[SQUEAKING]

[RUSTLING]

[CLICKING]

ROBERT TOWNSEND: OK, so let's look at where we are in the class. So this is the class calendar. So you can see today is virtually kind of the midpoint of the semester.

Now, the other thing is the reading list. So today, again, contract theory and mechanism design. There's only one starred reading. It's "Rentals with Unobserved Outputs," this chapter 5 of Medville. There is a non-starred reading, and I have material on that in the lecture today. It's this distributed ledger MIT book, this chapter. Although I've learned a little bit more since I wrote the book, which was published yesterday, and I'll share with you a little bit of that in the lecture.

Anyway, this is all optional. I'm not quite sure how many of you have been following Bitcoin and so on. But as I keep saying, every time an economic topic comes up that relates to Bitcoin, as it did with the ledgers last time, then I weave that into the lecture. So this is going to be about today.

The second half of the lecture is about smart contracts.

And then for the study guide today, I can call on a few of you or get volunteers. Can someone define for me the concept of a household balance sheet, and also, the income statement? Just say a few words about what's on those financial statements. Armando, do you want to take a crack at that?

AUDIENCE: Uh, yeah, sure. So the balance sheet is where you keep track of everything that-- so I'm actually not sure which is which, but I know one of them involves that you keep track of your assets and your liabilities. I believe, yeah, that's the balance sheet.

ROBERT TOWNSEND: Yes.

AUDIENCE: The statement of income is like all the money in and as well as all the money that goes out, expenditures.

ROBERT TOWNSEND: That's right. Good. OK. Yeah, the balance sheet has assets on one side, liabilities on the other. They don't necessarily balance. Hopefully, the assets are greater than the liabilities, and the difference is equity. So the liability side of the balance sheet also has an equity component, which is how much wealth you own that you do not owe to other people through debt.

The income statement, you stated it properly, yes. You have revenues and expenses. Now if you take the difference, you'll have profits or net income, and then a couple of other lines on that income statement were how you dispose of your net income in terms of either consuming it or saving it. But your answer was great, thank you.

Then we talked about, in class, life cycle planning. Can someone tell me what it is exactly? What we mean by life cycle planning?

AUDIENCE: Can I take a shot at it?

ROBERT TOWNSEND: Yeah, sure.

AUDIENCE: So it's planning what you're actually, like what choices you're going to make over your entire life. So you're sort of planning ahead for how much you're going to spend, like based on your information about what you expect your income is going to be and stuff like that.

ROBERT TOWNSEND: Yeah, that's good. Yeah. It's like and now to, when you die or when you bequest to your kids or something. And typically, it's year by year. And it's kind of-- I guess I would say the point, I didn't ask this, but the point of life cycle planning is to manage your assets in such a way that your consumption would be as smooth as possible over the years of your life. And sometimes you can't achieve that exactly.

The other thing that people associate with a life cycle planning is an outcome of that process, namely you have a peak where you're accumulating assets early on as a young saver with consumption less than income, and then you start spending down those assets when your income sources diminish, especially if you retire.

OK. And then we applied it to Thailand. I'm not sure exactly what to ask about. Well, maybe if one of you could tell me a special feature of the economic life of Thai villagers that somehow we incorporated into the life cycle planner that you would not typically see in the US.

AUDIENCE: The gift giving?

ROBERT TOWNSEND: Say it again.

AUDIENCE: The gift giving with or without the income.

ROBERT TOWNSEND: Excellent. That's what I was looking for. Yeah. So it looks like an individual operating in isolation, or at least in most borrowing or lending and acquiring financial assets, but in Thai villages, those households interact with one another by giving gifts and so on. So the way it was in the equation was that it smoothed out the downfalls in income because the gifts would be coming in, and likewise smoothed over the peaks because you give those peaks away in terms of gifts to other people. OK.

Can someone tell me what is the concept of a distributed ledger?

AUDIENCE: Is it basically just like a publicly recorded, like where you can have financial transactions be recorded and available to other people to see?

ROBERT TOWNSEND: Yeah. So it's a ledger as in an accounting ledger. It includes a list of transactions. But in addition to that applying to a given individual, it's consolidated so that it includes the transactions in and out as if it were on a common ledger, a ledger that would apply to all the households that are participating.

The twist in that is the word distributed. So although it's common, it's not supposed to be held by one central party. Everyone's supposed to have an identical copy of that ledger.

And in class I complained a little bit about when we tried to create one for the Thai data. It didn't quite work. Does someone remember why?

Dixon, you want to take a stab at it?

AUDIENCE: I'm actually not really sure why it didn't work.

ROBERT TOWNSEND: OK. That's fine. Anyone else?

Well, to be fair it was only on one slide and there was a lot of material. So let me highlight.

We didn't have the distributed ledger running when I collected the data. Instead, we took the answers the households gave us to the questionnaires and created financial accounts out of the answers-- income statement, balance sheet, and statement of cash flow. So we had the financial accounts for all the households, but we didn't try to reconcile them to see if they were consistent with one another.

So in fact, when they buy something for cash, they tell us who it was, potentially in the village. I buy something from J, J should show up with the same transection where J is selling and getting cash in return. So it's one common transection, it should show up on a distributed ledger.

