Editor: **Michael Wooldridge,** University of Liverpool, mjw@liverpool.ac.uk

Computation and the Prisoner's Dilemma

Michael Wooldridge, *University of Liverpool*

Since it was introduced in the middle of the last century, the prisoner's dilemma has aroused huge interest in the academic community, attracting comment from areas as diverse and seemingly unrelated as biology and moral philosophy. There are two key reasons for this level of interest. First, the game-theoretic analysis of the prisoner's dilemma leads to an outcome (noncooperation) that is worse for all participants than another outcome (cooperation). Second, the prisoner's dilemma seems to reflect many important real-world examples of multiagent interaction, and so the failure to rationally achieve a cooperative outcome seems to have worrying practical implications. Here, I explore how ideas from computer science can be brought to bear on the prisoner's dilemma, and how these ideas can lead to rational cooperation in natural variants of this problem.

The Prisoner's Dilemma

The prisoner's dilemma has two players: Alex and Bob. Each player must choose between two actions: cooperation or defection. Depending on the combination of choices made, the players receive payoffs, as the payoff matrix in Figure 1 shows.

In the matrix, Bob is the row player because his choices correspond to the rows of the matrix, whereas Alex is the column player because his choices correspond to the columns of the matrix. Each cell in the matrix is a possible outcome of the game, corresponding to the combination of choices made by the players. The numbers in a matrix cell are the payoffs that the players receive in that outcome: Bob's first, then Alex's. Thus, if Alex cooperates while Bob defects, we get the outcome in the top right cell of the matrix: Bob gets a payoff of 4, whereas Alex gets a payoff of 1. Players prefer higher payoffs, so this is the best possible outcome for Bob and the worst possible outcome for Alex.

The standard game-theoretic analysis of the prisoner's dilemma goes as follows (further discussion, references, and a gentle introduction to the terminology used here are available elsewhere[1]). Consider the game from Alex's viewpoint. If Bob defects, Alex can choose to defect (giving a payoff of 2) or cooperate (for a payoff of 1). In this case, Alex would do better to defect. If Bob cooperates, however, and Alex chooses to defect, then Alex would get a payoff of 4, whereas if he cooperates he would get 3. No matter what Bob does, the best response for Alex is to defect. Bob's reasoning is identical. Both players thus conclude that, no matter what their counterpart does, their best response is to defect. Thus, both defect, leading to the mutual defection outcome in the top left cell of the payoff matrix. However, this outcome is worse for both players than the mutual cooperation outcome in the bottom right of the payoff matrix. Thus, rational choice seems to lead to an outcome that is manifestly suboptimal for everybody. The (defect, defect) outcome is a Nash equilibrium, and it is the only Nash equilibrium in the prisoner's dilemma. What this means is that, assuming one player chooses to defect, the other can do no better than defect as well. In this article, I use Nash equilibrium as my basic analytical tool for determining what outcomes can rationally occur.

The structure of the prisoner's dilemma seems to reflect many real-world situations. For example, consider the *tragedy of the commons*. Villagers can use an area of common land to graze their cattle. If all the villagers overgraze the common land, it becomes barren; however, if the villagers exercise restraint, the land stays in reasonable shape. The best outcome for me is if you exercise restraint while I overgraze; but you reason likewise. The upshot is the land becomes overgrazed and barren, which is worse for all of us than if we had exercised restraint.

The apparent paradox (that rational choice leads to an outcome that is strictly worse for everybody than another outcome), coupled with the fact that the prisoner's dilemma seems to reflect many

1541-1672/12/$31.00 © 2012 IEEE
Published by the IEEE Computer Society

		Alex	
		Defect	Cooperate
Bob	Defect	2, 2	4, 1
	Cooperate	1, 4	3, 3

Figure 1. Payoff matrix for two players in the prisoner's dilemma game. The result pairs list Bob's result first, then Alex's.

scenarios that occur in real life, has led to the game achieving a rather celebrated status in the game theory community. Many researchers have tried to find some way to recover cooperation—that is, to find some argument for how and why mutual cooperation can rationally occur in the prisoner's dilemma. I introduce two of the more successful such ideas; techniques from computer science feature prominently in both.

