PROFESSOR: All right, well I'd like to thank you for inviting me again to talk to the poker class. It's always great to come here, and we're going to be having a tournament in a couple weeks, so good luck for the people participating in that. Actually, I'm coming back in another two weeks because I think [INAUDIBLE] a Harvard MIT math tournament for high school kids.

I really love visiting MIT. I just wish it were at some other time besides the winter. Then it would be perfect. All right, today I'm going to talk about the University of Alberta's Cepheus computer program. It supposedly solved poker. We're going to talk about what they actually did.

[LAUGHTER]

There seems to be a lot of buzz about this, so I thought this was a good [INAUDIBLE] to do. So I have to tell you that Jared and I did not work directly with the University of Alberta people, but we are very familiar with their methods and have actually tried some of their coding techniques. So we're pretty familiar with the same research that's going on. To It's sort of an, I think, objective commentary.

So by the way, as the lecture goes on, you can interrupt with questions. Just raise your hands if something is unclear because I've been told I have about 80 minutes. Probably spend 55 and then save the rest for questions.

All right, so that line of talk-- first I'm going to talk about what the Cepheus accomplished, what the University of Alberta people accomplished, and I'm going to bring that up by discussing game theory optimal energies in poker. How many of you know what game [INAUDIBLE] is. I just want to know [INAUDIBLE] or what a [INAUDIBLE] is. Raise your hands. OK. So about 1/2, 2/3. Good.

I'm going to do a quick introduction to what game theory [INAUDIBLE] is. We're going to talk about a simple poker game and solutions to it. And then I'm going to go into their algorithm, which is written [INAUDIBLE]. They used the method of counterfactual [INAUDIBLE]. Actually, the method they used to push through to the solution of the problem is counter CF plus, which is basically the original [INAUDIBLE] some shortcuts, which we'll discuss.

After this, though, we're going to think about extensions of computer solutions to other games, including [INAUDIBLE] games and multiplayer games. A couple people have questions about [INAUDIBLE] no limit program. We'll talk about what they're work entailed if questions lead in that direction.

All right, let's talk about what Cepheus accomplished. It's a game theory [INAUDIBLE] solution to heads up limit hold 'em. And so what does that mean? You guys all know what limit hold 'em is, right? Good.

Basically, after [INAUDIBLE] few years, they've achieved and exploited less than 1/1000 of a big blind. So the first thing is not a boo perfect optimal solution. You can still exploit it for about 1/1000 of a blind for a hand.

However, there are probably better games. This is like 1/20-- this is 1/2000 of a big bet. You can actually play heads up for 50 years at normal speed and still have some probability of losing. The reason for that is the standard deviation of heads up limit hold 'em is about five big blinds. So you can just imagine how many hands you have to play [INAUDIBLE] the significance. About, oh, 25 million.

So it's definitely a milestone. This is the first time a real poker game has been solved. In math of poker, we solved ace, king, queen, [INAUDIBLE] on paper, but [INAUDIBLE] a real poker games that's solved. However, given their previous work, it was just a matter of [INAUDIBLE]. I remember two or three years ago they passed the 1/100 of a big bet, which is sort of our measurement of significance. If you're playing and you're winning more than 1/100 of a big bet for a hand, you can [INAUDIBLE] it's a probable game. Below that comes theoretical. So it's definitely a milestone.

And basically I knew that, if they just maybe spent more CPU power, they would get the solution. For 900 CPU years, we finally got the solution. So I don't know. If I had that much CPU power, I'd solve a few problems, too. But it's still the miles [INAUDIBLE]. It's great.

So what effect does this have on other games? Does this mean poker is going to go the way of chess for computers who are just much better than we are? I don't think we're there yet, and we'll talk about that later.

So let's talk about Nash equilibrium. So John F. Nash won the Nobel Prize in 1994 "for pioneering analysis of equilibrium in the theory of non-cooperative games." And he extended the work of John Von Neumann and Oskar Morgenstern, [INAUDIBLE] actually first considered these two player zero sum games. So Nash equilibrium is just a set of strategies such that no player can actually improve their strategy and make more [INAUDIBLE]. [INAUDIBLE] whatever.

In a the two player zero sum games, we refer to Nash equilibria as also very optimal. The reason is because Nash equilibria are also the min/max solution. It's the best you can do given that he can see what you do and respond. Simplest case of Nash equilibria is, if you're playing rock, paper, scissors, what's the Nash equilibrium? 1/3 each.

So that's not that exciting in this case, because both players kind of just t0. You can't make more than 0, you can't make less than 0. So it doesn't seem to be that exciting a solution, but in poker it's kind of exciting because they're kind of dominated mistakes people play, or mistakes that actually lose money to the optimal solution. So the reason 1/3, 1/3 is the Nash equilibrium because nobody can do anything to improve their lot. It may not be the best thing to play. If a guy is playing 1/2 scissors and 1/2 rock, what should you play? 100% rock. Yeah, sort of like the Aerosmith strategy.

