Evo Edu Outreach (2012) 5:467–476
DOI 10.1007/s12052-012-0434-x

CURRICULUM AND EDUCATION ARTICLE

Oyun: A New, Free Program for Iterated Prisoner's Dilemma Tournaments in the Classroom

Charles H. Pence · Lara Buchak

Published online: 16 August 2012

Abstract Evolutionary applications of game theory present one of the most pedagogically accessible varieties of genuine, contemporary theoretical biology. We present here Oyun (OY-oon, http://charlespence.net/oyun), a program designed to run iterated prisoner's dilemma tournaments, competitions between prisoner's dilemma strategies developed by the students themselves. Using this software, students are able to readily design and tweak their own strategies, and to see how they fare both in round-robin tournaments and in "evolutionary" tournaments, where the scores in a given "generation" directly determine contribution to the population in the next generation. Oyun is freely available, runs on Windows, Mac, and Linux computers, and the process of creating new prisoner's dilemma strategies is both easy to teach and easy for students to grasp. We illustrate with two interesting examples taken from actual use of Oyun in the classroom.

Keywords Game theory · Prisoner's dilemma · Altruism · Natural selection

Oyun was written by CHP and implemented as a teaching tool by LB. This paper is jointly authored.

C. H. Pence (✉)
Program in History and Philosophy of Science,
University of Notre Dame,
453 Geddes Hall,
Notre Dame, IN 46556, USA
e-mail: charles@charlespence.net

L. Buchak
Department of Philosophy,
University of California at Berkeley,
314 Moses Hall #2390,
Berkeley, CA 94720-2390, USA
e-mail: buchak@berkeley.edu

Game-theoretic models in evolutionary theory have been used to explain many diverse phenomena, including altruism (Bowles 2006; Fletcher and Zwick 2007),[1] frequency-dependent selection (Nowak and Sigmund 2004), eusociality (Nowak et al. 2010), and many other facets of evolution (Vincent and Brown 2005). Importantly, they are also one of the most pedagogically accessible varieties of sophisticated biological modeling. We present here Oyun (OY-oon, http://charlespence.net/oyun),[2] a program designed to run iterated prisoner's dilemma tournaments—competitions between strategies developed by the students themselves. Using this software, students are able to readily design and tweak their own prisoner's dilemma strategies, and to see how those strategies fare both in round-robin tournaments and in "evolutionary" tournaments, where the scores in a given "generation" directly determine contribution to the population in the next generation. We describe the motivation behind teaching the iterated prisoner's dilemma, show how students can craft their own strategies in Oyun, and then show the results of some sample tournaments based on two surprising results from the classroom: one showing host/parasite behavior, and the other showing the detection of anti-social behavior.

Background

When we are investigating biological cases in which cooperation of various varieties has evolved, often a pattern

[1] Altruism is here meant in the restricted sense of biological or evolutionary altruism—"costly helping" behavior that decreases fitness (Allchin 2009a, p. 592), not behavior with an "intent to help" more generally (which is often termed "psychological" altruism). See Sober (1988) for a thorough discussion of the scope and limits of evolutionary altruism.

[2] Oyun is a word for "game" in Turkish, Azerbaijani, and allied Turkic languages.

appears—these examples seem particularly amenable to "freeloading" or "cheating." Consider the case of predator inspection in guppies (Dugatkin and Alfieri 1991). In some species of fish, a small number of individuals in a school (the "inspectors") will leave the group to slowly approach and gather information about a much larger possible predator. Now, consider the possible fitness advantage or disadvantage that might hold between each pair of fish in a group. If both fish choose to inspect a predator, they gain information about the predator while sharing the risk. If neither inspect, they lack the information about the predator but neither incurs any risk. But the best outcome of all for any individual fish is the outcome in which the other inspects (incurring all the risk) while it itself stays back. It is best, it seems, for any individual fish to be a freeloader, to refuse to contribute to the group's mutual benefit.

This structure of incentives, as it turns out, is quite common in nature—examples include food gathering, tree height, the expansion of plant roots, body size, and even the replication of virus populations (see Easley and Kleinberg 2010, Chap. 7). Axelrod and Hamilton (1981, p. 1392) describe the same pattern in the differential response of bacteria to environmental change and in the behavior of primates. It is also apparent in human behavior, such as deciding whether to shoot at the enemy during trench warfare in World War I (Axelrod 1984, pp. 73–87). A particularly pedagogically important example is the evolution of behavior that conforms to moral norms—it is easy to see that being a "moral cheater" while those around you do the right thing presents many of the same advantages as the sort of freeloading described here. The importance of teaching the evolution of morality has been stressed by Allchin (2009a, b).

What's more, this network of incentives is equivalent to a well-studied problem in game theory: the prisoner's dilemma. In this example, we are asked to consider two imprisoned members of a gang, one of whom has committed a crime. The state lacks enough evidence to convict either one, so they attempt to get each prisoner to turn the other in. Each prisoner (confined separately and unable to communicate with his partner) is given the choice to remain silent (to "cooperate" with their partner) or to turn state's evidence and testify against the other (to "defect" against their partner). If both cooperate (remain silent), they will each receive a small jail term. If both defect (turn the other in), the state knows that one of them must be lying and gives both a moderate jail sentence. If one defects and another cooperates, however, the defector walks free for his assistance, and the cooperator receives the maximum possible sentence.

