# Near-Optimal Play in a Social Learning Game

**Ryan Carr, Eric Raboin, Austin Parker, Dana Nau**
Department of Computer Science, University of Maryland
College Park, MD, 20742, USA
{carr2,eraboin,austinjp,nau}@cs.umd.edu

## Abstract

We provide an algorithm to compute near-optimal strategies for the Cultaptation social learning game. We show that the strategies produced by our algorithm are near-optimal, both in their expected utility and their expected reproductive success. We show how our algorithm can be used to provide insight into evolutionary conditions under which learning is best done by copying others, versus the conditions under which learning is best done by trial-and-error.

**Keywords**: Evolutionary Games, Social Learning, Cultaptation, Lookahead Algorithms, Near-Optimal Strategies

## Introduction

*Social learning*, in which members of a society learn by observing the behavior of others, is an important foundation for human culture, and is observed in many other species as well. It seems natural to assume that social learning evolved due to the inherent superiority of copying others' success rather than learning on one's own via trial-and-error innovation. However, there has also been substantial work questioning this intuition (Boyd and Richerson 1995; Laland 2004; Barnard and Sibly 1981; Nettle 2006; Giraldeau, Valone, and Templeton 2002). For example, blindly copying information from others is not useful if the information is wrong—or if it once was right but has since become outdated. Under what conditions does social learning outperform trial-and-error learning, and what kinds of social-learning strategies are likely to perform well?

One attempt to gain insight into these questions is an evolutionary simulation called The Social Learning Strategies Tournament (Boyd et al. 2008),[1] which was created in order to study the conditions under which communication outperforms trial-and-error and vice-versa. More than 100 researchers worldwide have entered strategies in the tournament, vying for a €10,000 prize. To date, the tournament's organizers have not yet finished evaluating the strategies.

Moves in the social learning game are highly simplified analogs of the following real-world activities: spending time and resources to learn something new, learning something from another player, and exploiting learned knowledge. By developing a formal way of analyzing this set of activities, we hope it will allow us to perform case studies, and to identify how different patterns of behavior fare in different environments.

This paper makes the following contributions to knowledge about the social learning game. First, we have derived a formula for approximating (to within any $\epsilon > 0$) the expected utility of a strategy in the social learning game. Second, we have produced an algorithm that incorporates a lookahead search to find near-optimal strategies. Third, we have shown that locally optimal moves are not necessarily optimal in the long term, but one can derive an upper bound on how many levels of lookahead are needed to find a globally optimal move. Fourth, we have given proofs of correctness and big-$O$ runtime analyses for our algorithms.

## Definitions

This section gives a more detailed description of the social learning game, adapted from (Boyd et al. 2008). The game is an $n$-player round based game, where one move is made by each agent each round. No agent knows of any other agent's moves at any point in the game except through the observation action specified below. The moves available to each agent are: to innovate ($Inv$), to observe ($Obs$), or to exploit one of $mvs$ possible moves ($X[1], \ldots, X[mvs]$). Each $Inv$ and $Obs$ move informs the agent what the utility would be for one of the $mvs$ exploit moves, while each exploit move $X[i]$ provides utility. A player may only make an exploit move $X[i]$ that she has learned of through an innovate or an observe move.

To further complicate things, on round $j$ of game play each move's utility will be replaced with another utility with probability $c(j)$. The function $c$ specifies the *probability of change* for every round of the game. $c$ is provided as a parameter to the game. These changes are invisible to the agents playing the game until the agent interacts with the changed move. For instance: if a move's value happens to change on the same round it is exploited, the player receives the new utility, and is informed of the change.

The number of agents in the game will be $plrs$. Each agent $a_i$ can make two other moves apart from the $mvs$ exploit moves: innovate ($Inv$) and observe ($Obs$). Upon making an innovate move, the agent discovers the value of an exploit move $X[i]$ chosen at random from the set of all exploit



[1] NOTE: None of us is affiliated with the tournament in any way.

moves that the agent doesn't already have information about. When an agent already knows of all exploit moves then $Inv$ is illegal, and indeed undesirable (when there is nothing left to innovate, why innovate?). The agent receives no utility on any round where she makes an $Inv$ move. Upon making an observe move, an agent will get to observe the value received by some other agent who made any exploit move on the last round. Agents receive no utility for observe moves, nor any information other than the move innovated and its value: the agent being observed, for instance, is unknown. If no other agent made an exploit move last round (no $X[i]$ moves were made), the observing agent receives no information.

Innovate and observe moves are essential precursors to exploitation: it is against the rules to make any exploitation move $X[i]$ unless that move has been discovered via an observation or an innovation.

**Example 1.** *Consider two strategies: the innovate-once strategy (hereafter I1), which innovates exactly once and exploits that innovated move (whatever it is) for the rest of the game, and the innovate-twice-observe-once strategy (hereafter I2O), which innovates twice, observes once, and exploits the highest valued move of the moves discovered for the rest of the game. For simplicity of exposition, we allow only four exploit moves:* $X[1]$, $X[2]$, $X[3]$, *and* $X[4]$*; and exactly two agents, one I1 and one I2O. We suppose a uniform distribution over* $[1, 10]$ *(with mean* $5$*) and a probability of change of* $0$ *on every round. The initial (randomly chosen) utilities for the exploit moves are:* $X[1] : 3, X[2] : 5, X[3] : 8, X[4] : 5$*. On the very first move, I1 will make an innovate, which we suppose gives I1 the value of move* $X[1]$*. On every sequential move, I1 will make move* $X[1]$*, exploiting the initial investment. If the agent dies* $k$ *rounds latter, then the history of moves and utilities will be that given in Table 1; giving a utilities of* $3 \cdot (k-1)$ *and a per-round utility of* $3\frac{k-1}{k}$*.*