[LAUGHTER]

Right. So there are much better ways to play if your opponents deviate from Nash equilibrium. So actually game theory optimal is not necessarily the best way to play, even heads up. It's a way to play to kind of guaranteed you never lose. So that's sort of the accomplishment. That's why we like to find these things. I know I could just play this, and I'm not taking total advantage of my opponent's mistakes, but at least I'm playing in away where he can't take advantage of me at all.

Let's do a simple example. So this is an example that I shared with the class a couple years ago. So there are two players, Rose and Colin, and the reason the players are called Rose and Colin are because this refers to [INAUDIBLE] games. One player chooses a row, the other player chooses a column. That's their payoff.

And for a three player game, we introduce Larry, because there are layers. So the two players are Rose and Colin. So each player antes $50 for $100 in a pot. Rose looks at a card [INAUDIBLE] full deck, who will win in the pot a showdown if the card is. Otherwise she will lose. So Rose can decide to bet $100 or check after she looks at her card. So there's $100 in the pot. She looks at her card. She [INAUDIBLE] whether to be $100 or to check.

If Rose bets, Colin may decide to call $100 or fold. If Colin folds, Rose wins. Well, you guys know how poker works. If Colin calls, there's a showdown, and her card is actually a spade. She wins the whole pot. Colin wins the pot.

So what's the optimal strategies for Rose and Colin? Does anybody know the answer? Well, let's do one [INAUDIBLE] part of it. How often do you think [INAUDIBLE] should call? Colin wants a call [INAUDIBLE] enough to make Rose's bluffs probable. If Rose gets a spade, what is she going to do? Bet. She has nothing to lose by betting, unless she's being very, very tricky, but it is correct to bet.

So let's see. If Rose doesn't pick up a spade and bluffs, how often does that have to succeed for it to be profitable? There's $100 in the pot. She looks. If it's not a spade, she has to bet $100, and how much is she risking? How much is she going to win? It's actually $100 and another $100, right? Because there's $100 in a pot. Sure, she anted something and made the pot, but she's spending $100. And if Colin calls, she's going to lose the $100. If Colin folds, she's going to win the $100 in the pot, or she could have just given up.

So it's 1 to 1. So Rose should call half the time-- I mean Colin should call half the time. Rose should bet to bluff in a 2 to 1 ratio, because that's the odds Colin gets to call. So Rose should always bet a spade. If Colin calls 100% of the time, Rose will just never bluff. If Colin never calls, Rose would just be every time. So there is kind of an equilibrium there.

If Colin calls half the time, Rose will be indifferent to bluffing. She'll be negative $50 either way without a spade, and then $100 with a spade. Now, this is strategy for [INAUDIBLE] and the correct strategy for Rose is this ratio of bluff to spade, which is 1 to 2. So Rose should basically bet half of her hearts. She can bet the high hearts, and I guess with the eight of hearts she can decide whether-- is it the eight or the seven? No, it's the-- yeah, it's the eight. [INAUDIBLE] with the eight of hearts she can decide whether to bet or not like half the time. So these are Nash equilibrium and game theory optimal strategies, and basically the value of the game is negative-- is worth $12.50 to Colin. Any questions about this?

All right, so these are the strategies that the algorithm tries to find. Let's go on to the algorithm now. Well, let's talk about what [INAUDIBLE] optimal is first. By the way, there will be about five or so transparencies [INAUDIBLE] of math equations. So just suffer through these. Those of you who understand are going to enjoy the later part, but let's just talk formally about what game theory optimal means.

So there's this game function, u. It takes two strategies, an x strategy and a y strategy, and it gives [INAUDIBLE]. If this was rock, paper, scissors, you would have u of rock versus scissors to be 1, so on and so forth. It's positive for x and negative-- x is trying to-- x gets u, and y loses u. That's the idea.

So one of things is we can take convex linear combinations of strategies. That is, if x sigma xk are strategies and we have some coefficients that are all non-negative and that sum to 1, we can make a new strategy as a linear combination of these strategies. And also u is bi-linear means that the value of the game here is just the linear combination that [INAUDIBLE] sigma x. And it would be the same also for sigma y.

This just means, suppose you have two strategies and you play 1/3 sigma x1 and 2/3 sigma x2, your payoff is going to be 1/3 of the payoff of sigma x1 and 1/3 payoff of sigma x2. Hopefully that's pretty clear. Now we define a pair of strategies to be an epsilonic rim if the best x can do against y is this strategy. The best y can do against x is this strategy-- is epsilon. And if epsilon equals 0, these are in Nash equilibrium. So after 900 PU hours, what they found were two strategies-- sigma x star, sigma y star-- that were within 1/1000 of a big blind of equilibrium. And that's basically [INAUDIBLE] accomplished.