We can now formalize this structure using the tools of game theory. Turn the various possible "outcomes" for the two prisoners into numerical "payoff" values—with higher "payoff" for lower jail time. If both prisoners choose to cooperate, they each receive a payoff of three (the light sentence). Should one cooperate and the other defect, the cooperator gets the "sucker's payoff" of zero (the maximum sentence), and the defector gets five (walking away free, the best of all). Should they both defect, however, they both receive a payoff of one (the moderate sentence; see Table 1). So it is better for both if both cooperate than if both defect.

But now, consider whether it is more beneficial for me to cooperate or to defect. If my opponent chooses to defect, then I should defect, to receive a payoff of one instead of zero. On the other hand, if my opponent chooses to cooperate, then I should also defect, to receive a payoff of five instead of three. Defection, therefore, is (in the terminology of game theory) strongly dominant—regardless of what my opponent does, defecting gives me a higher payoff than cooperating, so it seems that I should choose to defect even if I don't know what he'll do. Similar reasoning on my opponent's part leads to the conclusion that he should defect as well. Mutual defection is thus recommended by dominance reasoning, despite the fact that it would be better for both of us if we both were to cooperate rather than defect.

The evolution of cooperation in situations with the structure of the prisoner's dilemma therefore poses an interesting problem for evolutionary theory. For in cases in which phenomena in nature have the structure of the prisoner's dilemma, it seems that cooperation cannot evolve: defecting is always individually beneficial, regardless of what the partner's action is, and defection will thus be evolutionarily favored.[3] And yet, cooperation seems to have evolved in some of these situations—guppies *do* inspect predators.[4] How can we explain these instances of evolved cooperation?

One method for escaping the dilemma was brought to the fore by Axelrod and Hamilton (1981) and led to Axelrod's seminal book, *The Evolution of Cooperation* (1984). The key, as Axelrod and Hamilton argue, is to move to the iterated prisoner's dilemma. In many real-world cases, including the example of predator inspection in guppies, individuals each interact repeatedly with a limited number of others, and so (1) each individual is involved in many prisoner's dilemma-type situations with each other individual, and (2) each individual remembers how other individuals behaved in past interactions. If we add to the model the assumptions that the game is played more than once and that the players keep track of what happened during their

[3] Assuming that the "payoffs" in the dilemma have an impact on fitness values.

[4] Further, the fish can in fact be shown to solve this problem using something very close to the Tit-for-Tat strategy we discuss below—they studiously copy the last move performed by an individual fish (with memory for particular individuals, not just all co-inspectors) and begin by being "nice" (Dugatkin and Alfieri 1991, pp. 307–8).

Table 1 Payoffs for each player ("A, B") in the traditional prisoner's dilemma (Axelrod 1984, p. 8)

	B: Cooperate	B: Defect
A: Cooperate	3, 3 (mutual cooperation)	0, 5 ("sucker's payoff")
A: Defect	5, 0 (defector's payoff)	1, 1 (mutual defection)

previous interactions with each other, then we can allow players' choices in a particular interaction to depend on what happened in the past. And this might give individuals an incentive to cooperate: if they can elicit future cooperation by cooperating now, the long-run payoff associated with cooperating now might be higher than that associated with defecting now.

As long as there is some chance that two individuals will meet again, there is no best strategy independent of the strategy used by the other player. In particular, unless the other player is completely insensitive to what an individual does (duplicating, in effect, the behavior of the non-iterated prisoner's dilemma), "always defect" is not a very attractive strategy. Strategies that can successfully elicit future cooperation from the other player will net a higher payoff. (And strategies that can elicit future cooperation from the other player while cooperating only minimally themselves will net the highest payoff of all.) There is an enormously high number of possible strategies, so the question of which strategy generally does best—which strategy performs well against a wide variety of strategies that may be employed by other players—is difficult to answer analytically. Luckily, though, we can shed some light on it experimentally. To do so, Axelrod (1984) ran a very large iterated prisoner's dilemma tournament. He solicited entrants—computer programs each containing a strategy for playing the iterated prisoner's dilemma—from professional game theorists and received 14 entries in total. He then had each entrant compete head-to-head against each other entrant, for five games of 200 moves each. Each entrant's final score was the sum of its scores in each pair-wise matchup.[5] The strategies with the highest final scores, surprisingly enough, possessed some very "cooperative" characteristics. Most strikingly, the top eight strategies, and none of the others, were "nice": they never played "defect" before their opponent did. Of these top strategies, the most successful were "retaliatory" but "forgiving": they punished what Axelrod called "uncalled for" defection (1984, p. 44) but retained a "propensity to cooperate" with that opponent nonetheless (1984, p. 36). The highest-performing strategy, "Tit-for-Tat," has all three of these features: it cooperates on the first round, and then on each subsequent round repeats the previous action of the player with which it interacts. This implies that it is both retaliatory (if the opponent defects, Tit-for-Tat will defect in the next interaction) and forgiving (if the opponent cooperates, even after a defection, Tit-for-Tat will cooperate in the next interaction).

Axelrod then ran a second tournament with newly solicited entries (this time 62). In this tournament, each pairwise matchup again played the iterated game five times, each time with a finite number of rounds. But here, the number of rounds was determined by setting the probability that two strategies would meet again (i.e., that the game would continue) to 0.99654.[6] Interestingly enough, Tit-for-Tat won again. And this was true even though the results of the first tournament were publicly available and programmers could attempt to specifically devise a strategy that would outperform it. The success of Tit-for-Tat appears to be extremely robust. Here, it seems, is an opening for the evolution of cooperative behavior.

The simple model developed by Axelrod has been discussed extensively throughout the literature. Skyrms has described both the prisoner's dilemma results and a similar game known as the stag hunt in the context of the evolution of cooperation (Skyrms 2003), and has also applied these insights to considering the development of the social contract (Skyrms 1996). Sigmund (2010) has connected the prisoner's dilemma and a handful of other games to phenomena like learning, reputation, repetition, and public goods (such as in the tragedy of the commons).