*In contrast, I2O will make an innovate, giving the value for move* $X[3]$*:* $8$*, then makes another innovate giving the value for move* $X[4]$*:* $5$*, and finally observes. On the second round, I1 made move* $X[1]$*, and since these are the only two agents, this was the only exploit move made. Therefore I2O observes that another agent got a utility of* $3$ *from move* $X[1]$ *last round. On move 4, I2O's then knows of moves* $X[1]$*,* $X[3]$*, and* $X[4]$*, with utilities of* $3$*,* $8$*, and* $5$*, respectively. Since the probability of change is* $0$*, the obvious best move is move* $3$*, which I2O makes for the rest of her life. The utility of I2O on round* $k$ *is* $8 \cdot (k-3)$*, making the total utility* $8\frac{k-3}{k}$*, so for rounds 2 to 4, I2O will actually have a worse utility than I1, while after round 4, the utility of I2O will be higher.*

Formally, all of the information each agent receives on each round can be described by a triple $\langle act, mv, v\rangle$, where $act$ is whatever action the agent chose to perform on that round ($Inv$, $Obs$, or some $X[i]$), $mv$ is an exploit move or a null value ($X[1], \ldots, X[mvs], \emptyset$), and $v$ is the utility observed or received. While $act$ is chosen by the agent, $mv$ and $v$ are percepts the agent receives in response to that choice. When $act$ is $Inv$ or $Obs$, then $v$ is the utility of exploit move $mv$. If $act$ is $Obs$ and no agent made an exploit move last round, then there is no exploit move to be observed, represented with $mv = \emptyset$ and $v = 0$. When $act$ is some $X[i]$, then $mv$ will be the same $X[i]$ and $v$ will be the utility the agent receives that move. We call a sequence of such triples $h = \langle act_1, mv_1, po_1\rangle, \ldots, \langle act_k, mv_k, po_k\rangle$ a *history*.

**Example 2.** *The history for I2O in Example 1 is:*

$$\langle I, X[3], 8\rangle, \langle I, X[4], 5\rangle, \langle O, X[1], 3\rangle, \langle X[3], X[3], 8\rangle, \ldots$$

To concatenate a new triple onto the end of a history, we use the $\circ$ symbol: i.e. $h \circ \langle act, mv, po\rangle$ is the history $h$ concatenated with the triple $\langle act, mv, po\rangle$. Further, for $h = h_1 \circ \cdots \circ h_k$, we let $|h|$ be $k$ and $h_{1,\ldots,j}$ be the subhistory $h_1 \circ \cdots \circ h_j$. We denote the empty (initial) history by the symbol $h_\emptyset$.

The Cultaptation game uses 100 agents ($plrs = 100$), each with one of two strategies being compared against one another. On each round, each agent has a 2% chance of dying. As such, we also include a parameter $d$ in our formulation representing the probability of death. Upon death, an agent is removed from the game and replaced either through "reproduction" or "mutation," by a new agent whose strategy is chosen in the manner described below. Mutation happens 2% of the time, and reproduction happens 98% of the time. When reproduction occurs, the social learning strategy used by the newborn agent is chosen from the strategies of agents currently alive with a probability proportional to their average per-round utility (the utility gained by an agent divided by the number of rounds the agent has lived). The agent with the highest average per-round utility is thus the most likely to propagate its strategy on reproduction.

**Example 3.** *Again looking at the sequences of moves in Table 1, we see that both agents would have equal chance of reproducing on round 1. However, on round* $2$ *I1 has a per-round utility of* $1.5$*, while I2O has a per-round utility of* $0$*, meaning I1 gets 100% of the reproductions occurring on round* $2$*. Round three is the same, but on round* $4$ *I1 has a per round utility of* $2.25$ *and I2O has a per-round utility of* $2$*. This means that I1 gets* $100 \cdot 2.25/4.25 = 53\%$ *of the reproductions and I2O gets* $100 \cdot 2/4.25 = 47\%$ *of the reproductions on round* $4$*.*

When a "mutation" occurs, however, the new agent's strategy is chosen at random from the strategies currently playing. For instance, if there were a cultaptation game pitting strategies I1 and I2O against one another, then a new mutated agent would be equally likely to have either strategy I1 or I2O, even if there were no living agents with strategy I1. "Mutation" in this game refers only to the ability of a new agent to have one of several pre-specified strategies – it does not allow for changes to an agent's codebase such as might be expected from the use of the word "mutation." Through mutation, new strategies can be introduced into otherwise homogeneous populations. However, it is through reproduction that agents have the chance to spread their strategy. It is through reproduction that a social learning strategy can win the game.

If we have two social learning strategies $S_a$ and $S_b$, then at any round in the social learning game, there will be a number $n_a$ of agents using strategy $S_a$, and a number $n_b$ of agents

| Round # | 1 | 2 | 3 | 4 | 5 | ... | $k$ |
|---|---|---|---|---|---|---|---|
| I1's move | $Inv$ | $X[1]$ | $X[1]$ | $X[1]$ | $X[1]$ | ... | $X[1]$ |
| I1's util | 0 | 3 | 6 | 9 | 12 | ... | $3(k-1)$ |
| Per round | 0 | 1.5 | 2 | 2.25 | 2.4 | ... | $3(k-1)/k$ |
| I2O's move | $Inv$ | $Inv$ | $Obs$ | $X[3]$ | $X[3]$ | ... | $X[3]$ |
| I2O's util | 0 | 0 | 0 | 8 | 16 | ... | $8(k-3)$ |
| Per round | 0 | 0 | 0 | 2 | 3.2 | ... | $8(k-3)/k$ |

Table 1: Move sequences from Example 1, and their utilities.

using strategy $S_b$. A strategy $S_a$ wins the Cultaptation social learning game if after 7,500 rounds of play, the average value of $n_a$ *on the next 2,500 rounds* is larger than the average value of $n_b$ *on the next 2,500 rounds*.

For the purposes of this paper, we concern ourselves with the asymptotic values for $n_a$ and $n_b$: that is, as the round number approaches infinity, which is larger? If $n_a$, then $S_a$ is the better strategy. If $n_b$, then $S_b$ is the better strategy.