But because the data didn't try to reconcile all the entries as they were reported, it was not unusual for one household to report a transaction with another household that the other household did not in turn report back. The point being that distributed ledgers are an excellent way of reconciling transactions live in real time so that they become more accurate.

OK. And then we went to Thailand and talked about currency, and there was this-- the whole lecture was about solving planning problems. Can someone tell me in words a bit about this Miller-Orr model of cash management?

AUDIENCE: So in the Miller-Orr model, the net expenditures are the sum of smaller recurrent expenditures and infrequent larger ones that have a lower probability. And so the goal of the household would be to minimize the cost. And I remember there was a Bellman equation for that.

ROBERT TOWNSEND: That's good. You remember the two components of the cost?

AUDIENCE: There was the flow opportunity cost and also adjustment cost.

ROBERT TOWNSEND: Yeah, yeah. So the flow opportunity cost is the foregone interest from holding cash. If you have currency in the house, it's not in the bank, it's not earning interest. So that's kind of the flow of opportunity cost. And then every time you go to a bank, either deposit or withdraw the currency, there is a real transaction cost.

OK.

So finally, we get to Kenya and M-Pesa. Can someone describe for me-- let's see, who's online here-- what exactly is M-Pesa and how it works? I'll take a volunteer.

AUDIENCE: I think it's a mobile phone-based money transfer service. And you like, pay a small fee and then you can transfer money to someone else. It requires trust in the service, I guess.

ROBERT TOWNSEND: What are they actually transferring around electronically?

AUDIENCE: Uh, is it not just like, I guess like e-money?

ROBERT TOWNSEND: It's e-money. It's actually--

AUDIENCE: It's a telephone, isn't it?

ROBERT TOWNSEND: It's a cell account. Yes. That's right. It's actually credits. Like you buy credits for phone usage in advance from Safaricom. So it's just like you would pay a bill at the end of the month, they're kind of like paying a bill at the beginning of the month. They're buying cell phone credits, but they don't intend to use it on their cell phone. Not necessarily. Instead, they transfer it around, buy things.

And people cash out, they reverse the transaction. So it's like you can-- it's like mileage that you get on an airplane, but then you can sell your miles. So they can cash out by selling their e-credit back to the company.

So in some respects, it's not so novel. It's just an electronic account, in this case, denominated in terms of credits on your cell phone. But what is novel about it is that it can be transferred around at very low cost and you can cash in and cash out. So it kind of looks like money because you can use it on your phone like money.

Which then you start to think about it, what is money really? It doesn't have to be fiat money, the paper stuff. Like you can have a checking account in a commercial bank in the US and pay bills online out of that account. It's not central bank reserves or fiat money, but it is money in the sense that it's used widely in exchange and has a high velocity.

OK, the trust thing was a good comment because we're going to come to that today in the lecture in terms of, in this case, people trust Safaricom to enter the transactions properly and to not cheat after the fact, not to steal the money, and to allow them to cash it out. But a lot of people are more paranoid about third party providers, depending on the context. So we'll get to how to solve that problem today.

So this lecture comes in two halves. Contract theory and mechanism design is all of it. The first half is going to feature an application in-- back to our favorite stomping ground-- the medieval village economy. And then as I've already anticipated, the second half is going to focus on generalizing the concepts and talking about smart contracts and validation and trust.

So if you looked at this map, in this case, it's not England, it's France, you would see these solid filled in circles, the little Cheerios, and the empty circles, and small dependent settlements. So these stand for villas with only demesne, so to speak, which means everyone worked for the lord and didn't own their own land. There are mixed villages that have demesne, but the households own some of their own land as well. And then villages where the lord-- like the Bishop of Winchester in England doesn't own any of the land at all. And these are just scattered little hamlets.

So you can start to spot some patterns here. This is kind of the chief of state of the lord, the lord of this area; these little Cheerios surrounding it or halfway filled in with some land directly farmed for the lord; and then a little further out you get these open circles where the lord does not directly have a claim on the land, although there may be taxes due. So that's summarized in this first slide under alternative forms of organization.

When it's held in demesne only, then essentially the peasant population are entirely slaves. And others there was no land like that. Instead, there were not labor obligations, but households were obliged to pay rent to the lord a certain amount of grain that was due. And we'll come back to what that contract might have looked like.

And those empty, open circles were really far away from the central monastery. So we got three different kinds of organization here.

And theory that we're about to write down is going to say, well, maybe these forms of organization varied from one village to the next, depending on how close they were to the manor, in particular because there's private information. And if you're very close up, then essentially the lord sees everything. And if you're very far away, it's extremely costly and difficult for the lord to see everything. And so those exchanges have to be incentive, compatible, and induced, concepts we're going to define today. Or you may see nothing except with a cost where there's an audit, like a tax authority going in to actually take a close look, but at some cost.

So here's the model for simplicity. Just imagine there are two agents, only one lord named agent 2 and one agent named agent 1 that's a stand-in for the entire villa population. So agent 1 here is seeing his or her endowment, 1 stands for the household number, epsilon for the fact that the endowments are the yields of crops grown on the land farmed by agent 1 and a random. And furthermore, at least initially, only seen by agent 1, the villa.

