Proceedings of the International Symposium on AI-Inspired Biology, Jackie Chappell, Susannah Thorpe, Nick Hawes and Aaron Sloman (Eds.), at the AISB 2010 convention, 29 March – 1 April 2010, De Montfort University, Leicester, UK

Where Do Rewards Come From?[1]

Richard L. Lewis[2] and **Satinder Singh**[3] and **Andrew G. Barto**[4]

Abstract. Reinforcement learning (RL) 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 RL 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.

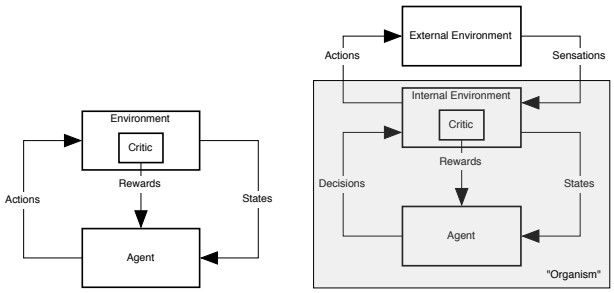

Figure 1. Agent-environment interactions in reinforcement learning; adapted from [1]. Left panel: Primary reward is supplied to the agent from its environment. Right panel: An elaboration in which the environment is factored into and internal and external environment, with all reward coming form the former.

1 INTRODUCTION

In the computational RL framework [10], 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 [7]), 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 computational 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 computational 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 *secondary* or *higher-order* reward processes whereby predictors of primary rewards act as rewards themselves. The learned value function provides immediate evaluations that are consistent with the infrequent 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* [3].

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 [10] and Barto, Singh, and Chentanez [1] 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 [5]). 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" [6]. According to this view, intrinsic motivation leads organisms to engage in exploration, play, and

[1] **This paper was originally published as Singh, S., Lewis, R. L. and Barto, A. (2009) "Where Do Rewards Come From?", *Proceedings of the Annual Conference of the Cognitive Science Society*, pp. 2601–2606.**

[2] Department of Psychology, University of Michigan, Ann Arbor, email: rickl@umich.edu

[3] Division of Computer Science & Engineering, University of Michigan, Ann Arbor, email: baveja@umich.edu

[4] Department of Computer Science, University of Massachusetts, Amherst, email: barto@cs.umass.edu

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. But the distinction between intrinsic and extrinsic reward is difficult to make precise. Oudeyer and Kaplan [5] 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 [2]. 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. 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 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:

1.1 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., [8, 9, 4]). While these computational approaches are all valuable explorations of reward formulations, they still incorporate some notion of external reward, and are not concerned with explaining how their internal rewards come about. Closer to our aims is recent work by Uchibe and Doya [11] that proposes a specific mechanism—embodied evolution—for evolving reward. But this work is also still concerned with combining internal and external rewards, depending on specialized RL algorithms for guaranteeing that the asymptotic policy does not differ from the one implied by the external reward. The framework we propose here shares the goal of providing an evolutionary basis, but dispenses with external rewards and seeks maximum generality in its formulation.

2 OPTIMAL REWARDS

Adopting an evolutionary perspective leads naturally to an approach in which adaptive agents, and therefore their internal 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 goodness values, and may take any form (including functions that are similar in form to discounted sums of external rewards).

2.1 Definition

More specifically we define the notion of optimal reward as follows. For a given RL agent A, there is a space of reward functions I_A that map agent internal state to a scalar primary reward that drives reinforcement learning. The nature of the internal state depends on the agent architecture. In general, A might include very specific commitments to particular learning algorithms and parameters so that the best I_A is partially determined by properties of the agent-as-learner. There is a distribution over Markov decision process (MDP; [10])[5] environments in some set $\mathcal{E}$ in which we want our agents to perform well (in expectation). A specific reward function $i_A \in I_A$ and a sampled environment $E \in \mathcal{E}$ produces h, the history of agent A adapting to environment E using the reward function i_A. A given fitness function G produces a scalar evaluation, $G(h)$ for all such histories h. The optimal reward function $i_A^* \in I_A$ is the reward function that maximizes the expected fitness over the distribution over environments.

The formulation is very general because the constraints on A, I_A, G, and $\mathcal{E}$ are minimal. A is constrained only to be an adaptive agent that uses a reward function $i_A \in I_A$ to drive its search for control policies. G is constrained only to be a function that maps (finite or infinite) histories of agent-environment interactions to scalar goodness 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).