The only way an agent may affect $n_a$ or $n_b$ is through reproduction. We will show in Section  that any strategy maximizing an agent's expected per-round utility will also maximize its reproduction. We will therefore focus on computing the expected per-round utility.

## Formal Model

In this section we introduce a formal mathematical model of the game, culminating with a formal definition of the problem we are solving. Figure 1 is a glossary of the notation used in this paper.

We use $r$ for the round number and $E(h)$ to specify the number of exploit moves available after history $h$. Notice that after all exploit moves $X[1], \ldots, X[mvs]$ have been innovated or observed in a history $h$, then $E(h) = mvs$ and innovate moves become illegal.

We model the payoffs supplied for exploit moves $X[i]$ by a probability distribution $\pi$ parameterized by round. $\pi(v|i)$ is the probability of $v$ being set as the payoff for an exploit move on round $i$. If no moves change on round $i$, then $\pi(v|i) = 0$ for all $v$. If we let $\pi_{Inv}(v|i)$ be the probability that value $v$ is innovated on round $i$, it can be defined recursively in terms of $c$ and $\pi$ as:

$$\pi_{Inv}(v|i) = \begin{cases} \pi(v|0), & \text{if } i\text{=}0, \\ c(i) \cdot \pi(v|i) + & \\ \quad (1 - c(i)) \cdot \pi_{Inv}(v|i-1), & \text{otherwise.} \end{cases}$$

That is, initially the chance of innovating a value $v$ is the same as that value being chosen for an exploit move. On later rounds ($i > 0$) the chance of innovating $v$ is the chance that a move's value changed to $v$ on the current round, plus the chance that a move's value was $v$ on the previous round and it did not change this round.

We assume a provided distribution $\pi_{Obs}$ that gives the probability of value $v$ being observed by an observe move, and we allow $\pi_{Obs}$ to dependent upon the current history. For instance $\pi_{Obs}(v|h)$ will be the probability of value $v$ being observed after history $h$.

Finally, we set $V$ to all move values that may occur with non-zero probability:

$$V = \{v | \exists i, \pi_{Inv}(v|i) > 0 \vee \exists h, \pi_{Obs}(v|h) > 0\}.$$

We require the set $V$ to have finite cardinality.

**Transition Probabilities.** A transition probability function $P(h, h'|a)$ defines the probability of transitioning from history $h$ to history $h'$ in the next round if the agent performs action $a$. This function can be defined for any state transition, given what we know about the moves in the game.

If the action $a$ is an innovate, then

$$P(h, h \circ \langle Inv, m, v\rangle | Inv) = \tag{1}$$

$$\begin{cases} \pi_{Inv}(v|r)/(mvs - E(h)) & \text{If } \langle -, m, -\rangle \text{ is not in } h, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Recall that an agent cannot innovate move $m$ if it is already in the repertoire. Observation moves are not subject to the same restriction, so if $a$ is an observe, then

$$P(h, h \circ \langle Obs, m, v\rangle | Obs) = \pi_{Obs}(v|h)/mvs \tag{3}$$

Finally, if $a$ is an exploit, and $v$ is the last known value for move $X[m]$ such that $\langle -, m, v\rangle$ is the most recent tuple of form $\langle -, m, -\rangle$ in $h$, then setting $last$ to the round number of that tuple, we have:

$$P(h, h \circ \langle X[m], m, v\rangle | X[m]) = \prod_{k=last}^{|h|} (1 - c(k)) + \sum_{k=last}^{|h|} c(k)\pi(v, k)[\prod_{i=k}^{|h|}(1 - c(i))]$$

That is, the probability of it not changing at all, plus the probability that it most recently changed to $v$ and has not changed since. If $v$ is not the last known value for $X[m]$, then

$$P(h, h \circ \langle X[m], m, v\rangle | X[m]) = \sum_{k=last}^{|h|} c(k)\pi(v, k)[\prod_{i=k}^{|h|}(1 - c(i))]$$

which is similar, but assumes that the value must have changed at least once.

$P(h, h'|a)$ will give a probability of zero if the transition from $h$ to $h'$ is nonsensical, such as when $h'$ is not one move longer than $h$, or when the extra move in $h'$ is not $a$, or when $h$ and $h'$ do not agree on all information in $h$.

| | |
|---|---|
| $plrs$ | Number of agents in the environment. |
| $r$ | Number of the round, ranging from 0 to $\infty$. |
| $c(r)$ | The probability of change on round $r$. |
| $d$ | The probability of death on each round. |
| $h$ | A history of an agent, $h = h_1, \ldots, h_k$. |
| $\langle act, mv, po \rangle$ | The action, move, and utility of a move in the history. |
| $E(h)$ | Number of exploitable moves given history $h$. |
| $mvs$ | Number of exploit moves in the game. |
| $\pi(r)$ | The probability distribution over generated move values on round $r$. |
| $\pi_{Obs}(v\|h)$ | Prob. of observing value $v$ with history $h$. |
| $\pi_{Inv}(v\|r)$ | Prob. of innovating value $v$ on round $r$. |
| $V$ | The set of potential utility values. |
| $P(h, h'\|act)$ | Probability of transitioning from history $h$ to history $h'$ with action $act$. |
| $L(h)$ | Probability of being alive with history $h$. |
| $T$ | Set of all tuples: $\langle act, mv, v \rangle$. |

Figure 1: A glossary of notation used in this paper.

**Utility Function:** A utility function $U(\langle act, mv, v \rangle)$ defines the utility gleaned on triple $\langle act, mv, v \rangle$.

$$U(\langle act, mv, v \rangle) = \begin{cases} v & \text{If } \exists i, act = X[i] \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

Notice that $U(\cdot)$ is only non-zero on exploit moves.

