because the cone that I'm working with is actually smaller-- it's half of this angle-- it might not intersect this cone. In that case, I'm going to rotate the cone to fall over a little bit. So instead of being like that, it's going to be like this. OK.

So I want the intersection of these two cones, which I've conveniently made this edge here. So the first edge will lie along here, and then it's going to spiral out from there. So I still have the canonical configuration. I've just tilted it. Tilting is going to be necessary because I have this angle to match up. The convenient thing about the canonical-- the alpha CCC, canonical configuration-- is that I have this half angle of alpha over two, so I can afford to tilt it by up to alpha over two. It will still stay within the ribosome, which is of half angle alpha.

And all I need to show here-- and once you think about it for a while, it's obvious-- that ideally, I don't tilt it at all. I've got tons of room. Huge amount of room here. But sometimes I'll have to tilt it, but by at most alpha over two. So I will stay inside the ribosome, because this apex was inside the cone and the smaller cone, even if I tilt it all the way to meet this edge, it will stay inside the cone. In fact, it'll stay inside the big cone. In fact, this cone that I tilt-- the one that contains the rest of the canonical configuration-- will always contain the up direction. That's how to see it.

If it always contains the up direction, then at most, it's that big. And that will-- because that's an angle of alpha because that was the half angle of the big cone. So that will be a half angle of alpha over two. As long as you contain the up direction at all times, you will not fall outside the big cone. So the rest is just momento. So you play this movie backwards. As things come in here, this cone is going to wiggle back and forth, depending on how this angle changes. Once the edge gets all the way in, you absorb it into the canonical configuration.

Little bit of work there but you just sort of twist the cone around until that algorithm that we described for producing canonical configuration would actually produce what's inside the cone. Then the next edge comes in, cone wiggles around until the edge gets all the way in, then you canonicalize what's inside the cone, and repeat. So if you had a way to get it out, you could put it back in and keep track of all the stuff that happens on the inside. That's what this theorem says. And then there's just slightly more to say.

If you have a flat state, we want to prove these things are flat state connected. So take some flat state-- I really should have only obtuse angles. I claim this flat state can be produced using a cone of the appropriate angle. If the sharpest turn angle here is alpha, then you need an alpha cone. And the way to think about that is to think of the cone moving instead of as the chain moving. It's a lot easier. So you want the chain to lie around here.

So you start by moving the cone, I guess like this. So that it just barely touches the plane, and this edge spews out. Is that the right way to think about it? I don't know. Should be like this. So you take the cone. This is like putting frosting on a cake. So you move your cone like here. You squirt out some-- this edge. Then, you do it so that when you're here and the new edge is created, it's still in the plane. And then you just sort of move around there. I'm just going to leave it as a sketch like that.

Once you know that you can produce any flat state, of course, you can reorient yourself by relativity so that the chain is moving instead of the cone. So you can produce this thing with a ribosome. Once you know all flat states can be produced and you know all producible configurations can be canonicalized, then you know it's flat state connected and you know all canonical things are flattenable and vice versa, by continuous motions without self intersection. And all of this is algorithmic. It can tell you how to go from one place to another.

And so this is a candidate algorithm, I would say, for how nature folds proteins. Just thinking about the mechanics, not worrying about how it's implemented. Maybe you take this model and then you try to make it physical forces, and you get a way to fold proteins. That, of course, remains a mystery. But next time we will talk about some very simple models that are motivated more closely via biology of how proteins might actually fold, and talk about the complexities that you get there.