Further, researchers have repeatedly extended the basic framework presented here in order to provide more robust models of real-world behavior. The tournament has been modified to include choice and refusal of partners (Stanley et al. 1994), to cope with noise in signaling the choice to cooperate or defect (Wu and Axelrod 1995), to interactions between more than two players (Yao and Darween 1995), and to include the effects of the spatial organization of players in the interactions (Ferriere and Michod 1995). Related models in evolutionary game theory have been investigated that explore the punishment of defection (Boyd et al. 2010), the choice to join either a group that punishes or a group that fails to punish (Hauert et al. 2007), the development of reward rather than punishment systems (Rand et al. 2009), and the dispensation of rewards to strangers who have in turn been kind to others (Ule et al. 2009). All these various extensions and expansions derive from the fundamental idea of the iterated prisoner's dilemma tournament.

[5] We should think of the players in Axelrod's tournament as operating under the fiction that the games were of unknown length. Many players in this initial game, in fact, submitted strategies that appeared insensitive both to actual and to expected game length.

[6] The entrants knew this probability but did not know the number of rounds that would be played. This eliminates the possibility of constructing strategies based on knowledge of when the game ends (e.g., 199 cooperates followed by one defect in a 200-round game).

Strategies in Oyun: Finite State Machines

The real beauty of the iterated prisoner's dilemma as a teaching tool for evolutionary game theory is the possibility of students devising their own strategies—matching wits with other players in an attempt to craft a new way to succeed at the game. This is the purpose of Oyun: to provide an environment where iterated prisoner's dilemma strategies can be tested, refined, and explored. And, in particular, Oyun allows for a class-wide tournament in which each student can participate and in which the whole class can see and discuss which strategies were successful and why as a way of seeing which behavior might be evolutionarily selected for.

The participants in Axelrod's original tournaments mostly had an extensive background in computer programming. So the challenge in creating a pedagogically useful computer-run prisoner's dilemma tournament is to find a simple way to represent even very complex strategies. Oyun solves this problem by implementing strategies as finite state machines (FSMs), which are simple, intuitive, and easy to represent visually. A finite state machine for playing the iterated prisoner's dilemma consists of a set of numbered "states," each of which is accompanied by instructions that specify (1) what act to perform in that state (cooperate or defect) and (2) how to respond to the other player's act in that state (which state to move into if the other player cooperates and which state to move into if the other player defects). Consider the classic Tit-for-Tat strategy (TfT) described above, which begins by cooperating and then responds by duplicating the action that the opponent performed in the last turn. Taking the initial state to be state #0,[7] we might diagram the finite state machine for TfT as in Fig. 1. In state #0, the player cooperates. Looking at the arrows originating from that state, we can see that if the opponent cooperates, the player will remain in state #0 and so cooperate on the next round. If the opponent defects, the player will move into state #1 and therefore defect on the next round. And similarly when in state #1: if the opponent cooperates, the player will move to state #0, and therefore cooperate on the next round, and if the opponent defects, the player will remain in state #1. This finite state machine thus "encodes" the Tit-for-Tat strategy.

There is a simple way to textually represent such a finite state machine so that a computer can quickly run each finite state machine against each other in an iterated tournament. Simply list the three important facts about each state: the action and where the two arrows lead. Oyun uses this very simple format to encode finite state machines. In a plain text

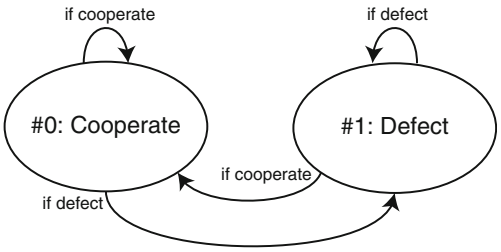

Fig. 1 Diagram for a Tit-for-Tat finite state machine

file, each entrant lists, on separate lines, (1) the author's name, (2) the name of the strategy, and (3) the number of states in the finite state machine. Next, (4) the information for each state is listed on a separate line, in numeric order beginning with state #0, in the following format: "action, {state to transition to if opponent cooperates}, {state to transition to if opponent defects}." The finite state machine in Fig. 1 could thus be represented as follows:

```
John Doe
Tit-For-Tat
2
C, 0, 1
D, 0, 1
```

With a set of strategies gathered in this format, Oyun automatically runs a tournament.[8] Therefore, Oyun requires minimal technical expertise on the part of both the student and the instructor.

Running Tournaments with Oyun

Oyun allows for two varieties of tournaments. The first, paralleling Axelrod's original tournament, is a round-robin tournament, where every strategy is pitted against every other strategy. In this type of tournament, each match is formed of five "sub-matches," whose lengths are each fixed in Oyun. These lengths were determined by using Axelrod's game-end probability of 0.00346 and are set at 168, 359, 306, 622, and 319 rounds. At the end of the tournament, the points earned by each strategy in all its matches are summed, and the strategy with the highest score is the victor. Full data on the points scored by each strategy, and even the precise sequence of moves executed during the games, may be studied, or saved in a variety of export

[7] This "zero-indexing" convention is common in computer programming. We can think colloquially of the "zero-state" as that which a player performs with no knowledge whatsoever of an opponent.

[8] Oyun also has two strategies built in: the Tit-for-Tat strategy and a random player, which chooses cooperate or defect randomly, with equal probability. The instructor can choose to include these in the tournament or not. Note that Oyun does not currently provide a way for students to submit "probabilistic" strategies, strategies in which a player's action on a given round is not determined by the history of the game but only probabilistically related to it.

formats. We will see two example sets of results from a round-robin tournament in the next section, in Figs. 2 and 4.