**Strategy Representation.** A strategy $S$ is defined as a function mapping each history $h \in H$ to the agent's next action $S(h) \in \{Inv, Obs, X[1], \ldots, X[mvs]\}$. For instance, the strategy $\mathsf{I1}$ from Example 1 is defined by the function:

$$S_{\mathsf{I1}}(h) = \begin{cases} Inv & \text{If } h \text{ is empty.} \\ X[i] & \text{For } h = \langle Inv, X[i], v \rangle, \ldots \end{cases}$$

In this paper we will deal with *partially specified* strategies. A partially specified strategy is defined by a finite set $PS$ of history move pairs ($PS \subset H \times \{Inv, Obs, X[1], \ldots, X[mvs]\}$). The set $PS$ defines a strategy in the following way:

$$S^{PS}(h) = \begin{cases} m & \text{If } (h, m) \in PS, \\ rand(h) & \text{otherwise} \end{cases}$$

where $rand(h)$ is a randomly chosen move legal in $h$.

Partially specified strategies have the advantage of being guaranteed to be finitely representable.

**Expected Per-Round Utility.** The per-round utility (PRU) of a history $h = h_1 \circ \cdots \circ h_k$ is defined to be the sum of the utility acquired in that history divided by the history's length:

$$PRU(h) = (1/k) \sum_{\langle act, mv, v \rangle = h_i} U(\langle act, mv, v \rangle$$

The probability of a given history occurring depends on the strategy $S$ the associated agent uses. For $h = h_1, \ldots, h_k$ it is the product of each $h_{i+1}$ following the sub-history $h_{1,\ldots,i}$, or:

$$P(h|S) = \prod_{i=1}^{k-1} P((h_{1,\ldots,i}), (h_{1,\ldots,i}) \circ h_{i+1} | S((h_{1,\ldots,i}))) \quad (5)$$

The probability of an agent living long enough to experience history $h$ depends on the probability of death. It is $L(h) = (1-d)^{|h|-1}$.

Finally, the expected per-round utility for a strategy $S$ is sum over all histories of the history's $PRU$, weighted by their probability of occurring.

$$EPU(S) = \sum_{h \in H} \underbrace{L(h)}_{\text{Prob. of living } |h| \text{ rounds}} \times \underbrace{P(h|S)}_{\text{Prob. } S \text{ causing } h} \times \underbrace{PRU(h)}_{\text{Per-round utility}}.$$

**Problem Specification.** The problem we focus on in this paper is: given distributions $\pi$ and $\pi_{Obs}$, as well as function $c(\cdot)$ describing the probability of change, probability of death $d \in [0, 1]$, and $\epsilon > 0$, find a strategy $S$ such that $EPU(S)$ is within $\epsilon$ of $\max_{S'} EPU(S')$.

We will show in Section  that the strategy maximizing $EPU(S)$ is the evolutionarily dominant strategy.

## Analysis of *EPU*

In this section we examine the expected per-round utility. First we present a method for computing an approximation to the expected per-round utility, then we present an argument that a strategy maximizing expected per-round utility will also maximize its population in the Cultaptation social learning game.

### Computation of *EPU*

We will now define a formula that can be used to compute $EPU$ exactly for a given strategy $S$. We will show that the

following formula, $EV_{exp}(r, v)$, computes the expected contribution to the $EPU$ of exploiting value $v$ on round $r$.

$$EV_{exp}(r, v) = v \sum_{k=r}^{\infty} \frac{1}{k}(1-d)^{k-1} \tag{6}$$

The existence of $EV_{exp}$ will be essential to deriving a finitely computable formulation of $EPU$. It can also be expressed as [2]

$$\begin{aligned} EV_{exp}(r, v) &= v \sum_{k=1}^{\infty} \frac{1}{k}(1-d)^{k-1} - v \sum_{k=1}^{r-1} \frac{1}{k}(1-d)^{k-1} \\ &= v \left( \frac{ln(d)}{d-1} - \sum_{i=1}^{r-1} \frac{1}{i}(1-d)^{i-1} \right) \end{aligned} \tag{7}$$

and is therefore computable.

We can now define the expected value of a strategy $S$ in terms of the average per-round payoff of an agent.

$$\begin{aligned} &EPU_{alt}(S, h) = \\ &\quad \sum_{t \in T} P(h, h \circ t | S(h)) \\ &\quad \times (EV_{exp}(|h \circ t|, U(t)) + EPU_{alt}(S, h \circ t)) \end{aligned}$$

Where $T$ is the set of all tuples of the form $\langle act, mv, v \rangle$, and $h \circ t$ represents a possible history on the next round. Note that the size of $T$ is finite so long as the number of moves is finite.

**Proposition 1.** $EPU(S) = EPU_{alt}(S, h_\emptyset)$

*Proof.* First, we will show that $EPU_{alt}(S, h_\emptyset)$ equals the summation of $P(h|S)EV_{exp}(|h|, U(h_{|h|}))$ for all histories $h$. Then we will show that this equals the summation of $L(h)P(h|S)PRU(h)$ for all $h$.

From the definition in equation 5, note that $P(h_\emptyset \circ t|S) = P(h_\emptyset, h_\emptyset \circ t|S(h_\emptyset))$ for histories of length one. Thus, the base case simplifies to

$$\begin{aligned} EPU_{alt}(S, h_\emptyset) = &\sum_{t \in T} P(h_\emptyset \circ t|S) EV_{exp}(1, U(t)) \\ &+ \sum_{t \in T} P(h_\emptyset \circ t|S) EPU_{alt}(S, h_\emptyset \circ t) \end{aligned}$$

Similarly, $P(h|S)P(h, h \circ t|S(h))$ simplifies to just $P(h \circ t|S)$. With this in mind, the computation of $P(h|S)EPU_{alt}(S, h)$ in all cases can be rewritten as

$$\begin{aligned} &P(h|S)EPU_{alt}(S, h) = \\ &\quad \sum_{t \in T} P(h \circ t|S) EV_{exp}(|h \circ t|, U(t)) \\ &\quad + \sum_{t \in T} P(h \circ t|S) EPU_{alt}(S, h \circ t)) \end{aligned}$$

This is the sum of $P(h \circ t|S)EV_{exp}(|h \circ t|, U(t))$ for each possible history $h \circ t$, plus a recursive call to $P(h \circ t|S)EPU_{alt}(S, h \circ t)$ for each of the subsequent histories.