2.2 Regularies within and across enviroments

The above formulation essentially defines a search problem—the search for i_A^*. This search is for a primary reward function and is to be contrasted with the search problem faced by an agent during its lifetime, that of learning a good secondary reward function specific to its environment. Thus, our concrete hypothesis is (1) the i_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 shared across environments.

[5] An MDP is a mathematical specification of agent-environment interaction in which the environment can be in one of a fixed number of states, at each time step the agent executes an action from a fixed set of actions, and the action executed stochastically changes the state of the environment according to fixed transition probabilities.

3 TWO SETS OF COMPUTATIONAL EXPERIMENTS

We now describe a set of computational experiments in which we directly specify A, G, and $\mathcal{E}$, and derive i_A^* via exhaustive 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 drives 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.

3.1 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 will be 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 keep 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 episodic. Each experiment will introduce objects into the gridworld such as food or water or boxes. The internal state of the agent includes its location in the grid as well as other features relevant to each experiment. These features as well as other experiment-specific aspects (e.g., the fitness functions used) will be described in the appropriate sections below.

Our agent uses the ε-greedy Q-learning [12] algorithm to learn how to act during its lifetime. This algorithm has three types of parameters: 1) Q_0, the initial Q-value function (we will use $Q_0 = 0.0$ throughout) that maps state-action pairs to their long-term utility, 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-value function with probability $(1-\varepsilon)$). At time step t, the current state is denoted s_t, the current Q-value function is denoted Q_t, the agent executes an action a_t, and the Q-learning update is as follows:

$$Q_{t+1}(s_t,a_t) = (1.0-\alpha)Q_t(s_t,a_t) + \alpha[r_t + \gamma \max_b (Q_t(s_{t+1},b)],$$

where r_t is the reward specified by the internal reward function i_A for the state s_{t+1}, 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-value function and hence the optimal policy [12]. Thus, our agent uses its experience to continually adapt its action selection policy to improve the amount of discounted sum of reward it obtains in the future (remaining in its lifetime). Note that the reward function is distinct from the fitness function G.

The psuedo-code below describes how the *mean cumulative fitness* for a reward function i_A and algorithm parameters (α,ε) is estimated using simulations.

for $i = 1$ **to** N **do**
  Sample an environment E_i from $\mathcal{E}$.
  In A, intialize Q-value function to zeros; set (α,ε)
  Generate a history h_i over lifetime T for A and E_i
  Compute fitness $G(h_i)$
end for
$G(i_A)$ = average of $\{G(h_1),\ldots,G(h_N)\}$

The fitness of a reward function i_A is thus the maximum fitness returned over all (α,ε) pairs[6].

3.2 Hungry-Thirsty Domain: Emergent Extrinsic Reward for Water

In this experiment, each sampled environment has two randomly chosen special locations (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 anywhere but when the agent is at the food location in which it causes the agent to consume food, and *drink*, which as no effect anywhere but at the water location in which it causes the agent to consume water. When the agent eats food it becomes not-hungry for one time step and then 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 time step). Each time step the agent is not hungry, its fitness is incremented by one. There is *no* fitness directly associated with 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 for at some point it will get 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 and that being thirsty makes the agent incapable of eating. What varies across environments is the location of food and water.

Figure 2 presents some of our results. The fitness-based rewards assign a positive reward to the events that increment fitness (in this case not being hungry) and zero to everything else. The first interesting result (seen in the left panel) is that there are many reward functions that outperform the fitness-based reward functions throughout the lifetime (of 80,000 time steps) in terms of mean cumulative fitness. Careful inspection of the left panel also shows that some reward functions outperform the fitness-based reward functions early on in the lifetime but then are worse later on in the agent's life. We expect this to be a general phenomenon (and will study this in more detail in future work). The best reward function in our search space assigns a penalty of -0.2 to the agent being hungry and thirsty, a smaller penalty of -0.1 to the agent being hungry but not thirsty, a small positive reward of 0.1 to being not hungry but thirsty, and a large reward of 1.0 to not being hungry and not being thirsty. It is apparent that this reward function differentiates based on the thirst status of the agent. Thus, the search procedure has encoded into the reward function the emergent drive to drink water when thirsty. The Q-learning performed by the agent during its lifetimes learns a policy specific to the location of the food and water and 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 that shows the policy learned by the agent (split into two panels; the right panel showing the actions when the agent is hungry

[6] Finding good reward functions for a given fitness function thus amounts to a huge search problem. In this paper, we conduct the search (mostly) exhaustively 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.