The second variety of tournament is an "evolutionary" tournament. In this tournament, Oyun begins with a "population" which is uniformly distributed across all submitted strategies. For each strategy *S*, at each generation, Oyun then calculates a "relative fitness" as follows: for each possible opponent *O*, compute the score that *S* receives against *O*, multiply the result by the prevalence of *O* in the population, and sum over each *O* (*S*'s "absolute fitness"); then, set the prevalence of the strategy *S* in the next generation to its relative fitness (its absolute fitness score divided by the sum of absolute fitnesses over all strategies) in the current generation. Put more colloquially, we begin with an equally mixed population and then compute how well each type does at each generation, converting the payoffs directly into the population frequencies in the next generation. Oyun repeats this process for as many generations as the user specifies, and records and graphs the relative fitness of each strategy over time. The results from this type of tournament may be saved as a graph or a spreadsheet for further analysis. We will also see a sample result from an evolutionary tournament in the next section, in Fig. 3.

Oyun in the Classroom: Two Examples

This section presents two examples of the kind of ingenuity that may be deployed by students in developing prisoner's dilemma strategies. At the time of writing, the iterated prisoner's dilemma tournament has been run three times using Oyun, once at Princeton University and twice at UC Berkeley.[9] In each case, after having read and discussed Axelrod (1984), the students were instructed to create finite state machine strategies for a tournament of the sort described earlier. The expectation, of course, is that Tit-for-Tat will prove victorious, as it did in Axelrod's large tournament. In fact, this was not the case: even though TfT was submitted by many students in each tournament, in none of the tournaments thus far has it been the victor. Indeed, the students came up with two separate sorts of innovations that allowed their strategies to beat TfT.

The first innovation involved several students coordinating with each other to submit complementary strategies (this occurred both in the Princeton tournament and in the first Berkeley tournament—in the latter, two separate groups of students coordinated their strategies).[10] In each case, entrants from the colluding group recognized each other via a "secret handshake." The handshake consisted in each entrant performing a seemingly random series of moves, provided it continued to get the "right" response from the other entrant on each round. Once the handshake was completed and colluders recognized each other, they executed the following strategy. One of the colluders became the "parasite," entering an all-defect loop, and the other colluders all become the "hosts," entering an all-cooperate loop. This resulted in a massive payoff for the parasite: on each round after the handshake, the parasite received five points and the host zero. Given that the non-colluding tournament entrants were each receiving three points per round for cooperating with each other and were unable to consistently take advantage of each other, a parasite won each round-robin tournament by a large margin (see Fig. 2 for a reproduction of the output from a similar tournament, restricted to one parasite, one host, a Tit-for-Tat player, and a Random player; the host and parasite strategies are reproduced in the Appendix).

The results are equally interesting when we move from a round-robin to an evolutionary tournament. While the parasite is highly successful in early generations, the host has helped the parasite at the expense of almost all of its possible payoff, and so its fraction in future generations declines very quickly (Fig. 4). How does the parasite perform? Each of the three groups who colluded had a different strategy, and this made a difference in how the parasite performed over the long run.

One interesting difference between the groups that colluded was in how they programmed the "hosts" to respond to players they recognized not to be the parasite. Some students designed their hosts to play all-defect when the secret handshake failed, and others designed their hosts to play Tit-for-Tat when the secret handshake failed (note that in all of the cases, the parasite played Tit-for-Tat when the handshake failed, since the students reasoned that Tit-for-Tat would be the most successful "single" strategy). The former strategy led to the parasite winning by a higher margin in the non-evolutionary tournament, but the latter strategy kept the hosts in the evolutionary tournament longer, and thus gave the parasite a boost for more rounds. Another difference lay in the secret handshake itself. Recall that a "nice" strategy, that is never the first to defect, generally outperforms a "nasty" strategy. Two of the three groups made the parasite's side of the secret handshake nice: it did

[9] The Princeton tournament was run during Adam Elga's "Philosophy of Science" course in spring 2005, and the Berkeley tournaments were run during –Lara Buchak's- "Philosophy and Game Theory" course in spring 2009 and fall 2010.

[10] A similar effect occurred in a "20th-anniversary" version of Axelrod's tournament run at the 2004 Conference on Evolutionary Computation (Kendall et al. 2007). Of course, one might wonder about the rationality of the strategies of some of the group members (the "hosts," a designation we will explain shortly).