A proof is omitted, but clearly all histories $h$ are evaluated by depth $|h|$ in the recursion. This results in the summation of $P(h|S)EV_{exp}(|h|, U(h_{|h|}))$ for all histories $h$ of length $|h| \geq 1$.

The proof then proceeds arithmetically:

$$\begin{aligned} EPU_{alt}(S, h_\emptyset) &= \sum_{h \in H} P(h|S) EV_{exp}(|h|, U(h_{|h|})) \\ &= \sum_{h \in H} P(h|S) \sum_{k=|h|}^{\infty} \frac{L(k)U(h_{|h|})}{k} \\ &= \sum_{h \in H} \sum_{k=|h|}^{\infty} \frac{L(k)P(h|S)U(h_{|h|})}{k} \\ &= \sum_{k=1}^{\infty} \sum_{h \in H(\leq k)} \frac{L(k)P(h|S)U(h_{|h|})}{k} \\ &= \sum_{k=1}^{\infty} \sum_{h \in H(k)} L(k)P(h|S)PRU(h) \\ &= EPU(S) \end{aligned}$$

where $H(\leq k)$ is the set of all histories of length $\leq k$, and $H(k)$ is the set of all histories of length exactly $k$. □

Unfortunately, $EPU_{alt}$ is not finitely computable – because there is no end to the game, it suffers from an infinite recursion. To handle this, we introduce a depth-limited computation of $EPU_{alt}$, which only computes the portion of the total $EPU$ contributed by the first $i$ rounds.

$$EPU^{\delta}_{alt}(S, h) = \begin{cases} 0 & \text{If } \delta = 0 \\ \sum_{t \in T} P(h, h \circ t|S(h)) \times & \\ \quad (EV_{exp}(r, U(t)) + EPU^{\delta-1}_{alt}(S, h \circ t)) & \text{otherwise} \end{cases}$$

We prove in Section  that if the search depth $\delta$ is large enough, $EPU^{\delta}_{alt}(S, h)$ will be within $\epsilon$ of $EPU_{alt}(S.h)$.

## Optimal *EPU* leads to Optimal Play

This section provides a sketch of a proof that if a strategy has the optimal expected per-round utility, it will also have the optimal expected probability of reproducing.

Assume we have two strategies $S_a$ and $S_b$ playing the Cultaptation Game, and that there are $n_a$ agents using $S_a$ and $n_b$ agents using $S_b$, where $n_a + n_b = plrs$. To simplify our discussion, we assume the game to have been running for an infinite number of rounds, such that for any integer $i$, it is possible for an agent to have been alive $i$ rounds.

Let $TU(S, i)$ be the expected utility (not per-round utility, but total utility) obtained by an agent using $S$ in the first $i$ turns. For each agent with strategy $S_x$, we let $a_x \in A$ be an agent with that strategy where $A$ is the set of all agents and $a_x$ has lifetime $age(a_x)$. The expected fraction of repro-

[2] The simplification $\sum_{i=1}^{\infty} \frac{1}{i}(1-d)^{n-1} = \frac{ln(d)}{d-1}$ is due to (Childers 2008).

ductions achieved by all agents using strategy $S_x$ is:

$$PR(S_x) = \frac{\sum_{a_x \in A} \frac{TU(S_x, age(a_x))}{age(a_x)}}{\sum_{a_a \in A} \frac{TU(S_a, age(a_a))}{age(a_a)} + \sum_{a_b \in A} \frac{TU(S_b, age(a_b))}{age(a_b)}} \quad (8)$$

For strategy $S_x$, the expected value of the numerator in Equation 8 is a sum over all possible ages $j$ of the expected number of agents with an age $j$ (or $n_x \cdot (1-d)^{j-1} / \sum_{i=1}^{\infty}(1-d)^i = n_x \cdot (1-d)^{j-1} \cdot d$) times the per-round utility of $S_x$ after $j$ rounds (or $TU(S_x, j)/j$), giving:

$$N(S_x) = n_x d \sum_{j=1}^{\infty} (1-d)^{j-1} \frac{TU(S_x, j)}{j}$$

Note that

$$PR(S_a) = \frac{N(S_a)}{N(S_a) + N(S_b)} \quad (9)$$

Thus the probability of strategy $S_a$ reproducing on this round is proportional to $N(S_a)$. It turns out that $N(S_a)$ is directly related to $EPU(S_a)$. $EPU(S)$ is equal to the sum, for all $i$, of an agent's probability of being alive on round $i$ times the expected per-round utility of $S$ on round $i$. Hence, one way of expressing $EPU(S)$ is:

$$EPU(S) = \sum_{j=1}^{\infty} (1-d)^{j-1} \frac{U(S_a, j)}{j} \quad (10)$$

Hence,

$$PR(S_a) = \frac{n_a \cdot d \cdot EPU(S_a)}{n_a \cdot d \cdot EPU(S_a) + n_b \cdot d \cdot EPU(S_b)} \quad (11)$$

Let $q = EPU(S_b)/EPU(S_a)$. Note that

$$PR(S_a) = \frac{n_a \cdot d}{n_a \cdot d + n_b \cdot d \cdot q} = \frac{n_a}{n_a + q \cdot n_b} \quad (12)$$

Therefore, $EPU(S_a) > EPU(S_b)$ implies $q < 1$, hence $PR(S_a) > \frac{n_a}{n_a + n_b}$ if $n_b > 0$. Therefore, if $EPU(S_a) > EPU(S_b)$, we expect the fraction of agents using $S_a$ to grow at each reproductive step until it reaches 1. Finally, if we have a strategy $S_{opt}$ such that, for all $S$, $EPU(S_{opt}) > EPU(S)$, we expect the fraction of agents using $S_{opt}$ to approach 1 when $S_{opt}$ and $S$ play one another. Thus $S_{opt}$ is the optimal strategy from an evolutionary standpoint.

