Where Do Rewards Come From?

Satinder Singh
baveja@umich.edu
Computer Science & Engineering
University of Michigan, Ann Arbor

Richard L. Lewis
rickl@umich.edu
Department of Psychology
University of Michigan, Ann Arbor

Andrew G. Barto
barto@cs.umass.edu
Department of Computer Science
University of Massachusetts, Amherst

Abstract

Reinforcement learning has achieved broad and successful application in cognitive science in part because of its general formulation of the adaptive control problem as the maximization of a scalar reward function. The computational reinforcement learning framework is motivated by correspondences to animal reward processes, but it leaves the source and nature of the rewards unspecified. This paper advances a general computational framework for reward that places it in an evolutionary context, formulating a notion of an *optimal reward function* given a fitness function and some distribution of environments. Novel results from computational experiments show how traditional notions of extrinsically and intrinsically motivated behaviors may emerge from such optimal reward functions. In the experiments these rewards are discovered through automated search rather than crafted by hand. The precise form of the optimal reward functions need not bear a direct relationship to the fitness function, but may nonetheless confer significant advantages over rewards based only on fitness.

Introduction

In the computational reinforcement learning (RL) framework (Sutton & Barto, 1998), rewards—more specifically, reward functions—determine the problem the learning agent is trying to solve. Properties of the reward function influence how easy or hard the problem is, and how well an agent may do, but RL theory and algorithms are completely insensitive to the source of rewards (except requiring that their magnitude be bounded). This is a strength of the framework because of the generality it confers, capable of encompassing both homeostatic theories of motivation in which rewards are defined as drive reduction, as has been done in many motivational systems for artificial agents (Savage, 2000), and non-homeostatic theories that can account, for example, for the behavioral effects of electrical brain stimulation and addictive drugs. But it is also a weakness because it defers key questions about the nature of reward functions.

Motivating the RL framework are the following correspondences to animal reward processes. Rewards in an RL system correspond to *primary rewards*, i.e., rewards that in animals have been hard-wired by the evolutionary process due to their relevance to reproductive success. In RL, they are thought of as the output of a "critic" that evaluates the RL agent's behavior. Further, RL systems that form value functions, using, for example, Temporal Difference (TD) algorithms, effectively create *conditioned* or *secondary* reward processes whereby predictors of primary rewards act as rewards themselves. The learned value function provides ongoing evaluations that are consistent with the more intermittent evaluations of the hard-wired critic. The result is that the local landscape of a value function gives direction to the system's preferred behavior: decisions are made to cause transitions to higher-valued states. A close parallel can be drawn between the gradient of a value function and *incentive motivation* (McClure, Daw, & Montague, 2003).

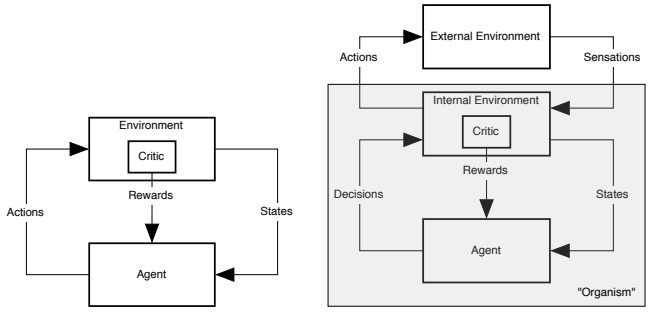

Figure 1: Agent-environment interactions in reinforcement learning; adapted from Barto et al. (2004). See text for discussion.

In the usual view of an RL agent interacting with an external environment (left panel of Figure 1), the primary reward comes from the external environment, being generated by a "critic" residing there. But as Sutton and Barto (1998) and Barto, Singh, and Chentanez (2004) point out, this is a seriously misleading view of RL if one wishes to relate this framework to animal reward systems.