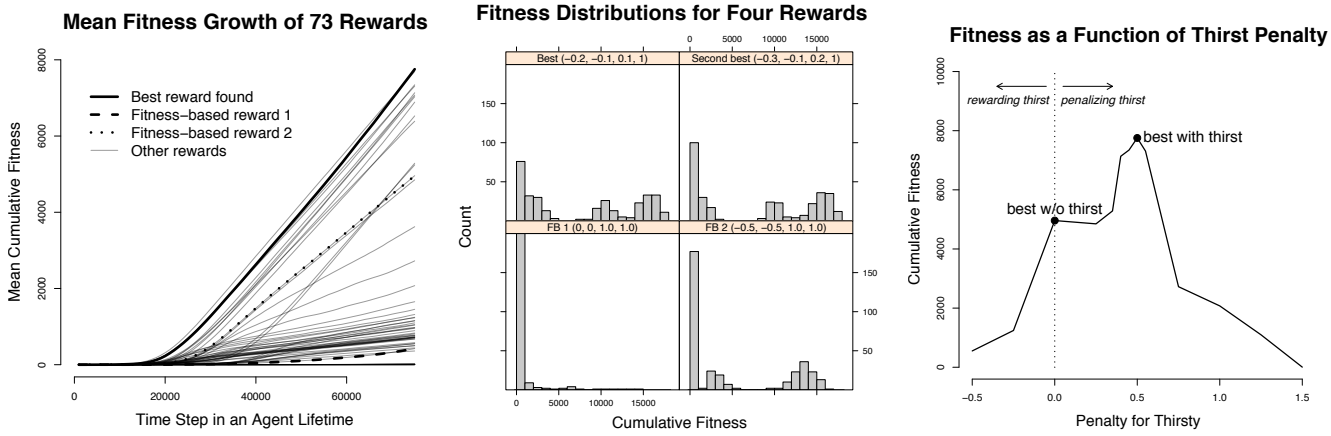

Figure 2. Results from Hungry-Thirsty Domain. See text for an explanation.

and thirsty and the left panel showing the policy when it is hungry and not thirsty)[7].

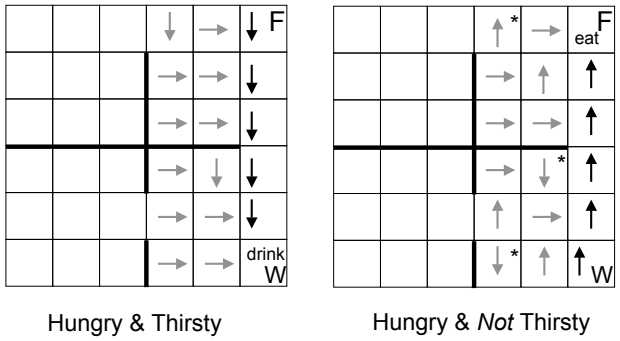

Figure 3. Policy for a single agent in Hungry-Thirsty Domain. See text for an explanation.

The middle panel in Figure 2 plots the distributions of cumulative fitness scores (at the end of the 80,000 time step lifetime) for 320 agents for each of four different reward functions (the two best rewards and the two fitness-based rewards). For all four rewards, the high variance and the multi-modal structure of the distributions is quite striking (and to us unexpected). The multimodal distribution of cumulative fitness also clearly shows that the fitness-based rewards do very poorly more than half the time.

Finally, the rightmost panel in Figure 2 shows how fitness is sensitive to the magnitude of the *penalty* that the reward functions provide for being thirsty[8]. Each point along the horizontal axis corresponds to a subspace of reward functions that assign a specific penalty to being thirsty. The zero point on the horizontal therefore represents all those rewards that do not discriminate based on thirst, and the point labeled "best w/o thirst" indicates the fitness of the best-performing reward in that subspace. This reward corresponds exactly to the reward labeled "Fitness-based Reward 2" in the other panels. The point at the peak labeled "best with thirst" corresponds exactly to the reward labeled "Best reward found" in the other panels. This panel provides just one simple way of viewing the *fitness surface* associated with the reward space. It clearly indicates that not just any penalty for thirst will work well—only a relatively narrow and sharply peaked region of thirst penalty outperforms the best rewards that are insensitive to thirst.