This goes to show that by optimizing $EPU$, one is, in fact, optimizing the expected fraction of agents eventually alive in a given social learning game. Since wins and losses are determined by the number of agents alive on game's end, the optimal evolutionary strategy optimizes $EPU$.

## Finding an $\epsilon$- Optimal Strategy

In many cases, such as in the example in Table 1, agents benefit in the long run from spending multiple rounds at the beginning of the game learning new actions. Strategies that only make locally optimal choices and begin exploiting as soon as they have learned their first action are clearly not optimal in these cases. Therefore, we are not likely to be able to find a good strategy by considering only the current round in the game; we will need to search ahead multiple rounds, to see where our agent's actions will put it later in the game.

We say a strategy $S$ is optimal if there is no strategy $S'$ such that $EPU(S') > EPU(S)$. We will now present a computable algorithm that, given an error bound $\epsilon$, finds a strategy $S$ such that, if $S_{opt}$ is the optimal strategy,

$$EPU(S_{opt}) - EPU(S) \leq \epsilon \quad (13)$$

The algorithm works by searching out some finite number of rounds $k$ and determining the best strategy for all histories of length $\leq k$. Because agents' probability of being alive on round $r$ decreases exponentially with $r$, a strategy's actions on earlier rounds are much more important in determining its $EPU$ than actions it performs on later rounds. We will use this fact to put a bound on the maximum contribution to $EPU(S)$ that can be made after some round $k$. Then, once the algorithm searches out to round $k$ such that the contribution to $EPU$ of all histories after that round is less than $\epsilon$, we can be sure that the strategy returned by the algorithm has computed an $EPU$ within $\epsilon$ of the optimal strategy $S_{opt}$ (a formula for $k$ is presented in section ).

**Algorithm 1** Produce strategy $S$ that maximizes $EPU^k_{alt}(S, h)$, given initial history $h$, and set of possible utility values $V$.

```
Strat(h,k,V)
  if k = 0 then return 0 endif
  Let U_max = 0, S_max = null, and a_max = null
  for each action a ∈ {I, O, X[1], .., X[mvs]} do
    Let U_temp = 0 and S_temp = null
    for each move m ∈ {1, .., mvs} do
      for each value v ∈ V do
        Let t = ⟨a, m, v⟩ and p = P(h, h ∘ t|a)
        if p > 0 then
          Let {S', U'} = Strat(h ∘ t, k − 1, V)
          S_temp = S_temp ∪ S'
          U_temp = U_temp + p·(EV_exp(|h∘t|, U(t)) + U')
        end if
      end for
    end for
    if U_temp > U_max then
      U_max = U_temp
      S_max = S_temp ∪ ⟨h, a_max⟩
    end if
  end for
  return {S_max, U_max}
```

## Algorithm

Algorithm 1 returns a 2-tuple with a partially specified strategy $S$ and the expected value for the strategy's average per-round utility up to round $k$. If $k$ is chosen to be large enough, then this algorithm will provide a strategy with expected EPU within $\epsilon$ of optimal.

The algorithm performs a depth-first search through the space of strategies that start from the input history $h$, stopping once it reaches a depth specified depth $k$. For each

possible action $a \in \{I, O, X[1], .., X[mvs]\}$ at $h$, it computes the expected per-round utility gained from performing $a$, and the utility of the best strategy for each possible history $h'$ that could result from choosing $a$. It combines these quantities to get the total expected utility for $a$, and selects the action with the best total expected utility, $a_{max}$. It returns the strategy created by combining the policy $\langle h, a_{max} \rangle$ with the strategies for each possible $h'$, and the utility for this strategy. Hence, $Strat(h_\emptyset, k, V)$ will return the $\epsilon$-optimal strategy for the entire game, provided $k$ is chosen large enough.

Seen another way, $Strat(h, k, V)$ computes $EPU^k_{alt}(S, h)$ for all possible strategies $S$, returning a (partially specified) strategy maximizing $EPU^k_{alg}$ as well as the maximal value of $EPU^k_{alg}$.

## Running Time Analysis

When Algorithm 1 considers a history $h$, it makes one recursive call for each possible combination of actions, moves, and values $\langle a, m, v \rangle$ that can be executed at $h$. The number of $\langle a, m \rangle$ pairs is at most $3(n_m)$, since each move $m \in \{1, 2, ..., n_m\}$ can be either Innovated, Observed, or Exploited. Each of these moves can also have any of $v$ values. Hence, the branching factor of the algorithm is at most $3(n_m v)$. Since the algorithm performs a depth-first search to depth $k$, and since it performs a constant amount of computation at each node in the search, its running time is $O((3n_m v)^k)$.

## Bounding *EPU*

In games where $0 < d < 1$, there is always a chance that the agent is still alive on any round $r = 1, \ldots, \infty$. In this case, considering all possible strategies is clearly not computable, since each strategy is potentially infinite in length. However, since the agent's probability of being alive on round $r$ decreases exponentially with $r$, the expected utility contributed by an agent's actions in later rounds is exponentially lower than the expected utility contributed by earlier rounds. We will use this fact in deriving a bound on $EPU_{alt}(S, h)$ for a given strategy and a history $h$ of length $k$.

Recall from Equations 6 and 7 that:

$$\begin{aligned} EV_{exp}(r, v) &= v \sum_{k=r}^{\infty} \frac{1}{k}(1-d)^{k-1} \\ &= v \left( \frac{ln(d)}{d-1} - \sum_{i=1}^{r-1} \frac{1}{i}(1-d)^{i-1} \right) \end{aligned} \quad (14)$$

where $EV_{exp}(r, v)$ is the expected contribution to $EPU$ made by exploiting a move with value $v$ on round $r$.