We can simplify. This notation was here deliberately to try to remind you a bit of things that we've done earlier in other lectures where epsilon was a shock. We talked about portfolios of land as a function of shocks, and so on. But in fact, this whole object can be just summarized by theta, a value lying in some sum set capital theta, occurring with realizations p of theta.

Agent 2, the lord, we're going to simplify. The endowment is public, there's no private information there. Agent 2 is, again, the central monastery. Both these endowments, the e1 of epsilon, or theta on the one hand and W on the other, can be more than one crop. It can be multiple crops. So K is the dimension of the vector.

So what do these guys do? The two agents agree to a plan, a resource allocation rule, which allows agent 1, the guy with the private information, to be sending messages to agent 2, and as a result of the message, some tax is paid from agent 1 to agent 2. So little m is a particular message, big M here is the set of all possible messages or the message space.

This is quite an abstract concept. It could be something like, it rained today, or, how's the weather over there, or, I lost my plow. This abstract message space. But we're going to give it some meaning.

The meaning happens through this allocation rule. Namely, if theta is the true endowment of agent 1, how much of the tax, f, is a function of what is said, M. And they all agree to this abstract message space and to this allocation rule f. So agent 1 can map what he says into his overall outcome.

So let's suppose the agent 1 knows exactly what he or she wants to do. Namely, when his or her endowment is theta. The best possible message that could be sent in that abstract space is, we'll put a star on, an m star of theta. This is the maximizing message sent by agent 1 to agent 2 when his endowment is actually theta. This is way to write the fact that it's maximizing. Namely, the overall outcome for agent 1 in terms of agent 1's utility is as high as it can possibly be relative to any other possible message that could be sent. There may be ties, but at least m star is maximal. The inequality doesn't go the other way.

If it went the other way, less than or equal to, there would be a message m that did better than what we said is the maximum, and that would be a contradiction. So 83 is just a statement of maximization.

Now, this holds for any other possible message, little m. And in particular, if the reality had been something else, say theta tilde and not theta, m star would still denote the maximizing message, but it would maximize that theta tilde. The role that it plays here is that is a particular message like m, a particular counterfactual message. But because it's a valid message, although it would have been sent in different circumstances, we just can substitute m star of theta tilde up on the right-hand side as a particular m since 83 holds for all possible m, and that gives us this expression 84.

So the way to read this is that when the agent's true endowment is theta, see theta on both sides, and m star theta is maximal, that weakly dominates, saying what that actual dominates what the agent would have said in the counterfactual situation, where though his endowment were theta he could have said m star of theta tilde.

With me so far? So these are just simple statements of maximization and notation.

Now we're going to make a bigger move here, although it seems rather innocuous. We're going to change the message space and we're going to change the allocation rule.

The way we're going to specialize the message space is to require that the agent announce values of theta. They're still not seen by the monastery agent 2, but the message space is my output was high this year, my output was low, et cetera, for as many possible values of endowment as there might be. So we change the message space.

The next thing we do is change to transfer rule. Well, for one thing, we have to define the transfer rule over messages about theta. So we denote that g of theta. But actually, we're not going to, quote unquote, "change the rule." We're going to define g to be the same function f, but f used to operate on abstract messages whereas g operates on messages about the actual endowment.

So the right-hand side here is a composite function. You have theta tilde, you send message m star of theta tilde, f acts on that. So it's a composition of f and m, whereas g operates directly on messages about output.

There is another way to see this, which is where we're going in terms of the result. The agent knows the maximizing action for every possible theta. So he could program the computer to do what he wants it to do in terms of sending a message up to whatever key gets pushed. So effectively, he starts pushing keys that denote the theta, in this case, theta tilde, instead of actually sending a message.

The computer sends the message on behalf of the agent. Of course, the agent has, a 1, has coded it all up. So the computer functions properly on behalf of the agent's best interests.

All right. So now we've changed two things, the message space and the allocation rule. And now we go back to previously derived equation, 84, that said this thing, but start substituting-- m star of theta tilde is composed with f will be g of theta tilde on the left-hand side, f composed with m star of theta will be g of theta just by the definition of g.

So we get this. We get 85. That's just substitution. But it's powerful. OK?

So now we can start calling it, misleadingly, a truth-telling constraint. Now, no one is choking this guy to death. He gets to say whatever he wants. But under this new message space with the g allocation rule, when theta actually happens, he wants to say it did, as opposed to when theta actually happens, lying and saying that it's theta tilde. So this constraint 85, which was actually derived just from a principle of maximization, actually starts looking like a truth-telling constraint, where the space of messages is the space of possible outputs, and now miraculously, the agent's going to tell the monastery the truth about it. The bill is going to tell the monastery the truth about it.

But we're not kind of insisting on truth telling. 85 is derived as a first principle. They're called incentive compatibility constraints.

Anyway, so once this mechanism is in place, whenever theta happens, the agent will say so, affecting g. The g is actually what would have happened under the old mechanism when theta happens and the agent announced m star of theta in the abstract space. So this is now achieving exactly the same set of allocations.

In fact, we could call them shock contingent allocations. If you go back to the very early notation with the shock in there, it looks like the agent is announcing what those shocks are, and under this allocation rule, will be telling the truth about them as long as 85 is part of the system.

So just like before when we had states of the world, we talked about state contingent allocations, like gift giving, the risk sharing rules, and so on, here, again, we're enumerating these states, epsilon in this case, when we talk about state contingent allocations. So the method is exactly the same as before as long as we append onto the system constraints like 85.