In a less misleading view of this interaction (right panel of Figure 1), the RL agent's environment is divided into external and internal environments. For an animal, the internal environment consists of the systems that are internal to the animal while still being parts of the learning system's environment. This view makes it clear that reward signals are always generated within the animal, for example, by its dopamine system. Therefore, *all rewards are internal*, and the internal/external distinction is not a useful one, a point also emphasized by Oudeyer and Kaplan (2007). This is the viewpoint we adopt in this paper.

But what of a distinction between intrinsic and extrinsic reward? Psychologists distinguish between *extrinsic motivation*, which means doing something because of some specific rewarding outcome, and *intrinsic motivation*, which refers to "doing something because it is inherently interesting or enjoyable" (Ryan & Deci, 2000). According to this view, intrinsic motivation leads organisms to engage in exploration, play, and other behavior driven by curiosity in the absence of explicit reinforcement or reward.

Barto et al. (2004) used the term *intrinsic reward* to refer to rewards that produce analogs of intrinsic motivation in RL agents, and *extrinisic reward* to refer to rewards that define a specific task as in standard RL applications. We use this terminology here, but the distinction between intrinsic

and extrinsic reward is difficult to make precise. Oudeyer and Kaplan (2007) provide a thoughtful typology. Space does not permit providing more detail, except to point out that a wide body of data shows that intrinsically motivated behavior does not occur because it had previously been paired with the satisfaction of a primary biological need in the animal's own experience (Deci & Ryan, 1985). That is, intrinsic reward is not the same as secondary reward. It is likely that the evolutionary process gave exploration, play, discovery, etc., positive hedonic valence because these behaviors contributed to reproductive success throughout evolution. Consequently, we regard intrinsic rewards in the RL framework as primary rewards, hard-wired from the start of the agent's life. Like any other primary rewards in RL, they come to be predicted by the value system. These predictions can support secondary reinforcement so that predictors of intrinsically rewarding events can acquire rewarding qualities through learning just as predictors of extrinsically rewarding events can.

In short, once one takes the perspective that all rewards are internal (Figure 1), it is clear that the RL framework naturally encompasses and provides computational clarity to a wide range of reward types and processes, and thus has the potential to be a source of great power in explaining behavior across a range of domains. But fully realizing this scientific promise requires a computational framework for reward itself—a principled framework with generative power. Our main purpose here is to specify and illustrate a candidate for such a framework with the following desired properties:

Criteria for a Framework for Reward

1. The framework is *formally well-defined and computationally realizable*, providing clear answers to the questions of what makes a good reward and how one may be derived.
2. The framework makes *minimal changes to the existing RL framework*, thereby maintaining its generality.
3. The framework *abstracts away from specific mechanisms associated with RL agents*, such as whether their learning mechanisms are model-based or model-free, whether they use options or other kinds of richer internal structure, etc. But it is in principle powerful enough to exploit such agent structure when present.
4. The framework *does not commit to specific search processes* for finding good reward functions, but it does define and give structure to the search problem.
5. The framework *derives rewards that capture both intrinsic and extrinsic motivational processes.*

Taken together, these features of our framework distinguish it from other efforts aimed at deriving or specifying the form of reward functions, e.g., Schmidhuber (1991); Singh, Barto, and Chentanez (2005); Ng, Harada, and Russell (1999). While these computational approaches are all valuable explorations of reward formulations, they still incorporate some notion of pre-defined extrinsic reward, and are not concerned with explaining how their intrinsic rewards come about. Closer to our aims is early work by Ackley and Littman (1991) and recent work by Uchibe and Doya (2008). The former differs from our work in that it directly evolves secondary reward functions and lacks a theoretical framework. The latter proposes a specific mechanism—embodied evolution—for evolving primary reward but is still concerned with combining intrinsic and extrinsic rewards, depending on specialized RL algorithms for guaranteeing that the asymptotic policy does not differ from the one implied by the extrinsic reward. The framework we propose here shares the goal of providing an evolutionary basis, but dispenses with pre-defined extrinsic rewards and seeks maximum generality in its theoretical formulation.