So I'm going to actually go through the nitty gritty of how they did this in case you would like to write on poker solver Sunday. So the big idea that they borrowed was this idea of regret minimization, which is actually pretty cool. Suppose that each time step t the player has a few pure strategies. We're assuming the player has a handful of strategies. In poker, obviously, there's trillions of strategies, but-- two to the trillions of strategies. But say he has two strategies. He can play one or two. Suppose it's odds, or evens, or something like that. Or he has three strategies like [INAUDIBLE].

So basically he chooses some sort of mixture of strategies at the beginning, and we're only dealing with one player at this time. We're assuming the other guy-- we're assuming he's playing against some adversary that's all knowing. That's the original set up, regret memorization. We'll talk about how this applies to game theory in general.

Now with each time t we're given values ut of sigma k. So basically after he determines this, the adversary decides what the value of use of t is, and basically his payoff is just [INAUDIBLE] a linear combination of the things he picked. But the idea is that the adversary can be adversarial. he can decide to make the [INAUDIBLE] strategy score well some of the time, and the [INAUDIBLE] strategy score badly some of the time.

So basically now the idea is to calculate a regret. By the way, this is not the notation that's used in the three or four papers they wrote on this, because I think they did great work-- it's really written as a math paper. It looks like a particle physics paper, which is-- actually for particle physics you need all the complex notation because they're trying to describe something [INAUDIBLE] difficult. I think for computer science papers usually don't need this. So I'll explain this, and then you guys through reread their paper. I think that [INAUDIBLE] give you a quicker way to understand their paper.

So there's this thing called regret of the k option at time t, which is just the sum of the difference of playing k versus playing whatever you played. So basically you can have positive regret or negative regrets. Negative regrets means that what you played-- what you decided to play up to time t was better than just playing k at each time step. So we're only concerned-- we're mostly concerned with the positive regret, which means, instead of playing, you should have made-- you could have made more money by playing option k.

So what's the significance of this? So the idea is we want the average regret, which is this element divided by t. So basically you want the average regret, average amount that you're kind missing out on to be less than epsilon sub t, where in epsilon sub t is the [INAUDIBLE] converging to 0. If you have this, you have some regret [INAUDIBLE].

So the cool thing about this is you can do regret matching. You can let these weights-- first of all, you just look at the positive, the things with positive regret, and weight the options. At each [INAUDIBLE], we basically weight the options that have positive regrets accordingly. And if you're so lucky that nothing is positive regret, you just randomly pick a strategy.

Let's do an example, because I think this is kind of unclear what it is. So let's just say we have two strategies. The player can pick one, or the player can pick two at each time, or the player can pick some mixture one and two. After a player does that, the adversary comes out and says, well, one of them is worth [INAUDIBLE] and one of them is worth 1.

So let's just see how this works. So suppose at the first time step we picked sigma 2 because we don't have any regrets yet. We're just randomly picking a strategy-- [INAUDIBLE] sorry, sigma 1. We'll just randomly pick sigma 1. So the adversary now gives us the value of sigma 1 0 and sigma 2 is 1. And you go, oh, well that means that the regret of the first option is 0 and the regret of the second option is 1. We're aware this first option is 0 is because we already played sigma 1, so you can't have any regrets, either positive or negative, for playing sigma 1, because your option was playing sigma 1, but you have some regret of not playing sigma 2. Sigma 2 was kind of the winner here. If the two [INAUDIBLE] reversed, we would have r1 equals 0 and r2 equals the negative 1. And then we'd become happy because all our regrets would be non-negative.

So at t equals 2, because we have zero regret here and regret 1 here, we actually pick the strategy to be all sigma 2. Now the adversary says, OK, well the value of sigma 1 is 1, and the value of sigma 2 is 0 for the second time step. So what happens? Well, the same thing happens as before. Now we have regret of 1 on the [INAUDIBLE], and then regret of [INAUDIBLE] on the second option.

So what do we do next? The regret [INAUDIBLE]. Well, flip a coin or just pick a linear even combination of the two strategies, half of one and half the other. That's what we can do. [INAUDIBLE] the same. So now the adversary says sigma 1 is 0 and sigma 2 is 1, which means that the regret of 1 actually goes to 0.5, and the regret of 2 actually goes to 1.5. [INAUDIBLE]. 1 goes down a 1/2.

So now with these regrets our waiting is kind of the ratio of the two. It's 1/4 sigma 1 and 3/4 sigma 2. So now the adversary goes, OK, well sigma is 0. Sigma 2 is 1. So this regret actually goes [INAUDIBLE] down by 3/4, and this goes up by a 1/4. And since this is negative, now we pick the strategy to be sigma 2. [INAUDIBLE] and so forth. Now the adversary [INAUDIBLE] for us and say, oh, it's really sigma 1. Then a regret of sigma 1 would go up to 0.75, and so on and so forth.