Since we know how much any given exploit contributes to the expected $EPU(S)$ for a given strategy $S$, we can calculate $G(k, v)$, the amount that exploiting the same action on *all* rounds after $k$ contributes to $EPU(S)$:

$$\begin{aligned} G(k, v) &= \sum_{j=k+1}^{\infty} \left( v \sum_{n=j}^{\infty} \frac{1}{n}(1-d)^{n-1} \right) \\ &= v \sum_{j=k+1}^{\infty} \sum_{n=j}^{\infty} \frac{1}{n}(1-d)^{n-1} \end{aligned}$$

Expanding the summations yields:

$$\begin{aligned} G(k, v) &= v \left( \frac{1}{k+1}(1-d)^k + \frac{2}{k+2}(1-d)^{k+1} + \cdots \right) \\ &= v \sum_{n=k+1}^{\infty} \frac{n-k}{n}(1-d)^{n-1} \\ &= v \left( \sum_{n=k+1}^{\infty} (1-d)^{n-1} - \sum_{n=k+1}^{\infty} \frac{k}{n}(1-d)^{n-1} \right) \\ &= v \left( \frac{(1-d)^k}{d} - \sum_{n=k+1}^{\infty} \frac{k}{n}(1-d)^{n-1} \right) \end{aligned}$$

Next, we pull $k$ out of the summation and use (14) to obtain:

$$G(k, v) = v\frac{(1-d)^k}{d} - \underbrace{kv \left( \frac{ln(d)}{d-1} - \sum_{n=1}^{k} \frac{1}{n}(1-d)^{n-1} \right)}_{a} \quad (15)$$

Note that for $0 < d < 1$, $G(k, v)$ is finite. The following theorem states that after round $k$, no strategy $S$ contributes more than $G(k, v_{max})$ to $EPU(S)$:

**Theorem 1.** *Let $v_{max}$ be the highest possible action utility in a game. Then for all $k$ and all strategies $S$, $APU_{alt}(S, h_\emptyset) - APU^k_{alt}(S, h_\emptyset) \leq G(k, v_{max})$*

*Proof.* Since it is not possible for any strategy to gain more utility than $v_{max}$ on any round, this follows from the discussion above. □

## Determining How Far to Search

To obtain an upper bound on $k$, we first note a bound on $G(k, v)$:

**Lemma 1.** $G(k, v) \leq v \cdot \frac{(1-d)^k}{d}$

This follows from noting that part (a) of Equation 15 is greater than or equal to zero, since it is equivalent to $EV_{exp}(k, k+1)$, which is always at least zero. Thus $G(k, v) = v(\frac{(1-d)^k}{d} - w) \leq v\frac{(1-d)^k}{d}$, since $w$ is always non-negative.

Now if we can find a $k'$ such that $\epsilon = v_{max}(1-d)^{k'}/d$, then we can be certain that $\epsilon \geq G(k', v)$. Solving for $k'$ in the above equation yields

$$k' = log_{(1-d)}(d\epsilon / v_{max}) \quad (16)$$

The fact that we can compute this $k'$ such that $G(k', v_{max}) \leq \epsilon$ implies the following corollary to Theorem 1.

**Corollary 1.** *For any $\epsilon > 0$, and any history $h$, there is a $k$ such that $|EPU^k_{alt}(S,h) - EPU_{alt}(S,h)| \leq \epsilon$.*

Combining this result with the fact that Algorithm 1 computes $EPU^k_{alt}(S, h_\emptyset)$, we have the following corollary, which states that for large enough $k$, Algorithm 1 computes a strategy whose $EPU$ is within $\epsilon$ of optimal.

**Corollary 2.** *For given $\epsilon > 0$ and set of move values $V$ with $v_{max} = \max(V)$, if $k' = \log_{(1-d)}(d \cdot \epsilon / v_{max})$ then $Strat(h_\emptyset, k', V)$ (Algorithm 1) returns $(S, U)$ such that for any strategy $S'$ achieving optimal $EPU$, $|EPU(S') - EPU(S)| \leq \epsilon$.*

## Example Computation

In this section, we detail some example results. We show, in particular, that locally optimal results are not necessarily globally optimal. Consider an agent in a social learning game defined by the following parameters:

- $c(i) = 0.4$ for all $i$;
- $d(i) = 0.5$ for all $i$;
- $\pi$ is a uniform distribution over $\{1, 2, 20, 30\}$;
- we have $\pi_{Obs}$ represent a set of extremely weak opponents, where $\pi_{Obs}(1|h) = 1$ for all $h$.

Now consider what the best move might be for an agent with the following two move history. On the first move, the agent innovates determining that move $X[1]$ has value 1. On the second move, the agent innovates determining that move $X[2]$ has value 2. At this point, the agent knows of moves $X[1]$ and $X[2]$ with values 1 and 2 respectively. One might think, therefore, that the agent should exploit the higher valued move ($X[2]$). However, it turns out that the optimal move, as computed by Algorithm 1, is to exploit $X[1]$. The approximated expected per-round utility of $X[2]$ is 2.25, while for $X[1]$ it is 2.38. We ran the algorithm with $\epsilon = 0.05$, so both of these utilities are within 0.05 of optimal, and we can guarantee that $X[1]$ is the best move. We understand this to be a result of the probability of change. Because moves are likely to change, and because $X[2]$'s current value is less than the mean of the distribution $\pi$, the algorithm is able assess more expected value in the probability that $X[1]$ changes value than in the probability of sticking with $X[2]$. Since $X[2]$ was discovered last turn, it has a smaller chance of having changed than $X[1]$ (probability 0.4 as opposed to probability $(1 - (1 - 0.4)^2) = 0.64$).

## Related Work

In (Carr et al. 2008), Carr et al. show how to compute optimal strategies for two highly simplified versions of the Cultaptation Project's social learning game. Their paper simplifies the game by completely removing the *observe* move—which prevents the agents from interacting with each other in any way whatsoever, thereby transforming the game into a single-agent game rather than a multi-agent game. Their model also assumes that exploitable moves cannot change value once they have been learned, which overlooks a key part of the full social learning game.

We know of no other work on the full-fledged social learning strategies game, although there are related problems in anthropology, biology and economics, where effective social learning strategies and their origins are of particular interest.

