Shaping Model-Free Habits with Model-Based Goals

Paul M. Krueger (pmk@berkeley.edu)
Thomas L. Griffiths (tom_griffiths@berkeley.edu)
Department of Psychology, University of California, Berkeley

Abstract

Model-free (MF) and model-based (MB) reinforcement learning (RL) have provided a successful framework for understanding both human behavior and neural data. These two systems are usually thought to compete for control of behavior. However, it has also been proposed that they can be integrated in a cooperative manner. For example, the Dyna algorithm uses MB replay of past experience to train the MF system, and has inspired research examining whether human learners do something similar. Here we introduce an approach that links MF and MB learning in a new way: via the reward function. Given a model of the learning environment, dynamic programming is used to iteratively approximate state values that monotonically converge to the state values under the optimal decision policy. Pseudorewards are calculated from these values and used to shape the reward function of a MF learner in a way that is guaranteed not to change the optimal policy. We show that this method offers computational advantages over Dyna in two classic problems. It also offers a new way to think about integrating MF and MB RL: that our knowledge of the world doesn't just provide a source of simulated experience for training our instincts, but that it shapes the rewards that those instincts latch onto. We discuss psychological phenomena that this theory could apply to, including moral emotions.

Introduction

You're at a dinner buffet and intend to choose a healthy salad to help achieve your dietary goal of losing weight and staying fit. Nonetheless, you are unable to resist taking a piece of pie. Eating the pie is pleasurable but afterwards you feel guilt. Why are our habits so often misaligned with our goals, and how might emotions like guilt mediate this misalignment? The interaction between habits and goals – how the former support or undermine the latter – is a critical and commonplace dilemma faced by people, and an important area of research in psychology and neuroscience (Aarts & Dijksterhuis, 2000; Dolan & Dayan, 2013). Here we present a reinforcement learning architecture that can be used to describe the interaction between a simple learning system (e.g. habits) and a higher-level, more sophisticated one (e.g. goals).

Dual-process theories – expressing human cognition as the result of two interacting systems, such as systems that produce habits vs. goals – explain a range of fundamental properties of human decision-making and judgment. Inspired by results in machine learning, recent research in psychology and neuroscience has explored how the brain might contain two systems that use different approaches to the problem of learning from environmental rewards, known as model-free (MF) and model-based (MB) reinforcement learning (RL) (Daw & Dayan, 2014; Daw, Niv, & Dayan, 2005; Otto, Gershman, Markman, & Daw, 2013). MF learning relies on direct trial-and-error interaction with the environment (Sutton, Barto, & Williams, 1992), while MB learning leverages knowledge about the causal structure of the environment (Barto, Bradtke, & Singh, 1995). MF learning offers a simple, computationally cheap approach to learning, while MB learning is more sophisticated and resource-intensive. In the domain of decision making, MF and MB learning have been used respectively to describe dual-processes including habits vs. goals, reflexive vs. reflective choice, retrospective vs. prospective decisions, and Pavlovian vs. instrumental learning (Boureau, Sokol-Hessner, & Daw, 2015; Dolan & Dayan, 2013).

Understanding the cognitive and neural relationship between MF and MB learning – and, by extension, between various dual-processes – remains an unresolved question. Historically, animal psychologists viewed these two approaches as distinct and competing hypotheses, with behaviorists arguing in favor of reflexive, MF learning based on stimulus-response associations (Thorndike, 1933), and Tolman and others positing an internal representation of the environment, or "cognitive map" (Tolman, 1948). Nowadays, while behavioral and neural data indicate that human learning relies on both systems (Daw et al., 2005; Dayan & Berridge, 2014; Gläscher, Daw, Dayan, & O'Doherty, 2010), it is typically assumed that they compete for control of behavior. However, it is also possible for them to cooperate. The Dyna architecture achieves such cooperation by integrating MF learning with MB planning (Sutton, 1991). In Dyna, as MF learning occurs, transitions between states of the environment and the resulting rewards are stored in a model. That model is used to replay these past experiences to further train MF state-action values. Recent behavioral data from people performing a retrospective revaluation task is consistent with a cooperative architecture like Dyna (Gershman, Markman, & Otto, 2014).

Here we introduce Model-Based Pseudoreward Approximation (MBPA), a method for cooperative interaction between MF and MB learning. The MB system generates pseudorewards that shape the reward function used in MF learning. According to the *shaping theorem* (Ng, Harada, & Russell, 1999), conditions exist under which the optimal decision policy will remain invariant to such modifications of the reward function, opening the possibility that pseudorewards can be used to guide agents toward optimal behavior. That is, since the optimal policy is guaranteed to remain unchanged, pseudorewards can potentially be used to guide the agent to the optimal policy. Using these principles, we show that pseudorewards can provide a link between MF and MB learning through modification of the reward function.