Another way to say that is all of the consequences of private information are captured by 85. So we can go and do our thing with the programming problems, maximizing lambda weighted sums of utilities subject to resource constraints, and so on, and just append on these extra so-called truth-telling constraints.

So we don't have to search over abstract message spaces, because what's the limit of that? Anybody can come up with a message space. It's too abstract and not very specific. We restrict the message space, but without loss of generality. And anything we could have done in an abstract way with m and f, we can now do with theta and g.

So here, in fact, would be a programming problem. We're going to maximize the lambda-weighted sums of utilities. A little bit of new notation, I guess, if I forgot to say it earlier.

Theta occurs with probability P of theta. So summing over theta makes this an expected utility representation on the left-hand side. Note that what the agent gives up, minus g, is what the monastery gets, plus g.

There is an expected utility term for the monastery, too. This is a two-party contract. They're going to maximize this thing.

We are going to maximize it for them to determine Pareto-optimal allocations and make a prediction about what they would be doing, subject to this extra constraint-- 87, the so-called truth-telling constraint. You may say, well, where is the resource constraint?

Well, it kind of got substituted in to the objective function because what agent 1 gives up is what agent 2 gets. So there's no reason to talk about consumption of agent 1 and consumption of agent 2, and make sure that it adds up to the total. That's just a handy convenience in this setup.

So more ways to implement it-- the villa is, at the end of the year-- and so far, this is a static problem-- the agent just hands over goods to the monastery. Could be a lot, could be a little-- whatever it is, it's totally accepted. As a solution to this problem, instead of saying, oh, I'm announcing theta and I'm going to give you g of theta, they just hand it over.

It's the same thing. It's just simpler. Now the key is, again, this incentive constraint, which makes sure that agent 1 has the incentive to hand over the right amount, the right amount being the solution to this programming problem.

Second consequence-- suppose there's only one good, wheat. When you look at 87, it's going to simplify to something trivial, namely, no trade. Why is that?

This says utility on the left weakly exceeds utility on the right. And g of theta on the left must be less than or equal to g of theta tilde on the right because it's the same function u. And theta is the same argument otherwise.

But you can reverse the logic, and put theta tilde for the actual situation and theta for the counterfactual situation. You'll get another inequality because this holds for all theta and theta tilde. And then you're going to have the consequence that g of theta tilde on the left must be less than or equal to g of theta on the right.

Now we've got two inequalities-- less than or equal to, greater than or equal to. It must be equal to. So g is the same regardless of theta.

So at most, the villa is paying over to the monastery a constant tax. But then it's not state contingent, and it could be 0 if we wanted to make them both better off. So that degeneration does not necessarily happen if there's more than one good.

And it could be another good is like labor, not just wheat. And the specification is, you've got to go, here's the transfer of wheat. And otherwise, I'm going to go work for you on the land in my village, or trundle off to the monastery to work there.

So there could be a wheat/labor trade-off, as an example of more than one good. And then this is not trivial. And some trade is possible.

Another special case-- if agent 1 is risk averse in a certain way, then we may be able to get some trade even if there's only one good. So this is just going to exploit the possibility that if agent 1 has a strictly concave utility function, then the degree of it will determine his or her adversity to lotteries. So for example, you might say I'm high today. I would pay a big tax, but I'm going to pretend that I'm low so that I pay less of a tax.

But that claim of having a low value may make the tax very random. And if he's really high, and that being a high value makes the guy very risk averse for odd reasons, then he may not want to claim low because he prefers to avoid the randomness associated with the lotteries. So lotteries in general are going to help sustain trade because they exploit differentiable risk aversion.

So with that said, we can now write this program up here in lotteries. Let's call pi of tau given theta the probability of a transfer from agent 1 to agent 2 of magnitude tau, conditioned on the agent saying theta. And then this program 6 over here becomes 7, where we're still maximizing a weighted sum of utilities subject to an implicit resource constraint. What one guy gives up, the other guy gets, in the tau, but also this now-revised incentive constraint.

And let me read it for you-- when theta comes up, that's the actual yield. Agent 1 says, Yep, that's my theta. That's the message, tells the truth about it.

That implements a lottery with probabilities pi of tau so that summing over all the tau gives the expected utility o the agent for having theta and telling the truth about it. And on the right-hand side is the counterfactual that although theta came up, he lies, and says, nope, I'm theta tilde. But that comes with this potential randomness, and the expected utility on the right-hand side is not greater than it is on the left.

So that's the same kind of truth telling incentive compatibility constraint as before, with the simple addition that we have those lotteries in there. You will recognize, because we've done this before, that with the lotteries, things turn into linear programs. So when you look at this, the control variables are these pi of taus, and in the objective function, they're multiplied by these leading coefficients.

And in the constraint set, the control variables are, again, the pi of taus weighted by certain coefficients. So I won't repeat what we talked about that day when we introduced this concept. I'm really just trying to remind you that we've already seen things like this, except that here, we have added the addition of these so-called truth-telling or incentive compatibility constraints, also written down in lotteries. And because it's a linear programming problem, we can compute solutions to it, which you've been doing on some of your homework problems, not with lotteries per se, but computing.