So it seems that the adversary can make the job tough on us. Well actually, there is a theorem that says, for our example, [INAUDIBLE]. The square of the first regret if it's positive plus the square of the second regret if it's positive is always going to be less than or equal to t. And that's because, if [INAUDIBLE] these are both positive, it goes, for example, you are really going r1 plus or minus whatever amount of r2 you're doing. And r2 of t now minus plus whatever amount of r1 you're doing. The things that [INAUDIBLE] this you can see the cross terms cancel each other out. This becomes 2 r1 r2 divided by r1 plus rt.

So you're left with this squared plus this squared plus this squared plus this squared. And this squared plus this squared is going to be less than 1, so we have this here, which means that the quadratic sum only [INAUDIBLE] by 1. We have this bound. Why is this bound so great? Well if the square of the regrets are less than t, that means the average regret is going to be [INAUDIBLE] 1 over root t. In fact, it's kind of left as a homework problem. In a general case, our kt over t is less than n minus 1 delta over root t, where delta is the maximum deviation of the options and is just the number of options. Yeah?

AUDIENCE: I'm curious, is [INAUDIBLE] in terms of what is the strategy sigma. Number of like a payoff?

PROFESSOR: No, no, no. A strategy sigma, in terms of poker strategy, is sort of a description of what you would do. Suppose you get ace, six off suit pre-flop. A strategy would be a descriptor of what you would do at each point of the hand.

So there's some significance in effect that this regret, average regret, goes to 0. Well, the significance in terms of game theory optimal is suppose a peer's strategies are-- suppose you have a bunch of peer strategies for x and bunch of peer strategies for y. If we regret match, but instead of doing an adversary, we just say t utility for x is just the utility for x playing against the sigma ty, and the utility for y is just negative utility-- the game utility for y playing against sigma xt. This is kind of a mutual regret matching. You do regret matching for x and y in each step, which means you just modify x-- you compute the regrets at each step. Then you modify x [INAUDIBLE] y strategy by this type of regret matching.

And basically the strategies that you choose, the average strategy, which is the sum of the strategies you have had all along divided by t. 1/t-- all the strategies you've done in these t steps. And basically what happens is now, if you try [INAUDIBLE] to exploit a [INAUDIBLE] strategy, again, this is the best x can do against y minus the best y does against x. You compute this, and you add the sum of what actually happened with x of t and y sub t, and so on and so forth.

You notice that this is the regret of k-- of x picking strategy k all the time. It's just y picking strategy j all the time. So that's less than 2 epsilon over t because regrets over t converge, so it's within [INAUDIBLE] game theory optimal. Basically what this all means is basically suppose you choose your strategy, some mixture of stuff. Your opponent tries to figure out how best he can exploit this strategy. By the way, this is often called nemesis. I really like that name. Opponent figures out his nemesis strategy against you.

Then, well, you get to see-- so his nemesis strategies-- unless you're playing the exact game theory optimal strategies-- is always going to be better than the game value. He looks at what you've done and finds the best response. And you do the same to him, and the difference of those two games kind of exploitable. Obviously, this means basically, if your opponent sees what you're doing, this is the best he can do against you. This number is the one that's less than 1/1000 of a big blind.

So counterfactual regret is kind of cool because-- it's a good thing I've drawn this tree. At each of your decision points, now you can regret match. So first of all, you don't need to be fed back the correct utility [INAUDIBLE]. Here in the example we gave, we had a u0 and u1. You'll just be fed back some unbiased stochastic number that averages the value of the game.

For example, if you're doing a regret chain on poker, it's hard to tell if I'm up with this strategy that has a bunch of terabytes, and you come up with a strategy that's also a bunch of terrabytes-- what's the value of playing against y [INAUDIBLE]? But we can just get a sample. We can get a sample. Well you can just run it once. Right, that's the idea. You get a sample by just saying, OK, just play one hand, and see the result of that hand. And you could use either random chance or whatever every time you decide to do whatever branches of your tree if you do a mix tree. So the cool thing already is, without counterfactual regret, you can quickly converge the solution, because a lot of strategies, like fictitious play-- it's the best response. The best response is hard to calculate sometimes, but each simulation can just be one iteration through it.

And this is counterfactual regret because [INAUDIBLE] is given assuming that the player does everything to play to that node. So the waiting here is nature just has its probabilities. If your opponent plays according to his strategy, but when you play you always kind of play towards that node, so your weight actually 1 for each of these options you pick. The cool thing is that once you have the structure set up where you're just doing one or a few iterations throughout the hand, it's actually pretty easy to set up different weighting schemes.

For example, if you have two options and the ace of hearts comes on the turn, or the deuce of clubs comes on the turn, and you don't really have to worry about the ace of hearts coming on a turn. That tree is fine. That part of the tree has very little positive regrets. You can say, OK, we'll just-- different game where the ace of hearts comes about [INAUDIBLE] at a time the deuce of clubs comes, but we're going to weight the results by 10.