MBPA offers an appealing alternative to Dyna, both conceptually and practically. With Dyna, the MB replay of past experience suggests that planning (by internal simulation) is one way that different learning systems might be linked in hu-

man cognition. MBPA offers an alternative approach, based on changing the reward function, which can be tested experimentally in humans. In particular, this offers a new way to think about the relationship between dual-process theories that involve MF and MB learning, with pseudorewards providing the crucial link. In the case of eating sweets when one's goal is to be on a diet, the emotion of guilt serves as the (negative) pseudoreward, generated by the MB goal to retrain the MF habitual system. Remorse could serve a similar function when one behaves unethically.

We begin by reviewing the Dyna architecture for integrated MF and MB learning. We then introduce our method and its theoretical background. We present two simulations which show the effectiveness of our method and how it compares with Dyna. The first simulation involves learning in a maze environment, and the second simulation uses the classic mountain car problem. We end by discussing how this integrated approach may serve as a metacognitive solution to the rational use of cognitive resources (Griffiths, Lieder, & Goodman, 2015), and how it may shed light on the function of emotion in mediating the relationship between dual-processes.

Background

In this section we introduce the modeling framework for MF and MB reinforcement learning, and describe a popular algorithm that cooperatively integrates the two.

Markov Decision Processes

We describe sequential decision problems that can be modeled as a *Markov Decision Process (MDP)*. The MDP is defined as the 5-tuple: $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma\}$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, and for each $(s,a) \in \mathcal{S} \times \mathcal{A}$, $\mathcal{P}(s,a,s')$ is the probability of transitioning to state s' when action a is selected in state s, $\mathcal{R}(s,a,s')$ is the reward received for transitioning from state s to s', and γ^t is the discount factor for rewards t time steps in the future, where $0 \leq \gamma \leq 1$ (Sutton & Barto, 1998). A *policy*, π, is a mapping of states, $\mathcal{S}$, onto actions, $\mathcal{A}$: $\pi : \mathcal{S} \mapsto \mathcal{A}$. A *value function*, $V^\pi(s)$, is the expected amount of discounted reward generated by following policy π beginning at state s:

$$V^\pi(s) = \sum_{s'} P_{\pi(s)}(s,a,s')(R_{\pi(s)}(s,a,s') + \gamma V^\pi(s')). \quad (1)$$

An *optimal policy*, π^*, is a policy that maximizes the value function: $\pi^*(s) = \arg\max_a V^\pi(s)$.

Model-free Reinforcement Learning

RL is concerned with learning an effective policy from rewards alone. MF methods require no knowledge about the environment, and the agent learns which state-action pairs lead to reward through trial-and-error. One of the most common MF methods, which is employed throughout the simulations in this paper, is Q-learning (Sutton & Barto, 1998). When the agent takes action a from state s, leading to state s' and reward $R(s,a,s')$, a value $Q(s,a)$ is learned via the update

$$Q(s,a) \leftarrow Q(s,a) + \alpha(R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)), \quad (2)$$

where α is the *learning rate* that determines how quickly the agent learns from new experience. The terms $[R(s,a,s') + \gamma \max_{a'} Q(s',a') - Q(s,a)]$ are called the *temporal difference error*. Initially all $Q(s,a)$ are zero, and Q-learning can eventually learn an optimal policy π^* over time. The agent uses a *decision policy*, such as the ε-greedy policy which is used in our simulations. At each state s, with probability $1-\varepsilon$, the agent chooses the action $a \in \mathcal{A}$ with the highest value $Q(s,a)$. With probability ε it chooses an action uniformly at random (ε calibrates the explore-exploit tradeoff).

Model-based Reinforcement Learning

Unlike MF RL, MB RL has knowledge of the environment in terms of the transition probabilities between states, $\mathcal{P}$, and the reward contingencies for state-action pairs, $\mathcal{R}$. One of the most common MB methods for finding an optimal policy π^* is *dynamic programming* which calculates the value of state s under policy π according to the *Bellman equation*:

$$V^\pi(s) = R_{\pi(s)}(s,\pi(s),s') + \gamma \sum_{s'} P_{\pi(s)}(s,a,s')V^\pi(s')), \quad (3)$$

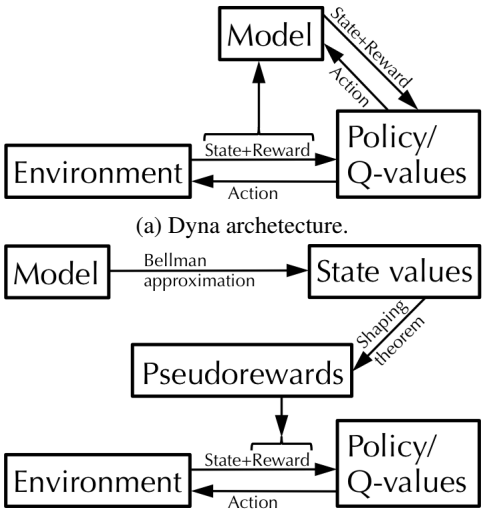
and finds the value of each state $V^*(s)$ under the optimal policy π^* by recursively updating these values using the *Bellman optimality equation*:

$$V^*(s) = \max_a R_{\pi(s)}(s,a,s') + \gamma \sum_{s'} P_{\pi(s)}(s,a,s')V^*(s')). \quad (4)$$