Playing Games with Automata

The first idea for recovering cooperation in the prisoner's dilemma is to play the game more than once. In the iterated prisoner's dilemma, the same two agents play a sequence of individual prisoner's dilemma games. After each round, they can each see what the other player did in the previous round, and the payoffs for that round are as defined in the payoff matrix for the "one-shot" prisoner's dilemma. In such a setting, one player can punish another: If you're nasty to me by defecting today, I can be nasty to you tomorrow by defecting then. If I start by cooperating and you start by defecting, this is good for you, but only in the short term. I can punish your naughtiness by defecting against you in all future rounds of the game. You might benefit in the short term, but you lose out in the long term, because you lose the opportunity to cooperate with me in the future. This simple idea is sufficient to obtain mutual cooperation as a rational outcome in the iterated prisoner's dilemma.

One important assumption that we make here relates to how many times the two agents will play the prisoner's dilemma. We assume that they play infinitely often. In practice, of course, this isn't possible, but we can justify it as a modeling assumption by observing that it models situations in which the players are uncertain about exactly how many times they will meet each other in the future. We will comment on this issue again later.

If the players are to play the game infinitely often, this raises the question of how we measure their payoffs. We know what each player will get in each round of the repeated game—one of the values in the prisoner's dilemma payoff matrix. But simply adding the payoffs received in each round will yield infinity if we play the game an infinite number of rounds. How can we compare the success or failure of two different strategies for playing the iterated prisoner's dilemma if both yield an infinite payoff?

There are many ways to answer this question. One possibility is to use *discounting*, in which a payoff of x received today is valued more than a payoff of x received tomorrow (see the "Discounting the Future" sidebar). However, our approach is even simpler: We consider the payoff an agent receives on average over all rounds of the game. As we'll see, this value is often easy to calculate.

The next issue we must consider is what form the strategies chosen by players will take. In the one-shot prisoner's dilemma, the players must simply choose between cooperation (which I'll hereafter denote by *C*) and defection (*D*). However, in the iterated prisoner's dilemma, the players must choose a long-term strategy, which involves selecting *C* or *D* at every round. Because we assume that players can correctly perceive the choices made by their counterpart in the preceding round, a strategy for the iterated prisoner's dilemma can be viewed as a function that maps histories of the game so far to a choice, *C* or *D*, representing the choice made by the player in the current round. We can naturally model such strategies as *finite-state automata* (technically, as Moore machines).[2] To understand how this works, consider the simple automaton in Figure 2a, which behaves rather naïvely in the iterated prisoner's dilemma.

The automaton in Figure 2a has a single state, indicated by the oval. The arrow that goes into this state from the left indicates that this is the initial state of the automaton. When the game begins, the automaton is in this state. Inside the state is an action, which the automaton selects when it's in this state. Thus, initially, this automaton chooses to cooperate (that is, the *C* inside the oval). The two arrows coming from the state correspond to the choices of the counterpart. We follow the arrow labeled *C* to find what this automaton will do if its counterpart chooses *C*, and we follow the arrow labeled *D* to find what this automaton will do if its counterpart chooses *D*. In fact, both arrows lead back to the *C* state, so the overall strategy defined by this automaton is as follows: Initially, do C; then, irrespective of whether my counterpart chooses to do *C* or *D*, choose *C* in all subsequent rounds. This strategy is called ALLC ("always cooperate").

I said that the ALLC strategy is naïve, so now let's see why. Consider the equally simple automaton in

Discounting the Future

How can we assess the value of an infinite sequence of payoffs? Simply summing the individual values won't work, because a nonzero positive infinite sequence will sum to infinity. A standard idea is to use *discounting*. The idea here is that $1 in your hand today is worth more than $1 in your hand tomorrow, so you should value a payoff of $1 today more than you should value a payoff of $1 tomorrow. On reflection, this seems a reasonable reflection of everyday reality. After all, over time, monetary inflation will steadily reduce the value of money in your pocket; a pie that is fresh today will not be so good tomorrow, and so on.