You still get the same answer. It's just that you get a much coarser kind of [INAUDIBLE] every time the ace of hearts comes, but already kind of know what to do with that. You can work on the deuce of clubs. So there a lot of different weightings schemes. This means that the hands can be kind of sampled intrinsically.

So the final algorithm they had was factual regret plus. So instead of having accumulated negative regrets, basically a lot of these option regrets can be really negative. Folding aces pre-flop quickly turns to really negative regret. You lose your small blind, and hopefully if you play it limit hold 'em, you could win more than the small blind. So you accumulate a lot of [INAUDIBLE] so set options falls off the map.

Their innovation in counter factual plus is to, instead of putting a big negative number to a lot of these things, they just floor them at 0. And the reason they floor at 0 is because you know this a simultaneous evolution of strategies where even strategies at the beginning just might not be great strategies, and you want to-- if regret of something is 0, you can route get regret faster if it's the right thing to do to respond to your opponent's strategy. All of these things-- suppose you start with a random initial guess for your opponent's strategy. Then you actually have a pretty reasonable strategy, which is bet and raise every time with every hand. If your opponent has a random strategy, he might just fold.

So later in the streets, it's probably [INAUDIBLE] just bet and raise every time with every hand. He raises you back. It's not like he knows anything. It's a random strategy. Just raise him back and hope he folds. If he doesn't fold and call, you bet again an x3, because now the pot is bigger. So he has a 1/3 chance of folding. You should bet.

So that evolves quick. If you start off with a random tree with no information, that starts off as the dominant strategy. And then you have to walk that back as your opponent's strategy evolves also. By the way, they're actually keeping two trees-- one for the small blind strategy, and one for the big blind strategy. And this is everything with respect to the small blind. The small blind isn't-- so let's just go into the next slide probably. [INAUDIBLE] have to be.

So let's try to figure out how big the strategy space in limit hold 'em has to be. So let's concentrate on river nodes because that's most of the nodes. It's a tree so we just have to calculate the leaves. So first of all, assuming a four bet cap-- the reason we assume a four bet cap-- well, I don't know why, but it seems that that's-- so this is one approximation, the four bet cap, but this is kind of normal in types of research papers. if we have a four bet, there are nine possible actions that get you to the next street. There are some actions that [INAUDIBLE] like player one bets and player two folds, but if you don't get to the street, you don't get to the river, and that's a pretty small percentage of the nodes.

So why are there nine possible actions? Let's count them. One of the actions that gets to the next street is check check. So that's one. What are the eight? What are the other eight?

AUDIENCE: [INAUDIBLE].

PROFESSOR: Right, check raise. Let's try systemic [INAUDIBLE] count them. So I claim that there are two ways-- one bet in the pot. Player one can bet, and player two can call, or player one can check, player two can bet, and player one can call. In fact, there are two ways to put k bets in the pot and k is greater than 0. If you want to put three bets in a pot, what are the two ways?

AUDIENCE: [INAUDIBLE].

PROFESSOR: Right. Yeah, right. Bet, raise, re-raise, call, and check, bet, raise, re-raise, call. So if the cap is k bets, there's always 2k plus 1 ways to [INAUDIBLE] three. So there are nine possible actions in each betting round before the river. So there are three betting rounds-- pre-flop, flop, and turn.

So let's use some symmetries because I don't think the optimal strategy has you playing something differently with ace, six of diamonds, ace, six of heart. [INAUDIBLE] very easy to prove. The optimal strategy doesn't have that. So using symmetries on a flop-- so how many distinct flops are there? Well, I like to think about it as where the suits have symmetries. I like to think about it as, well, there could be three suits in a flop, two suits in a flop, or one suit.

So if there's one suit on a flop, there's 13 [INAUDIBLE] combinations. That's pretty straightforward. If there are two suits on a flop, what's the combinations? There are 13 possibilities for one of the suits, and there are 13 [INAUDIBLE] for the other suit. It's based on heart or something like that. The suits are symmetric.

So there are 1014 things [INAUDIBLE]. This is [INAUDIBLE] the things. And if it's three suited, you just choose three ranks, but it's not 13 choose 2. It's 15 choose 2 because why? I guess the ranks can be equal. So it would 13 choose 2 if the ranks would be unique, but you'd have three aces on him. So this is actually 15 choose 2.

So there's 455 three suited flops, [INAUDIBLE] flops. That's kind of the big explosion in limit hold 'em, pre-flop to flop.

So there is not [INAUDIBLE] possible actions in each betting round. So let's count the number of turns and rivers. There's [INAUDIBLE] turns and 48 rivers. So counting that, you have a billion possible action sequences to the river. The [INAUDIBLE] things in each street, all the flops, then the turns and rivers. But each river, there could be up to 126 [INAUDIBLE] types. 47 times 46. Making about 6.5 trillion hand river types.