Now, there is another twist here in terms of incentives. If it's true, say, that high-theta guys might want to claim to be low because they would pay less of a tax, I should say, where is that intuition coming from? If agent 1 is risk averse and agent 2 is risk neutral, then agent 2 should bear all the risk, in which case when agent 1 has a high value of theta, he would transfer stuff to 2. And likewise, if agent 1 has a low value of theta, he would receive stuff from 2, and consumption would be constant if it weren't for these incentive constraints.

But agent 2 doesn't have to be risk neutral. They could both be strictly concave people. We still know that apart from 94, there is this monotonicity condition, that consumptions should co-move together.

So when theta is high, agent 1's consumption could go up. But so also should agent 2, meaning that some of that high value is being transferred to agent 2. So it's the same whether or not 2 is risk neutral or not.

Now, that intuition when the endowment is high, agent 1 could claim that it is low in order to avoid paying this transfer. But there is a way to preventing him from doing that. It's like-- I don't know, I'm not a gambler myself-- like when they call you, they put it on the table, and say, let me see your hand.

So they say, OK, you're claiming you're low. Actually, let me apologize-- this goes the other way. You only need to worry about high guys claiming to be low.

Low guys cannot claim to be high. Why? Because if they claim to be high without necessarily taking it away, they have to display their wealth. And then it would be clear, if they can't do it, that they were lying.

So we kind of have a one-way incentive constraint going on here. That's not terribly important, but the show and tell is a device that can be used. And we'll come back to it later.

All right, so now let's get into multiple periods, be a little bit more realistic about this, just slightly-- two periods rather than one. Same actors-- agent 1 having these random endowments in both periods, agent 2 having a potential time-varying endowment W. And then these probabilities of shocks, theta 2 to agent 1 in the second period, are basically potentially non-independent they could be determined by the shock that agent 1 incurs at date 1. So if you're into statistics, you could call this a Markov process and be fancy about it, but all you really need to know is the state-dependent notation.

OK, so I want to know what's an optimal arrangement in this world. So we write down the maximization problem, which is the lambda-weighted sum of expected utilities, obviously taking into account both periods subject to resource constraints and to incentive compatibility constraints. So there's a lot of lines here, unfortunately, but it's not so bad.

This is lambda 1 weight for agent 1, lambda 2 weight for agent 2. Otherwise, this is the first period expected utility when the agent is theta 1 and says so, affecting the transfer tau. This new term here is beta, discount rate, times the expected utility in period 2.

Period 2, agent 1 has theta 2-- actually says so. And then you integrate up over all possible thetas and discount it by beta. One little subtlety I'm about to feature is the transfers at day 2 can depend not only on what is said about theta 2, but what was said about theta 1-- the history, as it were-- in date 1.

Date 1 is the first day. Nothing's happened. There's no history, but there is a history to potentially use at day 2. So this is the objective function.

And again, the resource constraints are loaded in here because what agent 1 gives up, agent 2 is getting. So we don't need to worry about that, for both dates and whatever the outcome of the lottery. Now, let's think about the incentive constraints, working backwards, as in dynamic programming, from the last period.

So say some theta 1 were already announced, and that's history. They arrive at day 2. Agent 2 has a choice about saying the actual value of theta, theta sub 2, telling the truth, or lying about it, and saying theta 2 tilde.

So what does that incentive constraint look like? We're already in the last date, so there's one period left. It's period 2. And this is got theta 2, say so, versus got theta 2 and lie about it, saying theta tilde.

This history, theta 1, is the past in the archives. So it's in both sides because that's already public record-- not what theta 1 was per se, but that it was announced, theta tilde.

Now the incentive constrain at day 1, working backwards from day 2, looks a little bit more complicated. But it's just going to use what we already know from 96-- that they will tell the truth in the second date, regardless of whatever history was announced in the first date. So these branches here, with the beta in front of it on both sides, are here, have theta 2, say theta 2 truthfully, versus have theta 2, and say theta 2 truthfully.

So on both the left and the right-hand side, the second period is not in doubt. The agent is going to be telling the truth because of 96. The real action has to do with the first term of the first date, namely have theta 1 and say so versus have theta 1 and lie about it, and say theta 1 tilde. And this inequality constraint means that a weakly dominant strategy is to tell the truth-- have theta 1, say so.

Now note, again, the interesting way in which this history evolves. If they say theta 1, it will be part of the history of day 2. They could say-- they won't-- but they could have said theta 1 tilde and lied about it. And that would also be recorded on the books as history.

So the thing that's varying at the second date is this announcement in the first date, as we consider logically what the agent will want to say in the first date. So this is another incentive constraint. And again, there's more than one of them.

This is for every possible actual theta 1. In any potential counterfactual announcement theta 1 tilde, constraint like this holds. So I should have caught myself earlier on this, because I talked about the incentive constraint or the truth-telling constraint as if there were only one. But in practice, there are many because we have many possible realizations of theta. So what we've done today so far is called the "revelation principle," meaning that those truth-telling constraints can be imposed without any loss of generality in the search for private information-constrained optimum.

Now, there's an issue of language here. Instead of talking about a Pareto-optimal allocation, we should be talking about an information-constrained optimal allocation. Some people-- I don't love it-- talk about Pareto optimality as first best, and this notion of constrained optimality as second best, with the good intention of distinguishing those two cases.