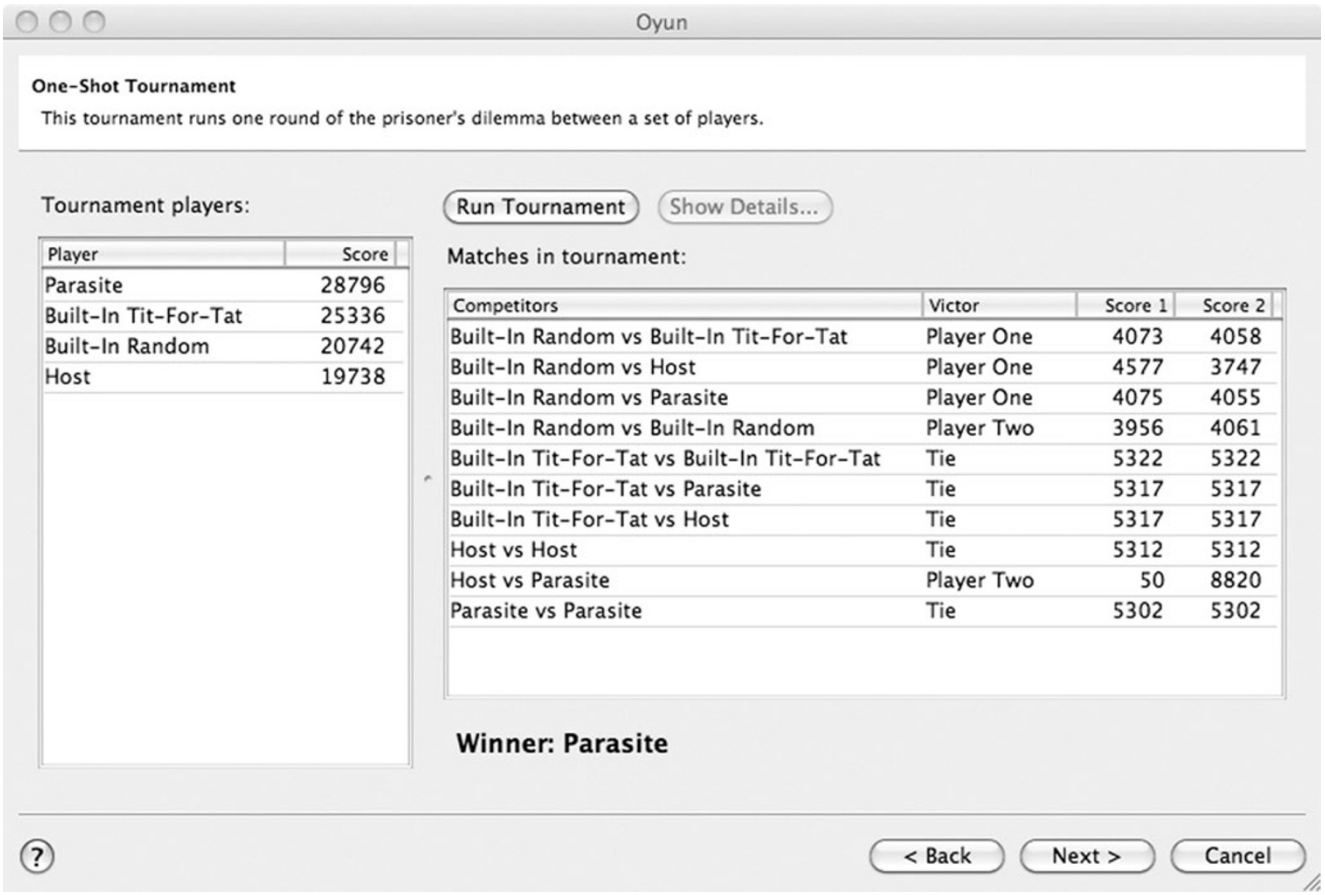

Fig. 2 Example results demonstrating the victory of a "parasite" finite state machine in a one-shot tournament. Note especially the score in the Host vs. Parasite matchup: 8,820 points to the parasite, 60% higher than the all-cooperate score of 5,322

not defect until after there was a defection in the other side of the secret handshake. This meant that the parasite would be nice generally: it would never be the first to defect against non-colluding players, either. For the other group, the parasite was the first to defect even against non-colluding entrants, and so was slowly driven out of the population by the ever-so-slight inefficiency of its secret handshake. Figure 3 represents the result of an evolutionary tournament in which the parasite was "nasty" (in this example, both the host and the parasite play Tit-for-Tat when the secret handshake goes foul). Nasty parasites, as we can see, are driven out of the population by TfT in the long run.

The strategy implemented by the parasite and (willing) host mimics some behavior in nature. Most saliently, it seems to mimic that of species like the social hymenopterans (such as ants, bees, and wasps), where some individuals (sterile workers) seem to willingly take on a massive fitness cost (namely, sterility) (Keller and Chapuisat 2010). Of course, the dynamics of the evolutionary process in our simulations here are far too simplistic to effectively model the emergence of this sort of eusociality, since, for example, in our simulation organisms can only produce future organisms of their own strategy type. Still, examining strategies like these provides an inroad into understanding the fitness benefits of treating members of an in-group differently from those of an out-group and why organisms might evolve to do so (see, e.g., Page and Mitchell 1998).

The other interesting innovation, which occurred in both Berkeley tournaments, was the introduction of a strategy feature that seemingly improved on Tit-for-Tat. If we remove the host/parasite group strategies from the two tournaments that included them and also consider the third tournament, then it is still the case that in none of these tournaments did Tit-for-Tat win. In the modified Princeton tournament, Tit-for-Two-Tats won [as it would have, incidentally, had it been submitted to Axelrod's first tournament (Axelrod 1984, p. 39)], and in the Berkeley tournaments, two strategies called AW and Consolation Prizefighter won (all three of these strategies are reproduced in the Appendix). All three strategies share with Tit-for-Tat the features of being nice and of responding to enough defections with a "retaliatory" defection. The Berkeley strategies were particularly interesting because of two ways in which they were different from Tit-for-Tat. The first difference is