The social learning competition attempts to shed light on some open questions in behavioral and cultural evolution. Despite the obvious benefit of learning from someone else's work, several strong arguments have been made for why social learning isn't purely beneficial (Boyd and Richerson 1995; Rogers 1988). The answer to how best to learn in a social environment is non-trivial. Game theoretical approaches have been used to explore this subject, but there is still ongoing research in improving the models that are used (Henrich and McElreath 2003; Enquist, Ghirlanda, and Eriksson 2007).

In (Laland 2004), Laland discusses strategies for certain kinds of social learning in detail, and explores when it is appropriate to innovate or observe in a social learning situation. Indiscriminate observation is not always the best strategy, and there are indeed situations where innovation is appropriate. This is influenced by the conclusions of (Barnard and Sibly 1981), where Barnard and Sibly reveal that if a large portion of the population is learning only socially, and there are few information producers, then the utility of social learning goes down.

In (Nettle 2006), Nettle outlines the circumstances in which verbal communication is evolutionarily adaptive, and why few species have developed the ability to use language despite its apparent advantages. Nettle uses a significantly simpler model than the Cultaptation game, but provides insight that may be useful to understanding social learning in general. In Nettle's model, the population reaches an equilibrium at a point that where both individual and social learning occur. The point of equilibrium is affected by the quality of observed information and the rate of change of the environment.

In (Galef Jr. 1995), Galef differentiates social learning in animals from mere imitation. The paper develops a model that associates rewards or punishments with expressed behavior, which may be how animals avoid strictly repeating what they have observed. Galef further states that in highly variable environments, socially learned information may not always be the most beneficial, yet animals that learn socially are still able to learn locally adaptive behavior.

Other work on similar games include (Giraldeau, Valone, and Templeton 2002), where Giraldeau et al. outline reasons why social information can become unreliable. Both biological factors, and the limitations of observation, can significantly degrade the quality of information learned socially. In (Schlag 1998), Schlag develops rules that can be applied in a similar social learning environment that will increase the overall expected payoff of a population by restricting how and when agents act on information learned through observation.

## Conclusion

We have developed an algorithm for synthesizing near-optimal strategies in the social learning game. To decide what move a strategy $S$ should make at each point in the game, our algorithm does a lookahead search to estimate each move's expected utility. The accuracy of

this estimate relies on the fact that since the agent has a nonzero probability of death at each round, moves farther into the future have diminishing effects on the expected utility. The algorithm looks far enough ahead that that all further moves will change the expected utility by less than $\epsilon$. We have proved that this occurs within a lookahead depth of $\log_{(1-d)}(d\epsilon/v_{max})$, where $d$ is the probability of dying on each round and $v_{max}$ is the value of the maximal-utility move.

One limitation of our work is the algorithm's exponential running time—but we are confident that pruning techniques and approximation techniques can be developed to make the algorithm run much more quickly. Once the algorithm has been speeded up, this should make it possible to use the algorithm to analyze different parameter settings for the social learning game, to see which kinds of moves are optimal under what kinds of conditions. When is it, for instance, that innovation is always preferable to observations and vice-versa? Such investigations are left for future work.

Also left for future work is the examination of information gathering in the social learning game. One of our algorithm's inputs is the probability distributions from which the action utilities are drawn. We have kept the algorithm fully general by allowing these distributions to change from one time step to the next—but what the distributions are, and how/whether they change at each time step, is information that the game's authors have deliberately not revealed. Without access to such information, a game agent must either approximate the distributions or develop an algorithm that can do well without them. If we choose to approximate, should our agent be willing to sacrifice some utility early on, in order to gain information that will improve its approximation? Are there strategies that perform well in a wide variety of environments, that we could use until our agent develops a good approximation? Are some of these strategies so versatile that we can simply use them without needing to know the distributions? These remain open questions.

## Acknowledgements

This work supported in part by AFOSR grant FA95500610405, NAVAIR contract N6133906C0149, DARPA's Transfer Learning and Integrated Learning programs, and NSF grant IIS0412812. The opinions in this paper are those of the authors and necessarily those of the funders.

## References

Barnard, C., and Sibly, R. M. 1981. Producers and scroungers: A general model and its application to captive flocks of house sparrows. *Animal Behavior* 29:543–550.

Boyd, R., and Richerson, P. 1995. Why does culture increase human adaptability? *Ethology and Sociobiology* 16(2):125–143.

Boyd, R.; Enquist, M.; Eriksson, K.; Feldman, M.; and Laland, K. 2008. Cultaptation: Social learning tournament. `http://www.intercult.su.se/cultaptation`.

Carr, R.; Raboin, E.; Parker, A.; and Nau, D. 2008. When innovation matters: An analysis of innovation in a social learning game. In *Second International Conference on Computational Cultural Dynamics (ICCCD)*.

Childers, M. 2008. personal communication.

Enquist, M.; Ghirlanda, S.; and Eriksson, K. 2007. Critical social learning: A solution to rogers's paradox of nonadaptive culture. *American Anthropologist* 109(4):727–734.

Galef Jr., B. G. 1995. Why behaviour patterns that animals learn socially are locally adaptive. *Animal Behavior* 49:1325–1334.

Giraldeau, L. A.; Valone, T. J.; and Templeton, J. J. 2002. Potential disadvantages of using socially acquired information. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences* 357(1427):1559–1566.

Henrich, J., and McElreath, R. 2003. The evolution of cultural evolution. *Evolutionary Anthropology* 12:123–135.

Laland, K. 2004. Social learning strategies. *Learning and Behavior* 32:4–14.

Nettle, D. 2006. Language: Costs and benefits of a specialised system for social information transmission. In Wells, J., and et al., eds., *Social Information Transmission and Human Biology*. London: Taylor and Francis. 137–152.

Rogers, A. R. 1988. Does biology constrain culture? *American Anthropologist* 90(4):819–831.

Schlag, K. 1998. Why imitate, and if so, how?, : A boundedly rational approach to multi-armed bandits. *Journal of Economic Theory* 78:130–156.