It's second best but unavoidable, because you cannot get around these kinds of incentive constraints. They may or may not be binding, but you definitely should be writing them down. And when they're binding, they constrain the solution, and hence the terminology "constrained optimum." Question so far?

So what do we know about the solution? One thing I've already hinted at-- if there were full information, then if we go back, this is a version of what you've seen before. It's written out as two periods. But remember, the way we solve these things in Debreu decision tree, we said for any date and for any history of states leading up to that, we'll have a risk-sharing rule, a transfer rule, which is a function only of the current aggregate endowment and nothing else.

So the full risk-sharing problem without the truth-telling constraints will devolve into a series of static problems, which means the transfer rule at day 1 would be the same as the transfer rule at day 2. But in fact, when we have this information-constrained efficient arrangement, it's almost surely the case that we're not going to be able to get to the full risk sharing. And likewise, if you're already at the second date, you're kind of doomed, especially with one good, no lotteries. You can't have a theta 2-contingent transfer at the second date.

But possibilities remain at the first date. Why? Precisely because you can tie the allocation rule in this lottery at the second date to what is announced at the first date.

So this history of messages starts to matter. Or another way to put that, when it's necessary to have that, what happens at the second date is no longer a static problem. You have to utilize what also happened at the first date, which you could call a tie-in of sorts-- intertemporal cross-period ties in of the second date to the first date.

It also reminded me of what the historians are talking about. They said, oh, these manners are terribly inefficient, bound by tradition. They can't break away from the past.

Well, the theory we're going over says that there's a very good reason why they would like to be bound by the past-- for incentive purposes. So one example of that is borrowing and lending. I mean, suppose the villa has a low output, and they say, look, I promise-- let me pay less or borrow, and I'll repay the loan next period. And there's a commitment device to make sure that the loan will be repaid.

So would the villa want to claim to be a lender when the income is low? No, they would want to be a borrower. And conversely, when the income is high, they're going to want to be a lender, as long as they're guaranteed of getting stuff back.

So what's going on here? You're tying what happens at the second period to what happens at the first period-- those so-called intertemporal connections. It's not like you say a theta, and then you pay high or low today.

You say a theta, you may pay high or low today, but there are consequences for tomorrow. So you choose what you're saying to give you the maximum possible utility. And you have an incentive to smooth consumption.

So when you're low, you want to say so. Why? Because that's better for you in terms of getting money today, where you value it more than tomorrow, where you're likely to be higher.

That was a bit of a digression. It also comes up at the bottom of the slide. So the borrowing and lending scheme is an example of the past mattering in a very trivial way with the borrowing and lending. What you've got to pay today is a function of whether you were a borrower or lender last time.

It's not time consistent. By the time you roll around to day 2, and ask these guys whether they would like to start over, they're, like, yeah, sure, we can both be better off. The incentive constraint is imposed to get them to behave properly, as best they can, at date 1.

Once date 1 has happened and we move to date 2, there is no longer a need for that constraint. And if we were to resolve the problem at day 2, they would likely do something different. So it's called "time inconsistent." The arrangement is not going to be consistently carried out over time, voluntarily. They're going to be bound by these rules.

So then the thing at the bottom of the slide, claim 1-- borrowing and lending is not optimal. So let's go back to this. Say the solution to this program 8 is the borrowing and lending solution. When theta is low, one borrows. When theta 1 is high, the agent lends, and re-pays back, with the payment going in the reverse direction in the second period.

I claim that's not optimal. It's probably still mysterious. When the agent is low, he strictly prefers to say he's low rather than saying that he's high. That makes this inequality constraint strict.

Likewise, when the agent is high, he strictly prefers to say he's high rather than saying he's low. That makes this inequality strict. Well, if both inequalities are strict, neither one can be binding.

We might as well solve the programming problem as if they weren't there in the first place. Because now we know they're not binding. And what happens if you solve this problem without the constraints?

That's the full risk-sharing problem. And it can't be optimal because they'll lie their head off about their income because there's no consequences. So that's the way to think through the bottom of this slide.

Full insurance isn't feasible, not implementable because it's going to violate the incentive constraints. And borrowing and lending is not optimal, either. Which means, well, what is optimal?

What is the actual solution to that program? Well, actually, it's a hybrid scheme that is a bit like risk-sharing, and borrowing, and lending. When you're low, you say you're low.

You borrow, but you actually get even more money, but less money than you would have if it had been full risk sharing, and vice versa for when you're high. So it's a hybrid contract that is mixing up the two aspects, which makes sense because neither extreme is a good thing. So you end up in the middle.

So this slide is about the other option I alluded to at the beginning. Let's revert back to a static problem. There's a cost K, use of resources, which the agent 2, the monastery, can incur in order to go and take a look at agent 1's actual outputs.

So instead of claims about it, which will also happen, there's kind of like this possibility of an audit. It's as if in the US, you file your income tax returns making claims about stuff, but with a certain probability the IRS will audit you, and find out the truth of the matter. And if you're caught lying, you go to jail. That's how they got Al Capone, actually-- not for all the crimes, but because he misreported his income.

So K is this audit cost. And we can introduce another variable called d, where d is the dummy. d equal 1 for audit or d equals 0 for not audit.