Optimal Rewards

Adopting an evolutionary perspective leads naturally to an approach in which adaptive agents, and therefore their reward functions, are evaluated according to their *expected fitness* given an explicit fitness function and some distribution of environments of interest. The fitness function maps trajectories of agent-environment interactions to scalar fitness values, and may take any form (including functions that are similar in form to discounted sums of extrinsic rewards).

Definition More specifically we define the notion of optimal reward as follows. For a given RL agent A, there is a space, R_A, of reward functions that map an agent's state to a scalar primary reward that drives reinforcement learning. The composition of the state can depend on the agent architecture and its learning algorithm. There is a distribution over Markov decision process (MDP; Sutton and Barto (1998))[1] environments in some set $\mathcal{E}$ in which we want our agents to perform well (in expectation). A specific reward function $r_A \in R_A$ and a sampled environment $E \in \mathcal{E}$ produces h, the history of agent A adapting to environment E using the reward function r_A. A given fitness function F produces a scalar evaluation $F(h)$ for all such histories h. An optimal reward function $r_A^* \in R_A$ is the reward function that maximizes the expected fitness over the distribution of environments.

The formulation is very general because the constraints on A, R_A, F, and $\mathcal{E}$ are minimal. A is constrained only to be an agent that uses a reward function $r_A \in R_A$ to drive its search for behavior policies. F is constrained only to be a function that maps (finite or infinite) histories of agent-environment interactions to scalar fitness values. And $\mathcal{E}$ is constrained only to be a set of MDPs, though the Markov assumption can be easily relaxed. (We leave this to future work.)

Regularies within and across enviroments The above formulation essentially defines a search problem—the search for r_A^*. This search is for a primary reward function and is

[1] An MDP is a mathematical specification of agent-environment interaction in which the environment can be in one of a number of states, at each time step the agent executes an action from a set of available actions, which stochastically changes the state of the environment to a next state, and a scalar reward is delivered to the agent.

to be contrasted with the search problem faced by an agent during its lifetime, that of learning a good value function, and hence a good secondary reward function, specific to its environment. Thus, our concrete hypothesis is (1) the r_A^* derived from search will capture physical regularities across environments in $\mathcal{E}$ as well as complex interactions between $\mathcal{E}$ and specific structural properties of the agent A (note that the agent A is part of its environment and is constant across all environments in $\mathcal{E}$), and (2) the value functions learned by an agent during its lifetime will capture regularities present within its specific environment that are not necessarily shared across environments.

Two Sets of Computational Experiments

We now describe a set of computational experiments in which we directly specify A, F, and $\mathcal{E}$, and derive r_A^* via search. These experiments are designed to serve three purposes. First, they will provide concrete and transparent illustrations of the basic framework above. Second, they will demonstrate the *emergence* of interesting reward function properties that are not direct reflections of the fitness function—including features that might be intuitively recognizable as candidates for plausible intrinsic and extrinsic rewards in natural agents. Third, they will demonstrate the *emergence* of interesting reward functions that capture regularities across environments, and similarly demonstrate that value function learning by the agent captures regularities within single environments.

Basic form of each experiment

Both experiments use a simulated physical space shown by the 6×6 gridworld in Figure 3 (the arrows in that figure are explained below). It consists of four subspaces (of size 3×3). There are four movement actions, **N**orth, **S**outh, **E**ast and **W**est, that if successful move the agent probabilistically in the direction implied, and if they fail leave the agent in place. The thick black lines in the figure represent barriers that the agent cannot cross, so that the agent has to navigate through gaps in the barriers to move to adjacent subspaces. The agent lives continually for its lifetime, i.e., the interaction is not divided into trials or episodes. Each experiment will introduce objects into the gridworld such as food, water, or boxes. The state includes the agent's location in the grid as well as other features relevant to each experiment. These features and other experiment-specific aspects (e.g., the fitness functions used) are described in the appropriate sections below.