To formally capture this idea, we use the idea of a discount factor, δ, with $0 < \delta < 1$. For example,

- If $\delta = 1$, a payoff of x tomorrow would be worth $\delta x = x$ today.
- If $\delta = 0.5$, a payoff of x tomorrow would be worth $\delta x = 0.5x$ today.
- If $\delta = 0.1$, a payoff of x tomorrow would be worth $\delta x = 0.1x$ today.

Thus, a player with discount factor δ close to 0 is not greatly concerned about the future, because the future has little value for him. He is more focused on the payoff he will receive in the present. However, players whose discount factor δ is close to 1 will be prepared to take a long-term view, because future payoffs will be more significant to them.

Given a discount factor δ, the value of an infinite sequence of payoffs $x_0, x_1, x_2, x_3, \ldots$ is given by $\delta^0 x_0 + \delta^1 x_1 + \delta^2 x_2 + \delta^3 x_3 + \ldots$. Now, in many cases, the sequence of values $x_0, x_1, x_2, x_3, \ldots$ will have some structure. For example, all the values x_i might be the same, or they might consist of a repeated sequence of values. In such cases, we can often derive a simple closed-form expression that gives the value of the infinite sum. When the values x_i are all the same, for example, the sum comes out as $x_i/(1 - \delta)$.

Figure 2b. This automaton is structurally similar to ALLC. It's called ALLD, because it always chooses to defect, no matter what its counterpart does. What happens when these two automata play the iterated prisoner's dilemma against each other? Clearly, they generate the sequence in the first two rows of Figure 3.

The value on the right-hand side of Figure 3 is the average payoff per round received by each strategy. So, we analyze the (ALLC, ALLD) strategy pair. Does it form a Nash equilibrium? If it does, no player would regret his or her choice of automaton. But (ALLC, ALLD) is not a Nash equilibrium. The player who entered ALLC would have done better by entering ALLD, for example. This choice would have given the player an average payoff of 2, rather than the 1 obtained by playing ALLC.

If both players entered ALLD, the history in the bottom two rows of Figure 3 would be generated. The strategy pair (ALLD, ALLD) is a Nash equilibrium: Assuming one player chooses ALLD, the other player can do no better than choose ALLD as well. Readers should convince themselves of this fact before proceeding any further.

So, we've identified one Nash equilibrium of the infinitely repeated

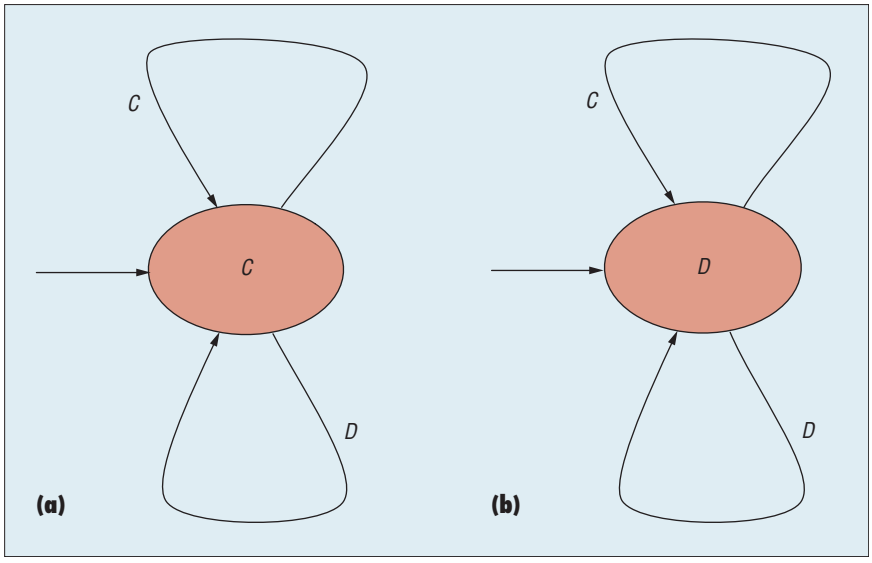

Figure 2. Simple automata for the prisoner's dilemma (a) "always cooperate," or ALLC, and (b) "always defect," or ALLD.