And then we could enhance our transfers or the lotteries of transfers to depend on whether or not there's an audit-- equal 1 for an audit, equals 0 for not. When there's no audit, they have to rely on the filing of the return or the message that was sent. When there is an audit, they get to compare the message that was sent with the actual theta.

And as you could imagine, going-- you could kill the guy, going to jail or whatever. So I'll spare you this slide except that the incentive constraint will now take advantage of the fact that lying is a pretty treacherous thing. Because with a certain positive probability, you'll get caught in the lie, and you can make the guy's utility really low.

Later after class, you can look at this inequality constraint. It's this line here that you can make a very low number. Because you would penalize the guy if he said he was theta tilde, and he was actually revealed to be theta.

And this is on the right-hand side of the incentive constraint. So the lower you can make this number, the easier it is to satisfy the incentive constraint, and the more you're going to move back toward the full information solution. But it comes at a cost, because this K is a cost, a real cost, that has to get subtracted off from resources.

So when that cost K is very, very high, you will not be auditing very much. The audit will be random. And only rarely will you actually carry it out in order to conserve on resources.

So at the beginning, I mentioned the institutional configuration. Is it true that those outlying villas doing their own farming were audited or audited less rarely than the ones close to the monastery? And this theory we just went through allows one to think about those as maybe what happened in practice, consistent with the institutional data.

So I want to turn to the second part of the lecture, which has to do with mechanism design that we just did, but splice that into smart contracts. The shorthand for this is, think about a contract as a smart contract implemented with today's modern technology. We just talked about mechanism design as delivering contracts between a villa and a monastery as an example.

And along the way, we'll talk about encryption, and hashes, and database management, and these protocols. Encryption-- we've talked about distributed ledgers. Encryption's like the third part of it.

We talked about ledgers as transferring value, as in m-Pesa or other situations. We talked about ledgers as contracts to reconcile trades. This third thing is keeping secrets through encryption, which is logical to have here because we just got done talking about private information.

Is it new? No, not at all. Encryption is ancient, literally thousands of years old. If you go back to Mesopotamia, they would be shipping goods around. They had created little tokens which they put in clay sealed envelopes.

If the seal was broken, you'd know there was some tampering. And otherwise, when the goods arrived, the receiver of the goods can reconcile the goods in the shipment with this invoice, after breaking the seal. So those are clay in Mesopotamia, and that's where Mesopotamian writing came from, actually, is from these tokens.

Or in medieval England, they had what they called "tally sticks," which were just wood, willow sticks. The borrower and the lender would make an agreement. The borrower would promise to repay.

They would write these things on wooden ledgers and split it in half. And so the investor lender would have half of the stick and carry it around, could present it to the borrower for payment, but could also sell it to third parties. And so those willow sticks circulated around like money as a means of payment.

The point of the willow-- it had unique grain, so you could match the stalk with the check. And if it didn't match, you know that the person presenting it is trying to commit some fraud. So it's another early example of encryption.

Encryption today, intuitively-- this isn't quite right-- can be thought of factoring. So as you may know if you're MIT students trained in Computer Science, and Algebra, and so on, when you multiply two prime numbers together, you get another number-- 3 times 7, then 21, and you know how to factor it. But when those primes get big and are multiplied, it's virtually impossible to know what are the two numbers that were being multiplied together.

So the output looks like an encrypted message. You can try to solve it, but you have to try all possible combinations. And it gets to be really hard to do that.

Another way you can use factors, though, is to give someone part of the solution. Like here's the product encrypted, multiplied together, but I actually know the answer. I'll give you one of the factors. You can easily figure out the other factors.

And so that's like hashes. The hash itself means nothing, but it is evidence of something if someone comes along and gives you one of both factors. Because then you can rehash the message and verify that the underlying message was the one that generated the hash.

And finally, we come to cryptographic puzzles, which means the factors are big, but not too big. So then, you could imagine spinning the wheels of the combination of a safe to try to figure out which two factors generated the number. It's solvable, but it takes time.

That's what Bitcoin uses-- these cryptographic puzzles, among other things. So the three objects, three types of things, are encryption, which is done now with private and public keys. And I'm happy to tell you more about that. I even have a primer that I wrote with Nicholas, a CS student here at MIT. Took me quite a while as an economist to understand this stuff.

You probably heard, though, about public and private keys. The public key is used for encryption. The private key is used to decrypt the message.

Hashing I've mentioned already, like multiplying factors together. Cryptographic puzzles, I guess I mentioned again-- the thing is to solve the puzzle, you don't spin the wheels of the safe. You actually run a computer. And that's where Bitcoin is chewing up all the electricity, as all these many miners are trying to find solutions to find the actual factors.

So Bitcoin uses all three. It's actually pretty amazing and pretty complicated. It's using signature schemes, cryptographic puzzles, and something else I haven't talked about-- hashes with Merkle trees. But the entire history of the transactions on the ledger is hashed together with an identifier, allowing what's called Merkle trees as an economic way of storing all that data.

Bitcoin uses proof of work. As I said, you need the equipment and the electricity to try to solve the cryptographic puzzle. The premise here is that it's kind of random who solves the puzzle. And the premise is most of the miners are honest.