3.3 Boxes Domain: Emergent Intrinsic Reward for Exploration and Manipulation

In this experiment, each sampled environment has two boxes placed in randomly chosen special locations (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 the agent is at the location of the box and has no effect otherwise, and *eat*, which has no effect unless the agent is at a box location, the box at that location is open, and there happens to be food in that box, in which case the agent consumes that food. An open box has 0.1 probability of closing at every time step. A closed box always has food. An open box never has food except for one time step after being opened from a closed state. Thus to consume food, the agent has to find a closed box, open it, and eat immediately in the next time step. When the agent consumes food it feels not hungry for one time step and its fitness is incremented by one. In addition to the location, the open/closed status of boxes is available to the agent as an observation at all times. The fitness based reward functions assign a positive reward for the fitness-inducing event, i.e., for being not-hungry and

[7] We only show the policy in the two subspaces containing the food and water because after learning the agent basically walks up and down the corridor connecting food with water and only falls off 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 the path in the two right-subspaces is mostly correct (the exceptions are the three locations marked with a star).

[8] The thirst penalty for a given reward is simply the mean of the thirst penalty when hungry (the difference of the reward for hungry-but-not-thirsty and hungry-and-thirsty) and the thirst penalty when not hungry (the difference of the reward for not-hungry-and-not-thirsty and not-hungry-but-thirsty).

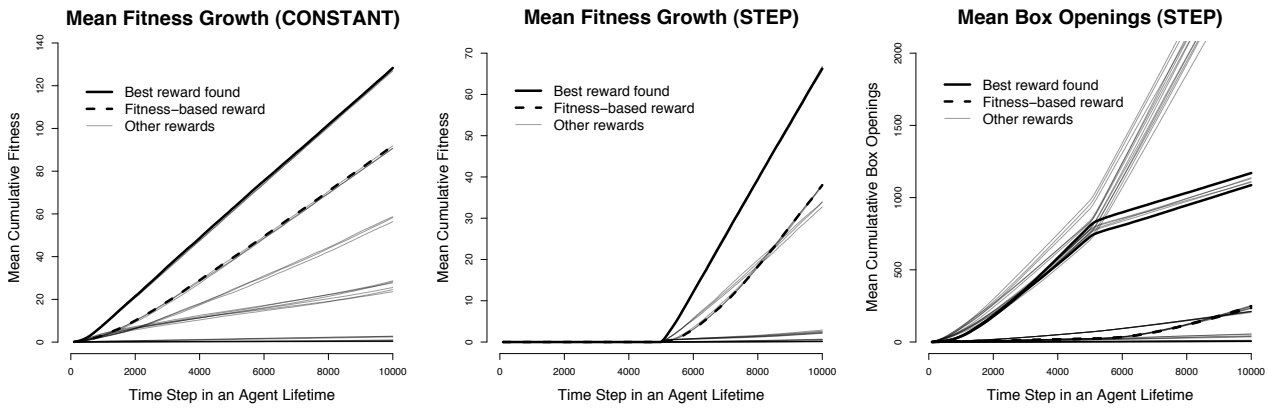

Figure 4. Results from Boxes Domain. See text for an explanation.

zero for everything else. What is unchanging across environments is the presence of two boxes and the dynamics of their open to close status. What changes across environments but held fixed within a single environment is the location of the boxes. Once again, this sets up the possibility for the search to hardwire across environment regularities into the reward function and then Q-learning to learn environment specific regularities.

In a different design from the first experiment, we ran this experiment under two different conditions. The first, called the *constant condition*, is one in which the food appears in closed boxes throughout the agent lifetime of $10,000$ steps. The second, called the *step condition*, is one in which for the first half of the agent's lifetime there is never any food in any boxes and then after $5,000$ time steps food always appears in a closed box. So in the step condition, there is no fitness to be obtained at all until after the $5,000^{th}$ 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 without having any fitness-inducing events present (in this case the fitness-events are good but in general there can also be bad ones that risk physical harm). Thus a reward function that is able to gain advantage through exposure to this first phase has to reward events that are intrinsically-motivating because they are by definition distal from any fitness-inducing events. Notice that this should happen in both the step and the constant conditions; it is just more stark and apparent in the step condition.