Dyna

Dyna uses MF learning combined with a MB system that replays past experiences, which are used to train the MF system (Figure 1a). After each real action taken in the environment, the model stores the state-action pair and reward received. It then randomly selects n past state-action pairs and replays them. These planned actions are used to update the MF system as if they were real actions. In Dyna-Q, the MF system uses one-step tabular Q-learning (which is what we use in our simulations). The number of simulated planning steps, n, is a parameter that can be set to any positive integer value. Dyna typically begins with no knowledge about the causal structure of the environment (that is, transitions between states and reward contingencies), but builds this knowledge based on experience. However, Dyna can also inherit a model of the environment, but this actually slows MF learning.

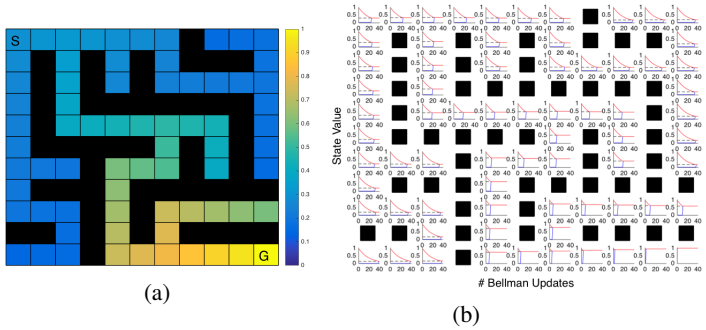
In addition to being a useful algorithm for integrating direct learning with indirect replay, Dyna has been proposed as a model of human cognition – behavioral experiments have found evidence in humans consistent with a Dyna architecture (Gershman et al., 2014). Participants performed a sequential decision task with separate learning phases that tested behavioral revaluation. When given either more time between phases or a smaller cognitive load, the magnitude of revaluation was larger, consistent with MB replay of past experience. There are also neurophysiological data that suggest Dyna-like cooperation between the two systems. Lansink et al. (2009)



(a) Dyna archetecture.

(b) Model-Based Pseudoreward Approximation.

Figure 1: Schematic illustrations of two approaches to cooperative RL: (a) Dyna and (b) our method, MBPA.



(a) (b)

Figure 2: (a) Maze environment. Colors correspond to state values under the optimal policy (S = start state, G = goal state). (b) Monotonic convergence of estimated state values. Each subplot corresponds to a state in the maze. Red lines are upper-bound estimates, blue lines are lower-bound estimate, and dashed lines are optimal state values.

identified hippocampal neurons in rats encoding spatial location and striatal neurons encoding reward. During sleep, the activation of those hippocampal cells correlated with and proceeded activation of the same striatal cells that encoded the value of those locations.

Model-Based Pseudoreward Approximation

Dyna integrates MF and MB RL by simulating past experience. We now consider Model-Based Pseudoreward Approximation (MBPA), a different way to merge the two. Our method uses dynamic programming to approximate state values. These values are used to calculate pseudorewards according to the shaping theorem. By shaping the reward function, pseudorewards provide a link between MB planning and MF learning. While MBPA can be initialized without a model of the environment, the cognitive phenomena we wish to describe entail situations where the model is already learned, but the MF system is misaligned with it. For example, one may have goals based on a known model of the environment, but habitually behave inconsistently with such goals. MBPA describes how the model can be used to align the two systems.

Pseudorewards and the shaping theorem

Pseudorewards offer a way of conferring extra information to an agent about the reward landscape. Essentially, a small reward is given to the MF agent (a Q-learner in our simulations) whenever it takes an action that helps the agent move towards the goal (or, conversely, a negative reward is given for moving away from the goal). Instead of the agent receiving actual reward $R(s,a,s')$ when moving from state $s \to s'$, the agent receives an augmented reward $R'(s,a,s')$ where

$$R'(s,a,s') = R(s,a,s') + F(s,a,s'). \quad (5)$$

Pseudorewards are defined using *shaping functions*, F. In Ng et al. (1999), conditions for which the optimal policy π^* remains invariant under a shaping function are developed. In particular, F necessarily must be a potential-based shaping function to possess this invariance property:

$$F(s,a,s') = \gamma\Phi(s') - \Phi(s), \quad (6)$$

where Φ is a real-valued function, $\Phi : \mathcal{S} \to \mathbb{R}$. If the shaping function is not potential-based, it is possible that Q-learning will converge to a suboptimal solution. The simplest example of invariant pseudorewards uses the difference in optimal values between the agent's current state and next state:

$$F(s,a,s') = \gamma V^*(s') - V^*(s). \quad (7)$$

This method is called the *optimal policy pseudoreward* – it encourages the agent to move down the optimal path from its current state. With an ε-greedy decision policy, if $\varepsilon = 0$, the agent would move directly to the goal along the shortest path.