We'll come back to that momentarily. So since the larger number are honest, and the person that solves the puzzle is chosen at random, the guy solving the puzzle is likely to be honest. Hence, they validate the message.

Another two objects here-- homomorphic encryption allows computation on top of cipher text for individual agents. And multiparty computation is a version of that that applies to multiple agents. It's easier to show you a little bit of the notation.

So m is the message. Oh, yeah, M like message, like we just did. But the message is supposed to be private.

On the right-hand side, you have this allocation rule or function, f, that seems to require the actual message, which means the message, if it's going to a third party, is revealed, whether it's Safaricom or something else. But on the left-hand side, there's another way to do this, which is as amazing as it is simple. You take the message m, and you encrypt it.

Then you take the function f, which doesn't know what it's doing-- it's operating on the encrypted message. And if you decrypt the whole thing at the outcome, at the end of the day, you get the right answer. You get the stuff on the right-hand side.

The encrypted space looks like junk. It's just all a bunch of bits, zeros, and ones, with hashes, and dollar signs, or whatever you want there. No one could possibly read it. That's what state of the art encryption is all about.

So you take one thing and do something else. You can add them together. You can do rank ordering. It's like junk compared to junk.

But it turns out that all those operations have meaning when done on the encrypted space. So homomorphic means isomorphic. Those underlying true message spaces and the encrypted spaces are equivalent to each other. It's on to one-to-one, and so on.

Multi-party computation is very similar, except there are multiple agents. So this means that what you want to do is take inputs from the different agents, like their wealth. They want to see who is the richest guy. And they apply a function f, which is like a rank ordering, and get an outcome.

But these guys don't want to reveal their wealth. They just want to know who is the richest guy. So another way to do this is take the afferent inputs from the agents, and encrypt it. Now nobody knows anything about what the other people are saying.

You apply f to those encrypted messages-- fine-- in the encrypted space. No harm done. And then you decipher the whole thing, allowing, say, summaries like a rank ordering, without them ever knowing what the amount of wealth actually was, just who is the wealthiest guy. So that's called multiparty computation.

Now, what's going on with these algorithms? We already talked about proof of work. Proof of work is Bitcoin. They're using electricity for the validation among the miners.

Proof of stake is another one, where the validation happens, but instead of chewing up electricity, they get to vote. And their vote counts more if they hold more coins.

Byzantine fault tolerance-- another algorithm-- this, one especially the practical one, is what I was alluding to before. If there are f failing nodes that are either faulty computers, or malicious actors like hackers, you just need 3 times f plus 1 replicas in order to be sure that you've got a safe algorithm. Because you can compare across enough of the entities who are safe to be able to figure out who was faulty.

Federated Byzantine agents decide not to rely on a third party to do the validation. They don't trust each other. But they can name the people they trust the most, and you have overlapping circles of trust. And Stellar is a version of that FBA protocol.

Now, what else about protocols? Nakamoto was very clever in giving Bitcoin miners an incentive to mine properly and not conceal outcomes, to be running the machines, and so on. But it turns out it is subject to some problems like collusion. So the economists have jumped in and have become very critical, although evidently, collusion has never actually happened, even though the number of miners is very small.

Steve Morris, my colleague, and Hyun Song Shin, now at the BIS, looked at a classic problem in computer science called the Byzantine Generals problem. And it has the property that it's not incentive compatible. Even though generals have an incentive to truthfully announce messages and send them to each other, it turns out if you put it in the context of what we just did-- a multi-agent mechanism design problem-- they will not follow the prescribed protocol.

It's not in their self interest, when they start doing Bayesian analysis, trying to figure out what the other general knows. So you can just impose the algorithm and hope people follow it. But from what we did today, that's a bit treacherous.

You really want to look at the incentive of the validators to follow the protocol. And that's where we come to mechanism design. In fact, with mechanism design, we don't even need to worry about agents telling the truth. We don't need to monitor or validate their messages.

The revelation principle tells us that agents will tell the truth if we set up the contract properly. And not only that, we can record past messages the way we did in that multi-agent problem. Past messages just get put on the ledger. Because of the signature schemes, we can be sure that they are valid, they haven't been tampered with over time, and so on.

So where we're going is to implement the mechanism design problem as a smart contract with the code, meaning like dynamic programming, you just map states into outcomes with contingencies. So you've already seen this in the dynamic programs that we've solved. And you can separate out that contracting part from the ledgers.

We just have people agree to the code, experiment with it, make sure it works, set up your encryption scheme. And then it's like having an auction without an auctioneer. We're implementing that hybrid borrowing/lending scheme without having a third party. It's just done by the parties themselves.

And encryption, MPC and homomorphic encryption, make sure that all the messages are private, and nobody gets to see those values. And then at the end of the day, you can kick in the borrowing and lending scheme so that for example, if it's a loan, it's going to be repaid because the loan is backed by collateral. So that would involve a transfer on the ledgers.

So I'm not against the distributed ledgers. As I said, I just wrote a whole book about it. But I just want to point out what we did today, with the privacy and the messages and all of that, can be done with encryption, and allows smart contracts to be executed on the ledgers.

All right. Well, I went over by a minute, and I was rushing a bit at the end. So take a look at these slides, especially toward the end. And see if you can map this encryption back into the earlier part of the lecture. That's really the main point. And I will let you go.