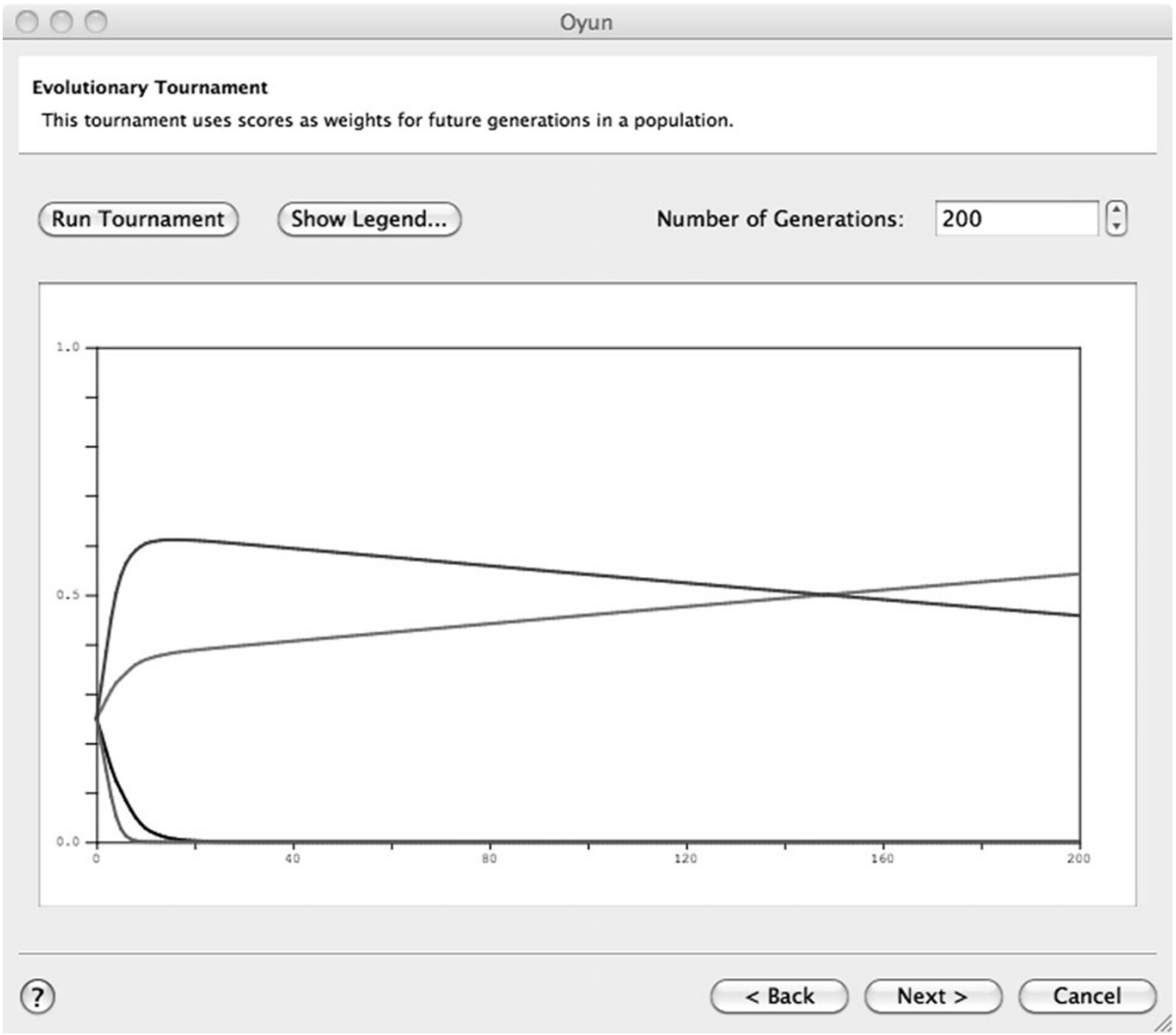

Fig. 3 Example results taking the same strategies from Fig. 2 and placing them in a 200-generation evolutionary tournament. The host is the lowest (worst-performing) line on the graph, only slightly outcompeted by the random strategy. We see the parasite (*top line at left*) initially perform very well, but eventually be outcompeted by Tit-for-Tat (the *center line*) around generation 145

that they tried to restore cooperation after a "retaliatory" defection by including "unsolicited" cooperation. But the really interesting difference is that once the opponent defected enough total times, they each reverted to all-defect for the rest of the game. These strategies were therefore successful not only because they were good at inducing and maintaining cooperation but also because they were good at recognizing strategies that would not respond to inducement (including totally unresponsive strategies like Random) and spent the rest of the game defecting against them. And this seems to mirror an adaptive feature of organisms, often called "cheater detection": if another individual cannot be induced to cooperate regularly, abandon efforts to promote social behavior (see, e.g., Stevens and Hauser 2004).

Implementation and Use

Oyun is available free of charge at http://charlespence.net/oyun and is designed to be cross-platform, running on Windows, Mac OS X, and Linux computers. It is open-source, and the source code is available for download under the GNU General Public License.

Available on the Oyun website are a variety of materials for users. The user's manual can be read online, and a PDF