Our agents use the ε-greedy Q-learning (Watkins, 1989) algorithm to learn during their lifetimes. This algorithm has three types of parameters: 1) Q_0, the initial Q-function (we use small values chosen uniformly randomly from the range $[-0.001, 0.001]$) that maps state-action pairs to their expected discounted sum of future rewards, 2) α, the learning rate, and 3) ε, the exploration parameter (at each time step the agent executes a random action with probability ε and the greedy action with respect to the current Q-function with probability $(1-\varepsilon)$). At time step t, the current state is denoted s_t, the current Q-function is denoted Q_t, the agent executes an action a_t, and the Q-learning update is as follows:
$Q_{t+1}(s_t, a_t) = (1-\alpha)Q_t(s_t, a_t) + \alpha[r_t + \gamma \max_b(Q_t(s_{t+1}, b)]$, where r_t is the reward specified by reward function r_A for the state s_t, and γ is a discount factor that makes immediate reward more valuable than later reward (we use $\gamma = 0.99$ throughout). It is well known that the form of Q-learning used above will converge asymptotically to the optimal Q-function and hence the optimal policy (Watkins, 1989). Thus, our agent uses its experience to continually adapt its action selection policy to improve the discounted sum of rewards, as specified by r_A, that it will obtain over its future (remaining in its lifetime). Note that the reward function is distinct from the fitness function F.

The psuedo-code below describes how we use simulation to estimate the *mean cumulative fitness* for a reward function r_A *given* a particular setting of learning parameters (α, ε).

```
set (α, ε)
for i = 1 to N do
    Sample an environment E_i from 𝓔.
    In A, intialize Q-function
    Generate a history h_i over lifetime T for A and E_i
    Compute fitness F(h_i)
end for
return average of {F(h_1), ..., F(h_N)}
```

In the experiments we report below, we estimate the mean cumulative fitness of r_A as the maximum estimate obtained (using the pseudo-code above) over a coarse discretization of the space of feasible (α, ε) pairs.

Finding good reward functions for a given fitness function thus amounts to a large search problem.[2]

Hungry-Thirsty Domain: Emergent Extrinsic Reward for Water

In this experiment, each sampled environment has two randomly-chosen special locations (from among the 4 corners and held fixed throughout the lifetime of the agent): one where there is always food available, and one where there is always water available. In addition to the movement actions, the agent has two special actions available: *eat*, which has no effect unless the agent is at the food location, where it causes the agent to consume food, and *drink*, which has no effect unless the the agent is at the water location, where it causes the agent to consume water. When the agent eats food, it becomes not-hungry for one time step, after which it become hungry again. When the agent drinks water, it becomes not-thirsty for a random period of time (when not-thirsty, it becomes thirsty with probability 0.1 at each successive time step). Each time step the agent is not-hungry, its fitness is incremented by one. There is *no* fitness directly associated with

[2] In our experiments we conducted exhaustive searches over a discretized parameter space because our focus is on demonstrating the generality of our framework and on the nature of the reward functions found. However, there is structure to the space of reward functions (as we illustrate through our experiments) that we will exploit in future work to gain computational efficiency.