Each node should be visited about 1,000 times. It's a big computational problem, but it still tractable, especially if you have 900 years of CPU. And they also used many shortcuts. They use all the symmetries I talk about, and they also have a few shortcuts. And you can see these trees are big. Terabytes of memory to actually store your strategy.

So you can't really get that on a node yet. I don't know. Can you fit that on a node now? Does anybody know? I don't know of a CPU that has [INAUDIBLE] bytes of RAM yet.

What they did was they broke the problem up into about 100 [INAUDIBLE] different sub-games, and they just worked on those sub-games. In fact, I guess if you're clever about it, you can use cache memory when you get down to the river. Things are pretty close, and you know that using cache memory is faster than using [INAUDIBLE] memory. You can take advantage of these things. A lot of these updates through these regrets are just simple addition, and you can just optimize the heck out of this, and I'm sure they did it.

Let's just try to solve some other games. I have two games that seem accessible. Suppose we do Omaha eight. Well, this is exactly the same structure as limit hold 'em. You just change the hole cards. So instead of having 47 choose 2 different river hands, you have 47 choose 4. That's like a multiple 82.5 x to the original tree, so that's not that bad. 900 CPU hours-- this is just 75,000 CPU hours. If it were a matter of national security to get the exact solution to Omaha, the military could just do it in a few months.

There's also [INAUDIBLE] you can do, by the way. Basically, what they did is-- before they did this, was that they solved the sub-game. In that, basically if you both get hands together and you say you have to play these hands the same way, that's basically a sub-strategy. You can consider subspace of your strategy x prime of x and y prime of y, and you just solved the x prime y prime game, meaning you both get hands together, probably on the river because that's when bucketing kind of becomes more necessary.

And you solve that game, and you go, well, how optimal is x prime in the hold game? And if you're good at bucketing, it may be pretty close. If you're bad at bucketing, like you put the aces in the same bucket as seven, five suited, you probably won't get a great answer. So you need to intelligently design your buckets. You can't-- well, I guess there are also evolutionary things you can do to try to design buckets and see what things are close to each other. People who have familiarity with this know that this is kind of hit or miss.

Another game that you can maybe solve is razz. It's definitely as simple as [INAUDIBLE] a stud. Why is razz simpler than all other games of stud? There are only 13 different cards. The deuce of spades is the same card as the deuce of hearts. You can't-- well, you could make flushes, but they're irrelevant.

So unfortunately there are 13 to the 8th power possible ways of cards can come, because there are four up cards. That's sort of the problem. Kind of the community information you have is a bigger set, and your trees just get bigger because now you have one extra street. And you still have 415 choose 3 combinations of any three ranks as river hand types. So There are 2.4 quadrillion river hands. So that's a factor of 374 [INAUDIBLE], but we think some of these roads are pretty null.

How many of you actually play razz? A couple of you. OK, great. Good poker class that people study razz. If you have a queen up and a deuce completes it, you're not really going to get into a raising war and make it [INAUDIBLE] cap on third street. Some of the [INAUDIBLE] may be null. You can do some bucketing, perhaps.

Razz is kind of more natural to bucketing because you can think about what hands to bucket together. Maybe the king, eight, six, deuce is very close to the king, eight, six, ace. And the two strategies-- and you can start in hands by rank order of cards or something like that. So this is 374. This is 82.5. Or you could apply for a grant and say we need x hours of CPU time. I don't know what the right strategy is, but these two problems are tractable.

Let's talk about big bet games because there's been some sort of discussion, even last night, about Snowie. A few people have tried big bet games, and they're problems. First of all, there's a continuum of bet sizes you can make. The Snowie solution just assumes three bet sizes. I can bet half the pot, I can bet the pot, or I can jam, I think. Maybe there's-- I can bet two times the pot, but the problem with that is that I think that's a little bit too coarse. The question is, if you solved that game, how close is that solution to the real game?

And that's kind of an interesting question, but you don't even have a complete strategy. What if some guy bets a quarter of the pot, or 1.5 times the pot, something that' not on your list? You have to exploit-- and then it gets kind of weird, because my response to a pot size bet is to raise the pot again. All right, what if he makes a 1.1 times the pot? Is it right to raise the pot-- just raise the pot 1.1 times or raise the pot 0.9 times so you get back to the same stack sizes so you can do the same thing in the future. These are difficult questions.

Even if some bets [INAUDIBLE] are non-optimal, our full strategy needs responses to to the bets. So simple approximations may work. I kind of feel this is kind of a tough problem, though. And you could just-- just playing a game where you can just make rigid pot bet sizes, then you might get something actually interesting. But one of the things with regret matching, if you actually have a lot of bet sizes, suppose you say, OK, I'm just going to kill this problem, and I'm going to do 0.01 times the pot, 0.02 times the pot, 0.03 times the pot, and so on and so forth.