	Round 0	Round 1	Round 2	Round 3	…	Average payoff
ALLC	C	C	C	C		1
ALLD	D	D	D	D		4
ALLD	D	D	D	D		2
ALLD	D	D	D	D		2

Figure 3. Sequences generated when (top two rows) an ALLC and an ALLD automaton play the iterated prisoner's dilemma against each other, and (bottom two rows) two ALLD automata play against each other.

prisoner's dilemma, but it isn't a very interesting one. Rather, it's nothing more than the unique Nash equilibrium of the one-shot game, repeated to infinity. So, are there other Nash equilibria, and in particular, are there more interesting Nash equilibria than this? Yes!

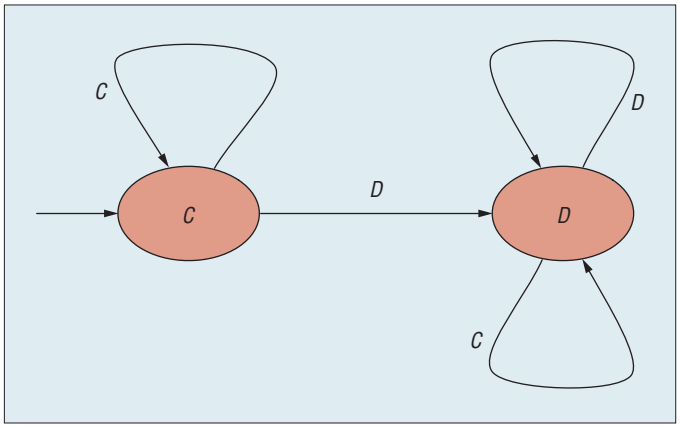

Figure 4. The GRIM automaton cooperates as long as its counterpart does, but once it switches to defection, it stays in that state forever.

Consider the GRIM automaton in Figure 4. It starts by cooperating, and will continue to cooperate as long as its counterpart cooperates. However, if its counterpart ever defects, it will switch to the punishment state, in which it defects, and it will never leave this state. It will continue to defect forever, no matter what its counterpart does. Strategies like this are called *trigger* strategies, for obvious reasons. They capture the essence of the idea that "I'll cooperate as long as you do, but I'll punish you (by defection) as soon as you defect." The top two rows of Figure 5 show what happens when GRIM is played against ALLD.

Although GRIM got the sucker's payoff on the first round, it flipped to its "punishment" state and stayed there. Its average payoff is 2. The one lost utility point on the first round effectively counts for nothing compared to the infinite number of rounds on which it receives a payoff of 2.

The middle two rows of Figure 5 show what happens when GRIM plays against itself. In this case, both players continue to sustain cooperation with each other and receive an overall average payoff of 3 each. Now, crucially, the strategy pair (GRIM, GRIM) forms a Nash equilibrium. If you use the GRIM strategy, I can do no better than use the same strategy. For if there was a strategy yielding a higher payoff, at some point this strategy would have to defect (otherwise it would obtain the same payoff as using GRIM), and this defection would trigger your punishment behavior. My overall average payoff would then be at best 2, as opposed to the 3 I would have obtained had I used GRIM. Thus, mutual cooperation can be rationally sustained in the iterated prisoner's dilemma through the use of trigger strategies such as GRIM. It is important to note that this is rational cooperation. The strategy pair (ALLC, ALLC) generates sustained cooperation, but (ALLC, ALLC) is not a Nash equilibrium. If I choose ALLC, you would do better choosing ALLD rather than ALLC.