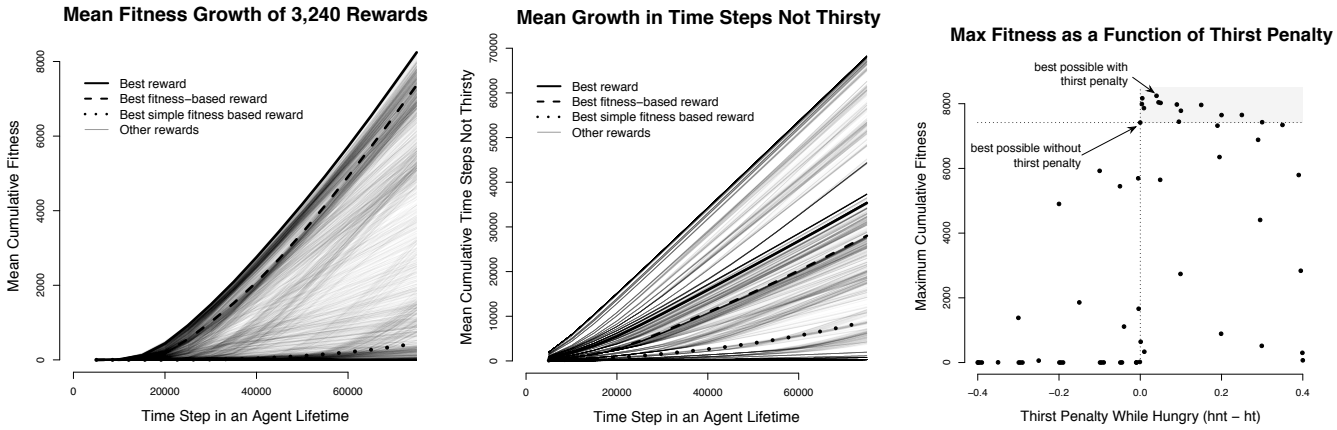

Figure 2: Results from Hungry-Thirsty Domain. See text for an explanation.

water at all. However being thirsty has a special effect: when the agent is thirsty, its eat action fails. Thus, the agent cannot just "hang out" at the food location and keep eating because at some point it will become thirsty and eating will fail. What is constant across environments is that there is food and there is water, that not-thirsty switches to thirsty with probability 0.1 on each step, and that being thirsty makes the agent incapable of eating. What varies across environments is the location of food and water. The state for use in Q-learning was four dimensional: the x and y coordinates of the agent's location and the binary thirst and hunger status features. We used only the hunger and thirst status features to define the space of reward functions. More specifically, the four combinations of hunger and thirst mapped to values chosen from a discretization of the range $[-1.0, 1.0]$; there were 3,240 different such reward functions considered; we provide some examples below.

Figure 2 presents some of our results. In this experiment as well as in the next, we distinguish between three increasingly-general classes of reward functions: 1) *simple fitness-based* reward functions that can assign positive reward to events that increment fitness (in this case being not-hungry) and zero to everything else, 2) *fitness-based* reward functions that can choose two arbitrary reward values, one for events that increment fitness (in this case being not-hungry) and another for all other events, and 3) *other* reward functions that are unconstrained (except by their range). The first interesting result (seen in the left panel) is that many reward functions outperform the fitness-based reward functions throughout the lifetime (of 75,000 time steps) in terms of mean cumulative fitness.[3] The best simple fitness-based reward function does very poorly.[4] The best reward function in our search space assigns a reward of -0.05 to the agent being hungry and thirsty, a larger but still negative reward of -0.01 to the agent being hungry and not-thirsty, and positive rewards of 1.0 to being not-hungry and thirsty, and 0.5 to being not-hungry and not-thirsty. It is apparent that this reward function differentiates based on the thirst status of the agent.

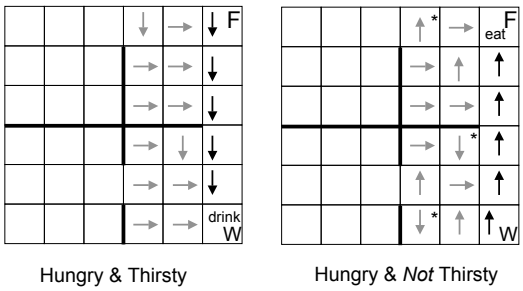

Figure 3: Policy for a single agent in Hungry-Thirsty Domain. See text for an explanation.