The problems is now you have a lot of options which are really close in equity together, so this regret minimization is going to take a while. It's going to have to sort out really close events. And then it's going to have to balance your value bets with your bluff and things like that. So even just trying to kill it by putting a lot of bet types may not solve the problem for you.

So two player, three player games are actually kind of interesting. The dress by the group and using counterfactual regret to create competitive multi-player agents. And this is a paper done in 2011 or so. And the program for actually first and second in annual three player limit event-- the first problem is that there's no guarantee of epsilon convergence. You're not necessarily within epsilon of Nash equilibrium.

Second problem is that, do you just want to play in Nash equilibrium? There could be multiple Nash equilibria in multi-way games, especially in these proportional payout tournaments, satellites where, say, two people get a seat. There are really nonlinear effects going, and it could [INAUDIBLE] which collusive equilibria are you playing? In our book, Jared and I point out a game called the rock maniac game where it's a real poker game where players can use a simple strategy and ensure you losing. A simple version non-poker version, like a game where you play even or odds with three players, but the odd man out wins.

So suppose you and I are colluding against the third chump. What would we do?

AUDIENCE: [INAUDIBLE].

PROFESSOR: Right, I would play one, and you'd play two. And the third guy could never win. There are situations which can come up in poker like that, but I think if there's no collusion and it's not a tournament, playing Nash equilibria usually turns out OK. I think that's sort of the argument they were making in creating these strategies.

All right, here are the references. This took about [INAUDIBLE] the time I estimated, so questions? OK, let's just-- you hand your hand up first.

AUDIENCE: Well, the original strategy finds that the Nash equilibria, if you're playing against someone who's trying to beat [INAUDIBLE] strategy-- does it work if one of the strategies is probabilistic. Let's say two strategy trees--

PROFESSOR: Yeah, yeah, yeah. It does work with--

AUDIENCE: Choose [INAUDIBLE], but you don't know always which one I'll choose.

PROFESSOR: Yeah, it works because you're going to play-- all of these strategies assume that they could be mixed strategies. If you're not allowed to play 1/3 rock, 1/3 paper, and 1/3 scissors, then you're going to have to play really bad strategy, and there's definitely times in which mixing is going to be necessary. So, yeah. All of these strategies have mixing. Yeah?

AUDIENCE: What effects do you think [INAUDIBLE] going to have on limit hold 'em games?

PROFESSOR: I don't now. I think pretty much before the solution came out the big online players kind of knew that a lot of people were playing near optimal, and I think the game is kind of dead. What do you think, Mike?

AUDIENCE: [INAUDIBLE].

PROFESSOR: Right. Too bad Matt doesn't come here.

AUDIENCE: [INAUDIBLE] are already basically doing this anyway.

PROFESSOR: Well, no. I mean, even if you have the strategy, you have to learn it. The problem is that, if you go to a casino and you play somebody who's a good limit hold 'em player, he's-- because these types of strategies have been out for a while, they already played much closer to optimal than they did before. So I think this would have absolutely no effect on heads up limit hold 'em. It's already kind of no one-- yes?

AUDIENCE: So can you talk more about different ways you can do approximations. [INAUDIBLE] mentioning earlier bucketing all of the different hands [INAUDIBLE] the ranks or what are some other things we can do?

PROFESSOR: It's an endless [INAUDIBLE] be clever in bucketing. So bucket hand types together.

One kind of clever thing you can do is try to cut out the river entirely by just estimating your equity on the river. Of course, that's not going to be your showdown equity because you may be forced to face of a bet. So you try some sort of implied value of your hand. Let's see. What other bucketing things.

I mean, in some games there's a sort of a natural way of bucketing hand types. Like In the river on Omaha, you could just try to bucket the cards that actually play and ignore the other cards. The thing is that, when you do things like that, [INAUDIBLE] losing assisting, we call it card removal. Card removal and blocking players from having the nuts and things like that are pretty important-- do turn out to be a pretty important part of the game theory optimal solution when you're getting down to the milli big blind kind of level.

And if you don't think about card removal at all, then you actually have a strategy that can be exploited pretty easily. Actually, I talked about this yesterday. The thing is typically when the pot is p and you're facing a bet, you want to make them indifferent to bluffing. He's betting 1 to win p, so you want to call about pr over p plus 1 at the time. If you don't call this much, he's going to bluff and take it. So that's sort of the thing. We're saying the bet is 1 and the pot is p.

So if the pot is 10, and he bets 1, and he takes it more than 1/11 at the time, he's going to just-- [INAUDIBLE] bluff everything. The real problem becomes that, if you don't think about card removal at all, he can start bluffing hands in which he knows it's more likely you have a mediocre hand or something that includes a strong hand. One real example is in PLO when there is a flush on the board, what's a good bluff?

AUDIENCE: You have ace of [INAUDIBLE].