This result is one of a general class of results called *Nash folk theorems*. The Nash folk theorems are concerned with the equilibria that can be obtained in repeated games. Put very crudely, the Nash Folk Theorems say something like this: In infinitely repeated games, outcomes in which each player gets on average at least as much as they could ensure for themselves in the component game can be obtained as equilibria. Trigger strategies such as GRIM are key to obtaining these results, just as we have seen in the prisoner's dilemma. A detailed discussion is available elsewhere.[3]

The (GRIM, GRIM) strategy pair is not the only Nash equilibrium strategy pair in the infinitely repeated iterated prisoner's dilemma. We've already seen that (ALLD, ALLD) forms a Nash equilibrium, leading to sustained mutual defection.

The bottom two rows of Figure 5 show what happens when GRIM plays against ALLC. Although we get sustained mutual cooperation, (GRIM, ALLC) is not a Nash equilibrium. The player entering GRIM would have done better to enter ALLD, which would yield an overall average payoff of 4, as opposed to 3.

	Round 0	Round 1	Round 2	Round 3	…	Average payoff
ALLD	D	D	D	D		2
GRIM	C	D	D	D		2
GRIM	C	C	C	C		3
GRIM	C	C	C	C		3
GRIM	C	C	C	C		3
ALLC	C	C	C	C		3

Figure 5. Sequences generated when (top two rows) a GRIM automaton plays against an ALLD automaton, (middle two rows) two GRIM automata play against each other, and (bottom two rows) a GRIM automaton plays against an ALLC automaton.

The World of Dr. Strangelove

Stanley Kubrick's acclaimed 1968 black comedy *Dr. Strangelove* is a film about a trigger strategy. The film is set in the Cold War era, when nuclear war between the USA and USSR seemed a daily possibility. In the movie, a rogue US general initiates a nuclear attack on the USSR. While the US military desperately tries to recall the attack, it transpires that the USSR has secretly installed a doomsday bomb: a device that would be automatically triggered by a nuclear attack and would destroy all life on earth. Their rationale was that the doomsday device would act as a deterrent against possible nuclear attacks, but frustratingly, they had not quite got around to telling anybody about the device before the rogue attack was launched. "The whole point of a doomsday device is lost if you keep it a secret!" exclaims the eponymous US scientist Dr. Strangelove. The film does not have a happy ending.

The doomsday bomb of *Dr. Strangelove* is, in our terminology, nothing more than a trigger strategy. Every day the US and USSR had to choose between cooperation (no attack) and defection (attack). If you attack, the doomsday bomb will punish you forever (once you are dead, you stay dead). The threat keeps you in line. The doomsday bomb was in fact a parody of the entirely serious Cold War doctrine of "mutually assured destruction"—the idea that no side would dare launch a nuclear first strike because the counterattack would ensure that they too were obliterated. I am no expert on Cold War international relations, or indeed on military strategy, but it does seem plausible that the threat of mutual destruction helped to keep the peace during the Cold War era—or at least, helped to prevent the nuclear trigger from being pulled. However, as Dr. Strangelove points out, such trigger strategies can only work if your counterpart knows you are using one. And, of course, trigger strategies can only act as a deterrent if the players of the game are rational ...

The use of a trigger strategy makes sustained rational cooperation possible—but only by the threat of wielding a big stick, in the form of unforgiving, relentless defection. The real world contains genuinely grim examples of trigger strategies, which have important consequences for us all (see "The World of Dr. Strangelove" sidebar).

Before we leave repeated games, let's pause to consider what happens when we play the game a finite number of times. Suppose you and I agree to play 100 rounds of the prisoner's dilemma game. Can mutual cooperation be rationally sustained? No. It's easy to see why, using a standard technique called *backward induction*.

Imagine we are in the last round. I know I won't be meeting you again, so in that round, we're simply playing a one-shot prisoner's dilemma, and of course in the one-shot prisoner's dilemma, mutual defection will occur. So, in the 100th round, both players will defect. But this means that the last "real" round is the 99th round; but again, this means the 99th round is a one-shot prisoner's dilemma. Following this chain of argument, we conclude that mutual defection will occur throughout if we play the prisoner's dilemma a fixed, finite, predetermined, and commonly known number of rounds.