We illustrate this in the rightmost panel of Figure 2, which shows how fitness is sensitive to the magnitude of the *penalty* that the reward functions provide for being thirsty. We compute this penalty as the difference between the reward for (hungry, not-thirsty) and (hungry, thirsty). (We ignore the cases where the agent is not-hungry because they occur so infrequently). Each point in the graph plots the maximum mean cumulative fitness obtained over all reward functions with the same penalty. The vertical dotted axis separates positive penalties from negative penalties, and the horizontal dotted axis is the performance of the best fitness-based reward function, which by definition has a penalty of exactly zero. Noteworthy is that all the reward functions that outperform the best fitness-based reward function have a positive thirst penalty. Indeed, the performance is quite sensitive to the thirst penalty and peaks at a value of 0.04, i.e., not just any penalty for thirst will work well—only a relatively narrow and peaked region of thirst penalty outperforms the best reward functions that are insensitive to thirst. Another view of this result is seen in the middle panel of Figure 2 that plots the mean growth in the number of time steps the agent is not-thirsty as a function of time for the different reward functions. The best reward function is somewhere in the middle of the range of curves;

[3] Here and in all subsequent comparisons of expected fitness values between reward types, the comparisons are highly statistically significant at the $p < 10^{-5}$ level, using paired t tests appropriate to the structure of our computational experiments.

[4] Presumably because it cannot take advantage of the added exploration provided by an optimistic initialization of Q-functions because it cannot realize negative rewards.

clearly other reward functions can keep the agent not-thirsty far more often but do not achieve high cumulative fitness because they make being not-thirsty so rewarding that they prevent the agent from eating often enough.

Turning next to what happens within each lifetime or environment, the Q-learning performed by the agent produces a policy specific to the location of the food and water. This policy takes the agent back and forth between food and water, even though water does not directly contribute to cumulative fitness. This can be seen by inspecting Figure 3, which shows the policy learned by one of our agents (split into two panels: the right panel showing the actions when the agent is hungry and thirsty, and the left panel showing the policy when it is hungry and not-thirsty).[5]

Boxes Domain: Emergent Intrinsic Reward for Exploration and Manipulation

In this experiment, each sampled environment has two boxes placed in randomly chosen special locations (from among the 4 corners and held fixed throughout the lifetime of the agent). In addition to the usual movement actions, the agent has two special actions: *open*, which opens a box if it is closed and the agent is at the location of the box and has no effect otherwise (when a closed box is opened it transitions first to a half-open state for one time step and then to an open state), and *eat*, which has no effect unless the agent is at a box location, the box at that location is half-open, and there happens to be food (prey) in that box, in which case the agent consumes that food. A closed box always has food. The food always escapes when the box is open. Thus to consume food, the agent has to find a closed box, open it, and eat immediately in the next time step when the box is half-open. When the agent consumes food it feels not-hungry for one time step and its fitness is incremented by one. An open box closes with probability 0.1 at every time step. The state used for Q-learning was 6 dimensional: the x and y coordinates of the agent's location, the agent's hunger-status, the open/half-open/closed status of both boxes, as well the presence/absence of food in the square where the agent is located. We considered reward functions that map each possible combination of the status of the two boxes and hunger-status to values chosen from a discretization of the range $[-1.0, 1.0]$ (we searched over $54,000$ rewards in this space).

Unchanging across environments is the presence of two boxes and the rules governing food. Changing across environments—but held fixed within a single environment—are the locations of the boxes.

In a different design from the first experiment, we ran this experiment under two conditions. In the first, called the *constant condition*, the food appears in closed boxes throughout each agent's lifetime of $10,000$ steps. In the second, called the *step condition*, each agent's lifetime is $20,000$ steps, and food appears only in the *second half* of the agent's lifetime. Thus in the step condition, it is impossible to increase fitness above zero until after the $10,000^{th}$ time step. The step condition simulates (in extreme form) a developmental process in which the agent is allowed to play in its environment for a period of time in the absence of any fitness-inducing events. (In this case, the fitness-inducing events are positive, but in general there could also be negative ones that risk physical harm.) Thus, a reward function that confers advantage through exposure to this first phase must reward events that have only a very distal relationship to fitness. Through the agent's learning processes, these rewards give rise to the agent's intrinsic motivation. Notice that this should happen in both the step and constant conditions; we simply expect it to be more striking in the step condition.