PROFESSOR: Right, you have the ace in a suit. You don't have anything else. That's a great bluff, because you're blocking him from having a great hand, and you're blocking all of his not hands and a lot of his really good hands. And he's much more likely to fold, because if you bet the pot, a lot of his hands he's [INAUDIBLE] himself with [INAUDIBLE] with the nut flush. Oh, I have a natural call. Are you all in? I have the nuts? OK, I call. So that's why card removal is important. Yeah?

AUDIENCE: So is my understanding correct that optimal [INAUDIBLE]?

PROFESSOR: Yes.

AUDIENCE: And has there been any study of optimal [INAUDIBLE].

PROFESSOR: Sort of like utility theory. In poker in general, it's kind of weird. People think a lot about that [INAUDIBLE] what tournament they should enter, what games they should play. But there hasn't been a study really optimizing your own personal utility within the games. The assumption is kind of like, well, I'm going to use all this cool utilities theory [INAUDIBLE] to figure out what game I'm playing. As long as I'm playing the game, I'm just going to try to win the most money. That's sort of been the attitude, and I think that's actually correct for most [INAUDIBLE].

In limit hold 'em, [INAUDIBLE] you need bank rolls of hundreds of bets. You're not going to try to optimize and try to win some fraction of a bet with your utility function by lowering the variance. That is an interesting question, because maybe-- I feel that, if there is some utility consideration-- like maybe in a tournament you feel your chips are non-linear-- maybe you are going to quit playing your marginal hands because of utility considerations.

AUDIENCE: [INAUDIBLE] like the fountain table of major events. They'll go beyond ICM to say maybe I won't coin flip for a $10 edge [INAUDIBLE] step up.

PROFESSOR: I mean, if you use ICM, those utilities are already kind of calculated, but yeah. For example, final table of the main event, I'm not only using ICM, but I'm thinking, well, $3 million-- $4 million compared to $2 million is a much smaller step to me than $2 million is compared to 0 in my own personal utility. Like $0.5 million compared to $2 million versus $2 million compared to $3.5 million. So I need to optimize utility. I mean, yeah. I think that's kind of worthy of study. Yeah?

AUDIENCE: What is it about the analytics of poker that makes it so popular with trading firms? And how does it--

PROFESSOR: Oh, OK. That's a great question.

AUDIENCE: How do you use it professionally, all of this stuff?

PROFESSOR: Well, I mean, I think poker is just kind of-- if you think what one game-- if you could teach traders one game, what one game would represent what traders have to know? Well poker-- there are a lot of actors. There's incomplete information. That's one big thing.

And you do have to do a lot of thinking of what your counter party is doing. If he wants to trade against you, he puts a bid or offer-- some of that is why there's this [INAUDIBLE]. Are you trying to get out of risk? [INAUDIBLE] big position he's trying to get out of, or do you have to be worried about these orders and things like that?

And also poker gives you sort of the skills to trade that-- suppose you know something is worth $10. [INAUDIBLE] you're going to make around it [INAUDIBLE]. Knowing nothing, you might make-- bid [INAUDIBLE] offer at 10/10, which means you're willing to buy the [INAUDIBLE] or sell it at 10/10, but you know something about the counter party. You may know the counter party can be a better buyer than seller or that buying is the risky part [INAUDIBLE] is the risky part. That kind of has a quant.

Also, as a quant, doing poker analytics is very similar to the analysis we do in trading. A lot of this analysis-- how these strategies work, do these strategies really return what we think they return are similar to discussions we have in our trading strategy. I'm glad I'm able to talk to you about this, because if you're interested in doing poker strategies, you'll probably be interested in doing trading strategies, too. Any more questions? Yes?

AUDIENCE: What about doing the deviation from [INAUDIBLE] the [INAUDIBLE] detecting deviation or let's say somebody goes from playing optimally [INAUDIBLE] not playing optimal [INAUDIBLE].

PROFESSOR: Yeah, I mean that's a very interesting thing, and that's actually hard to determine because that feels a little bit harder than this because this is [INAUDIBLE]. It's like

I'm trying to figure out the optimal strategy, and I just play this, and whatever money comes to me comes to me. You open your arms. The money comes to you.

The other thing is, oh, well he's playing badly, so I'm going to go there and take his money. But then if I deviate from optimal, I'm also opening up myself to being exploited. So that's kind of hard. That's much more of a dynamic problem. When does he go on tilt? How long was he on tilt? What evidence do we have that he's on tilt. I know that [INAUDIBLE], the guys in CMU, were looking into some sort of zero loss way to exploit your opponents, because you just figure out when your opponents are playing badly, how much they've given up in playing sub-optimally, and then you go to a [INAUDIBLE]. But you only open up yourself to, say, half the money he's given up, or something like that, playing badly. And the metric is-- so there's some sort of gaming algorithm you can do to do that, but yeah that's definitely another field of study. There are a lot of interesting fields that can come out poker [INAUDIBLE]. All right. I guess that's it.

[APPLAUSE]