The top row of Figure 4 shows the average cumulative fitness as a function of time under the two conditions. As expected, in the step condition, there is no fitness under any reward function for the first $5,000$ time steps. The best reward function for the step condition is as follows: not being hungry has a reward of 1.0, being hungry but with both boxes open has a slight reward of 0.1, one box open and hungry has a 0.0 reward, and both boxes closed and hungry has a slight penalty of -0.1. Clearly, the best reward function in our reward space has chosen to reward opening boxes. This makes the agent learn to open boxes during the first half of the step condition so that when the food appears in the second half the agent is immediately ready to exploit that situation. This is reflected in the constant slope from time $5,000$ onwards of the average cumulative fitness curve of the best reward function. In contrast, the curve for the fitness-based reward function has a increasing slope because it has to learn to want to open boxes from point $5,000$ onwards. This difference is more clearly seen in the rightmost panel where we plot the average cumulative number of box openings for the different reward functions. The best reward function clearly opens the boxes far more often than the fitness-based reward function.

4 DISCUSSION AND CONCLUSIONS

We have outlined a general computational framework for reward that complements existing reinforcement learning theory by placing it in an evolutionary context—a context that clarifies and makes computationally precise the role of evolved reward functions in helping adaptive agents achieve fitness. We presented several computational experiments 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 was related to 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 life time, 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.

These are initial results in simple domains but they suggest several promising parallel avenues for future research. It is especially important to explore environments in which fitness selects for rewards that yield a diverse set of tasks and behaviors—environments in which the traditional notion of an external reward function provides the least guidance. It is also clear that the search for good reward functions represents a major computational challenge. Further work is required to develop more efficient approximate search algorithms that exploit the structure of this space, while still preserving its potential richness.

ACKNOWLEDGEMENTS

Satinder Singh was supported by AFOSR grant FA9550-08-1-0418 and by NSF grant IIS 0905146. Richard Lewis was supported by ONR grant N000140310087 and by NSF grant IIS 0905146. Andrew Barto was supported by AFOSR grant FA9550-08-1-0418. Any opinions, findings, conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of the sponsors.

REFERENCES

[1] A. G. Barto, S. Singh, and N. Chentanez, 'Intrinsically motivated learning of hierarchical collections of skills', in *Proceedings of the International Conference on Developmental Learning (ICDL)*, (2004).

[2] E. L. Deci and R. M. Ryan, *Intrinsic Motivation and Self-Determination in Human Behavior*, Plenum Press, N.Y., 1985.

[3] S. M. McClure, N. D. Daw, and P. R. Montague, 'A computational substrate for incentive salience', *Trends in Neurosciences*, **26**, 423–428, (2003).

[4] A. Ng, D. Harada, and S. Russell, 'Policy invariance under reward transformations: Theory and application to reward shaping', in *Proceedings of the Sixteenth International Conference on Machine Learning*, Morgan Kaufmann, (1999).

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

[6] R. M. Ryan and E. L. Deci, 'Intrinsic and extrinsic motivations: Classic definitions and new directions', *Contemporary Educational Psychology*, **25**, 54–67, (2000).

[7] T. Savage, 'Artificial motives: A review of motivation in artificial creatures', *Connection Science*, **12**, 211–277, (2000).

[8] J. Schmidhuber, '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*, pp. 222–227, Cambridge, MA, (1991). MIT Press.

[9] S. Singh, A. G. Barto, and N. Chentanez, 'Intrinsically motivated reinforcement learning', in *Advances in Neural Information Processing Systems 17: Proceedings of the 2004 Conference*, Cambridge MA, (2005). MIT Press.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, MA, 1998.

[11] E. Uchibe and K. Doya, 'Finding intrinsic rewards by embodied evolution and constrained reinforcement learning', *Neural Networks*, **21**(10), 1447–1455, (2008).

[12] C. J. C. H. Watkins, *Learning from Delayed Rewards*, Ph.D. dissertation, Cambridge University, Cambridge, England, 1989.

[13] A. P. Wolfe, T. Tran, and A. G. Barto, 'Evolving reward and initial value functions', Technical report, University of Massachusetts, Department of Computer Science, Amherst, MA, (2008).