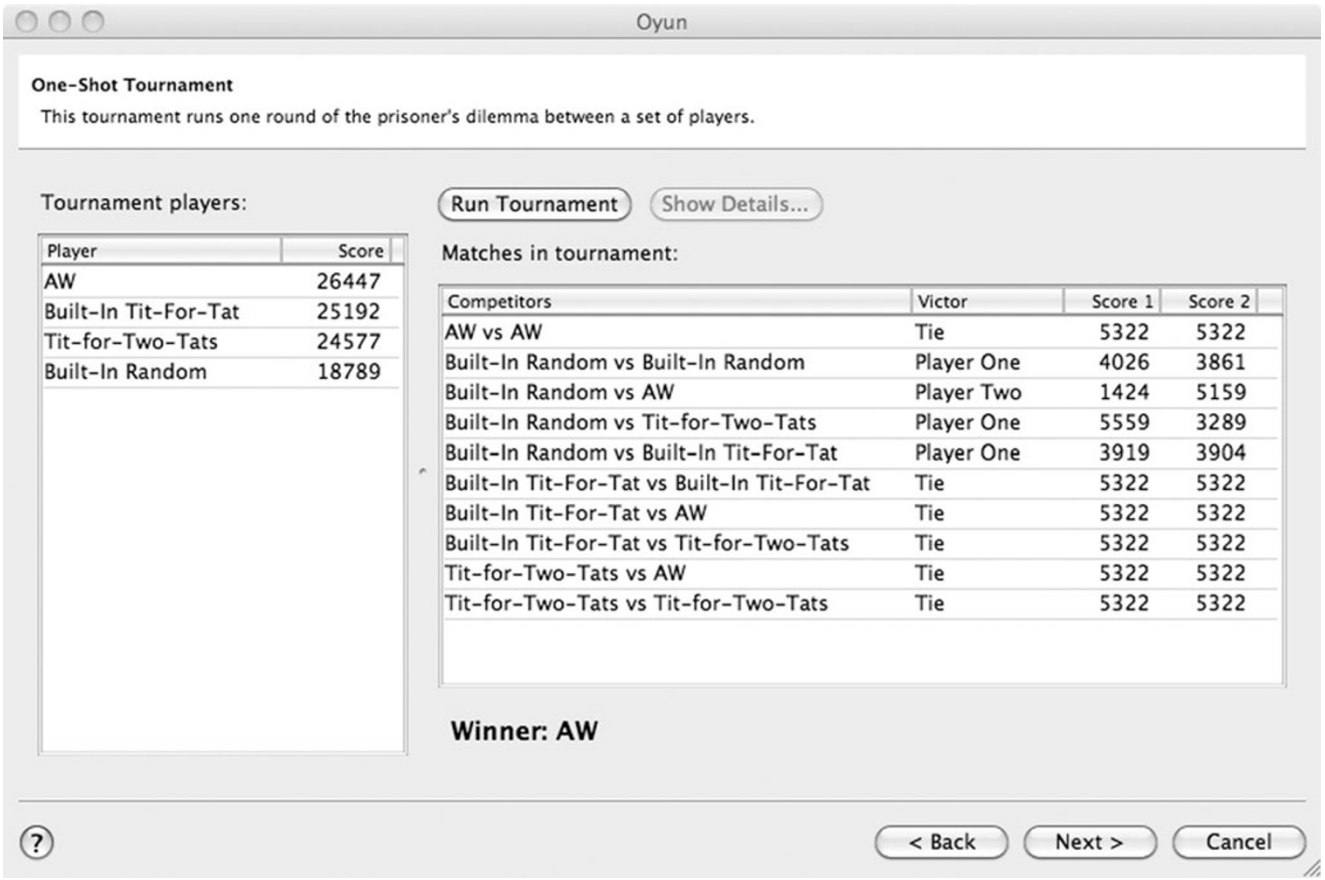

Fig. 4 Example results from a round-robin tournament including a strategy that beats Tit-for-Tat outright (in this case, "AW"). Note that when AW plays Tit-for-Tat, it scores the same 5,322 points as Tit-for-Tat scores against itself (the all-cooperate score). Against Random, however, AW scores 5,159 points (due to its all-defect trigger), while Tit-for-Tat scores only 3,904

copy of the manual is included when one downloads the program installer (for Mac or Windows). This manual includes screenshots and descriptions of each of the steps required to run the various varieties of tournaments described here.

To execute a classroom tournament, only the instructor needs to have the Oyun software installed, though students may install the software themselves to familiarize themselves with the format and test out various strategies. Understanding the prisoner's dilemma requires relatively little background in game theory, and the finite state machine format can be introduced in less than a class period to students with no background in computer science or college-level mathematics; however, some level of basic technical competence is needed. The example strategies that are available on the Oyun website can be used to illustrate how finite state machines work. Once all the entrants have been turned in to the instructor, a second class period suffices to run the tournament in public and discuss the results with the students. The instructor must run the tournament in advance herself in order to verify that there are no improperly formatted strategies; to discern the interesting "global" trends, such as the relative performance of nasty and nice strategies; and to pick examples of how strategies performed against each other to discuss in class. The running of almost all varieties of tournaments (we have tested sizes up to 40 or 50 entrants) is nearly instantaneous—this can be effectively performed as a demonstration on a laptop connected to an overhead projection system. However, since larger tournaments make the data complicated to look at, we recommend the instructor prepare summary statistics or simulations with a smaller subset of the strategies in advance for ease of discussion.

The authors are glad to provide support for the software in the future. If users have trouble downloading, installing, or using Oyun, they should first check the website to see if their issue has been discussed there. Contact information is available in this article and on the website.

Conclusions

Oyun is a freely available, cross-platform program that makes it simple for students to deploy iterated prisoner's

dilemma strategies in both traditional, Axelrod-type tournaments, and "evolutionary" tournaments that make the connection between evolutionary game theory and population change explicit. At the present time, Oyun has met with success in multiple classroom deployments, allowing students to experience evolutionary modeling in a new and exciting way. We anticipate that it can potentially be a useful tool for a wide variety of teachers in different classroom environments.

Acknowledgments Special thanks to Adam Elga for the inspiration to write this program, for introducing the finite state machine format, and for running the first Oyun tournament. Thanks also to the students of Philosophy 321 in Spring 2005 at Princeton University and of Philosophy 141 in Spring 2009 and Fall 2010 at the University of California at Berkeley for participating in tournaments using Oyun. Particular thanks go to students who created the winning entrants discussed in this paper: solo winning entrants, Angelo Wong and Robert Justin Sutton; and group winning entrants, Daniel Greco, Brian Hedden, Philip Kidd, Amar Trivedi, Peter Epstein, Max Gee, Kevin Lee, Philip Hwang, Joe Busby, and Alex Kozak.