The left and middle panels of Figure 4 shows the mean cumulative fitness as a function of time under the two conditions. As expected, in the step condition, fitness remains zero under any reward function for the first $10,000$ steps. The best reward function for the step condition is as follows: being not-hungry has a positive reward of 0.5 when both boxes are open and 0.3 when one box is open, being hungry with one box half-open has a small negative reward of -0.01, and otherwise being hungry has a reward of -0.05. (Note that the agent will spend most of its time in this last situation.) Clearly, the best reward function in our reward space rewards opening boxes (by making their half-open state rewarding relative to other states when the agent is hungry). This makes the agent learn to open boxes during the first half of the step condition so that when food appears in the second half, the agent is immediately ready to exploit that situation. This is reflected in the nearly constant slope from step $10,000$ onwards of the mean cumulative fitness curve of the best reward function. In contrast, the curve for the best fitness-based reward function has an increasing slope because the agent has to learn from step $10,000$ onwards that opening boxes leads to food. The policy learned under the best reward function makes the agent run back and forth between the two boxes, eating from both boxes, because this leads to higher fitness than staying at, and taking food from, only one box. This can be seen indirectly in the rightmost panel where the mean number of times both boxes are open is plotted as a function of time. It is clear that an agent learning with the overall best reward function keeps both boxes open far more often than one learning from the best fitness-based reward.

Discussion and Conclusions

We have outlined a general computational framework for reward that complements existing RL theory by placing it in an evolutionary context. This context clarifies and makes computationally precise the role of evolved reward functions: they convert distal pressures on fitness into proximal pressures on immediate behavior. We presented several computational ex-

[5] We only show the policy in the two subspaces containing the food and water because after learning the agent basically moves up and down the corridor connecting food with water and only departs it due to random exploration. Thus the agent gets very little experience in the other subspaces, and its policy there is mostly random. The policy off this corridor in the two right subspaces is mostly correct (the exceptions are the three locations marked with stars).

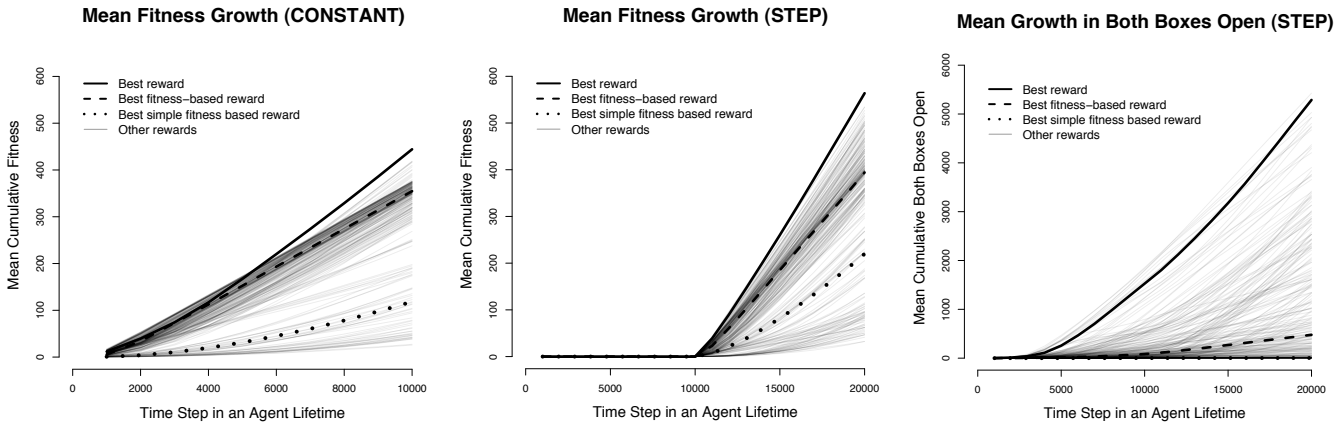

Figure 4: Results from Boxes Domain. See text for an explanation.

periments that serve to draw out and provide empirical support for two key properties of the framework.