Cooperation in the One-Shot Prisoner's Dilemma

The analysis so far doesn't help us with our original problem: the one-shot prisoner's dilemma. This is because trigger strategies rely on the threat of future punishment, and in the one-shot prisoner's dilemma there is no future. In the one-shot prisoner's dilemma, we must choose between *C* and *D*, whereas what we really want to do is to make a conditional commitment to cooperation. Specifically, we want to make our commitment to cooperate conditional on our counterpart's cooperation—that is, "I'll cooperate if he will." The difficulty is making this idea precise. In a 2004 paper, Moshe Tennenholtz suggested an ingenious solution to this problem, which directly uses ideas from computer science.[4]

Tennenholtz proposed that instead of choosing just *C* or *D*, players should be able to enter a *program strategy*. Such a program strategy is a computer program that takes as input all the program strategies entered by the players in the game. That is, the program strategies of all the players are passed as string parameters to all the players in the game. A program strategy can then make its decision (either *C* or *D*) conditional upon the other players' program strategies. Then, Tennenholtz suggested, suppose you enter the following program (in honor of Tennenholtz, we will call this program Moshe):

```
If HisProgram == MyProgram then
      do (C);
else
      do (D);
end.
```

Here, `HisProgram` is a string variable containing the program text (source code) of the counterpart's program strategy, whereas `MyProgram` is a string variable containing the program text of my own program (that is, the sequence of characters above), and "==" is an ordinary string comparison. Now, if I enter Moshe, what should you do? Suppose you enter the following program (which for obvious reasons we will call ALLD):

```
do (D);
```

In this case, the string comparison test in my program Moshe will fail, and I will choose to defect; you of course will also defect. The upshot is we both get a payoff of 2. But suppose you had also entered the program strategy Moshe. Then the string

comparison would have succeeded, and we would both have cooperated, yielding mutual cooperation. And the program strategy pair (Moshe, Moshe) forms an equilibrium: Neither of us can do better than enter Moshe, assuming the other player enters Moshe! Because if you entered any other program, you would trigger my defection, leaving you with a payoff of at best 2, as opposed to the 3 you would obtain by entering Moshe. We thus get cooperation as a rational outcome in a (kind of) one-shot prisoner's dilemma.

As in the iterated prisoner's dilemma, this is not the only equilibrium. The pair (ALLD, ALLD) is also an equilibrium: If you are going to defect no matter what, I can do no better than to defect no matter what.

On examination, it should be clear that Moshe is a trigger strategy with a structure similar to the GRIM automaton: It punishes its counterpart for failing to exhibit the desired structure. Using such trigger strategies, Tennenholtz was able to prove a version of the Nash folk theorems for one-shot games.

Tennenholtz used the term *program equilibrium* to refer to the kinds of equilibria that can be obtained using program strategies as described earlier. Program equilibria are a relatively new area of research. Program equilibria present many interesting questions for computer scientists and AI researchers. For example, what happens if we allow richer, semantic comparisons of program strategies, rather than the simple string comparison of source code as in Moshe? What other kinds of equilibria can we obtain using such techniques? And what other kinds of applications do program strategies have? ■

References

1. M. Wooldridge, "The Triumph of Rationality," *IEEE Intelligent Systems*, vol. 27, no. 1, 2012, pp. 60–64.
2. A. Rubinstein, "Finite Automata Play the Repeated Prisoner's Dilemma," *J. Economic Theory*, vol. 39, no. 1, 1986, pp. 83–96.
3. M. Osborne and A. Rubinstein, *A Course in Game Theory,* MIT Press, 1994.
4. M. Tennenholtz, "Program Equilibrium," *Games and Economic Behavior*, vol. 49, no. 2, 2004, pp. 363–373.

Michael Wooldridge is a professor of computer science at the University of Liverpool. Contact him at mjw@liverpool.ac.uk.