Appendix: Selected Student Entries

```
Jane Doe
Tit-for-Two-Tats
3
C, 0, 1
C, 1, 2
D, 0, 1
```

```
Urocerus gigas
Host
12
C, 1, 11
C, 2, 11
C, 3, 11
D, 7, 4
C, 5, 6
C, 5, 5
C, 6, 6
C, 8, 8
C, 9, 9
C, 10, 10
C, 10, 11
D, 10, 11
```

```
Rhyssa persuasoria
Parasite
12
C, 1, 11
C, 2, 11
C, 3, 11
D, 7, 4
D, 5, 6
D, 5, 5
C, 6, 6
C, 8, 8
C, 9, 9
C, 10, 10
C, 10, 11
D, 10, 11
```

```
Angelo Wong
AW
19
C, 0, 1
D, 1, 2
C, 2, 3
C, 3, 4
D, 4, 5
C, 5, 6
C, 6, 7
C, 7, 8
C, 8, 9
D, 9, 10
C, 10, 11
C, 11, 12
C, 12, 13
C, 13, 14
C, 14, 15
C, 15, 16
C, 16, 17
C, 17, 18
D, 18, 18
```

```
Robert Justin Sutton
Consolation Prizefighter
20
C, 0, 1
D, 2, 2
C, 3, 3
C, 3, 4
D, 5, 5
D, 6, 6
C, 7, 7
C, 7, 8
D, 9, 9
D, 10, 10
D, 11, 11
C, 12, 12
C, 12, 13
D, 14, 14
D, 15, 15
D, 16, 16
D, 17, 16
D, 18, 19
C, 18, 19
D, 19, 19
```

References

Allchin D. The evolution of morality. Evo Edu Outreach. 2009a;2:590–601.

Allchin D. Why we need to teach the evolution of morality. Evo Edu Outreach. 2009b;2:622–8.

Axelrod R. The evolution of cooperation. New York: Basic Books; 1984.

Axelrod R, Hamilton WD. The evolution of cooperation. Science. 1981;211:1390–6.

Bowles S. Group competition, reproductive leveling, and the evolution of human altruism. Science. 2006;314:1569–72.

Boyd R, Gintis H, Bowles S. Coordinated punishment of defectors sustains cooperation and can proliferate when rare. Science. 2010;328:617–20.

Dugatkin LA, Alfieri M. Tit-for-tat in guppies (*Poecilia reticulata*): the relative nature of cooperation and defection during predator inspection. Evol Ecol. 1991;5:300–9.

Easley D, Kleinberg J. Networks, crowds, and markets. Cambridge: Cambridge University Press; 2010.

Ferriere R, Michod RE. Invading wave of cooperation in a spatial iterated prisoner's dilemma. Proc R Soc Lond B. 1995;259:77–83.

Fletcher JA, Zwick M. The evolution of altruism: game theory in multilevel selection and inclusive fitness. J Theor Biol. 2007;245:26–36.

Hauert C, Traulsen A, Brandt H, Nowak MA, Sigmund K. Via freedom to coercion: the emergence of costly punishment. Science. 2007;316:1905–7.

Keller L, Chapuisat M. Eusociality and cooperation. In: Encyclopedia of life sciences. Chichester: Wiley; 2010.

Kendall G, Yao X, Chong SY. The iterated prisoner's dilemma: 20 years on. Singapore: World Scientific; 2007.

Nowak MA, Sigmund K. Evolutionary dynamics of biological games. Science. 2004;303:793–9.

Nowak MA, Tarnita CE, Wilson EO. The evolution of eusociality. Nature. 2010;466:1057–62.

Page RE, Mitchell SD. Self-organization and the evolution of division of labor. Apidologie. 1998;29:171–90.

Rand DG, Dreber A, Ellingsen T, Fudenberg D, Nowak MA. Positive interactions promote public cooperation. Science. 2009;325:1272–5.

Sigmund K. The calculus of selfishness. Princeton: Princeton University Press; 2010.

Skyrms B. The evolution of the social contract. Cambridge: Cambridge University Press; 1996.

Skyrms B. The stag hunt and the evolution of social structure. Cambridge: Cambridge University Press; 2003.

Sober E. What is evolutionary altruism? In: Linksky B, Matthen M, editors. New essays on philosophy and biology. Calgary: University of Calgary Press; 1988. p. 75–99.

Stanley EA, Ashlock D, Tesfatsion L. Iterated prisoner's dilemma with choice and refusal of partners. In: Langton C, editor. Artifical life III, volume XVII, Santa Fe Institute studies in the sciences of complexity. Reading: Addison-Wesley; 1994.

Stevens JR, Hauser MD. Why be nice? Psychological constraints on the evolution of altruism. Trends Cogn Sci. 2004;8:60–5.

Ule A, Schram A, Riedl A, Cason TN. Indirect punishment and generosity toward strangers. Science. 2009;326:1701–4.

Vincent TL, Brown JS. Evolutionary game theory, natural selection, and Darwinian dynamics. Cambridge: Cambridge University Press; 2005.

Wu J, Axelrod R. How to cope with noise in the iterated prisoner's dilemma. J Confl Resolut. 1995;39:183–9.

Yao X, Darween PJ. An experimental study of N-person iterated prisoner's dilemma games. In: Yao X, editor. Progress in evolutionary computation. Berlin: Springer; 1995.