First, multiple aspects of the domain may emerge to influence a single reward function. The combination of these multiple aspects is *implicit* in the form of the optimal reward function. Its precise properties are determined by the global goal of producing high fitness, but the relationship between the optimal reward function and fitness may be quite indirect. In the Hungry-Thirsty domain, two aspects of reward emerged, one related to food and hunger (directly related to fitness), and one related to thirst and water (not directly related to fitness). Both aspects were combined in a single function that represented a delicate balance between the two (Figure 2). In the Boxes domain, the optimal reward function related food and hunger (directly related to fitness), and curiosity and manipulation of boxes (not directly related to fitness). The latter aspect of the optimal reward produced distinct *play and exploratory* behavior that would be thought of as intrinsically-motivated in the psychological sense. This was especially evident in the second condition of the Boxes experiment: during the first half of the agent's lifetime, no fitness-producing activities are possible, but intrinsically rewarding activities are pursued that have fitness payoff later.

The second key property of the framework is that two kinds of adaptation are at work: the local adaptation of the RL agent within a given environment, and the global adaptation of the reward function to both a population of environments and the structure of the agent itself. The two kinds of adaptation are apparent in both experiments. In the Hungry-Thirsty domain, each agent benefits from the regularity across environments that drinking water ultimately helps it to achieve fitness—a regularity captured in the optimal (primary) reward function. Each agent also learns the specific locations of water and food sources in a given environment and good navigation patterns between them—regularities captured in the value functions learned by RL (Figure 3). Similarly, in the Boxes domain, each agent benefits from the regularity across environments that food is to be found in boxes, but also learns how to navigate to the specific locations of boxes in a given environment.

Acknowledgements Satinder Singh and Andrew Barto were supported by AFOSR grant FA9550-08-1-0418. Richard Lewis was supported by ONR grant N000140310087. Any opinions, findings, conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of the sponsors.

References

Ackley, D. H., & Littman, M. (1991). Interactions between learning and evolution. *Artificial Life II, SFI Studies in the Sciences of Complexity*.

Barto, A. G., Singh, S., & Chentanez, N. (2004). Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of the international conference on developmental learning.*

Deci, E. L., & Ryan, R. M. (1985). *Intrinsic motivation and self-determination in human behavior*. N.Y.: Plenum Press.

McClure, S. M., Daw, N. D., & Montague, P. R. (2003). A computational substrate for incentive salience. *Trends in Neurosciences*, *26*, 423-428.

Ng, A., Harada, D., & Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the sixteenth international conference on machine learning.* Morgan Kaufmann.

Oudeyer, P.-Y., & Kaplan, F. (2007). What is intrinsic motivation? A typology of computational approaches. *Frontiers in Neurorobotics*.

Ryan, R. M., & Deci, E. L. (2000). Intrinsic and extrinsic motivations: Classic definitions and new directions. *Contemporary Educational Psychology*, *25*, 54-67.

Savage, T. (2000). Artificial motives: A review of motivation in artificial creatures. *Connection Science*, *12*, 211-277.

Schmidhuber, J. (1991). A possibility for implementing curiosity and boredom in model-building neural controllers. In *From animals to animats: Proceedings of the first international conference on simulation of adaptive behavior* (p. 222-227). Cambridge, MA: MIT Press.

Singh, S., Barto, A. G., & Chentanez, N. (2005). Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems 17: Proceedings of the 2004 conference.* Cambridge MA: MIT Press.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.

Uchibe, E., & Doya, K. (2008). Finding intrinsic rewards by embodied evolution and constrained reinforcement learning. *Neural Networks*, *21*(10), 1447-1455.

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. Unpublished doctoral dissertation, Cambridge University, England.

Wolfe, A. P., Tran, T., & Barto, A. G. (2008). *Evolving reward and initial value functions* (Tech. Rep.). Amherst, MA: University of Massachusetts, Department of Computer Science.