With optimal policy pseudorewards the agent can maximize long-term reward simply by taking the most rewarding action at each step. However, in real-world scenarios, it may be unrealistic for a human to have such complete information. Computing the optimal policy may require many iterations of the Bellman equation, or solving a linear program.

Approximating the value function

Optimal policy pseudorewards require knowing the value function under the optimal policy, but that may be costly to compute. Alternatively, the optimal value function can be approximated, requiring less computation. Bounded Real-Time Dynamic Programming (BRTDP) is a planning algorithm that attains certain performance guarantees if its lower- and upper-bounded estimates of state values converge monotonically toward state values under the optimal policy (McMahan, Likhachev, & Gordon, 2005). This monotonic convergence toward optimal values is guaranteed to occur if the lower and upper bounds are initialized properly. Here we take advantage of this monotone property to calculate approximate state values using dynamic programming. Specifically, any num-

ber, n, of iterations of the Bellman equation can be used to approximate state values, and as n increases, the state values converge toward optimal values. These values after n iterations are used to approximate pseudorewards using the shaping theorem. Thus, there is a tradeoff, determined by n, between the proximity of pseudorewards to their optimal value and the amount of computation. As discussed later, learning which n minimizes overall computation is a bounded rationality optimization that can be solved with metacognition.

Linking MF and MB RL with the reward function

Figure 1b provides a schematic illustration of MBPA, wherein dynamic programming is used to approximate pseudorewards, which in turn shape the reward function and policy of the MF agent. We are interested in describing situations in which humans already have a model of the environment and use this information to train their MF instincts. A model containing state-action pairs and reward contingencies is used to estimate state values using n iterations of the Bellman equation. These values are used to calculate pseudorewards with a potential-based shaping function, then added onto real rewards whenever the agent chooses an action. In this way, the MF agent is guided by pseudorewards that are generated using MB RL. In the remainder of the paper we present simulations focused on evaluating MBPA and comparing it to Dyna.

Simulation 1: Maze learning

Methods

Our first simulation involved an agent learning in a maze environment (Sutton, 1991). The agent (a simple Q-learner), began each episode in the upper-left corner of a maze, and was rewarded one point for reaching the lower-right corner (Figure 2a). The state space consisted of 121 locations, 50 of which were walls, in the grid shown in Figure 2a, and actions consisted of each of the four cardinal directions. The agent was trained for fifty episodes, with each episode ending when the goal was reached or 2,000 steps were taken. An ε-greedy decision policy was used with $\varepsilon = 0.25$. The colors in Figure 2a correspond to state values under the optimal policy. Rewards were discounted with $\gamma = 0.95$, and therefor the value of each state is 0.95^d, where d is the minimum number of steps to the goal. In all simulations, simulations were run one-hundred times and averaged.

mcmahan2005bounded **Approximate pseudorewards** Dynamic programing was used to approximate state values by iterating over the Bellman equation. In McMahan et al. (2005) conditions are defined under which initial state values will provably converge monotonically toward optimal values. Here, all states were initialized with a lower bound of zero and an upper bound of one, which in our environment is known to bound state values. Figure 2b shows that the approximate state values for each state do indeed converge monotonically. The point at which each state reaches its optimal value is exactly equal to the minimum number of steps that state is from the goal, d. At each state, the pseudoreward for each action was calculated according to the shaping theorem as the difference between the value of the current state and the value of the next state given that deterministic action. Either the lower-bound or the upper-bound of state values after n iterations of Bellman updates was used to approximate pseudorewards.

Trading off MF and MB computation The closer pseudorewards are to their optimal values (to some precision), the easier the learning for the MF agent. However, whereas Q-learning is simple and quick, the MB method of approximating state values is slower and computationally costly. Therefore, we found the most efficient tradeoff between MB pseudoreward approximation and MF learning. This was done by computing the CPU time required for each algorithm to learn.

Results

Figure 3a shows the number of steps per episode needed to reach the goal, averaged across 50 episodes, as a function of the the number of Bellman updates used to approximate pseudorewards. As expected, learning is quicker when pseudorewards are closer to their optimal values. We also show performance of the Dyna agent as a function of the number of planning steps taken after each real step. While approximate pseudorewards are calculated just once using n iterations, the n planning steps used by Dyna are taken *after every single step of every episode*.

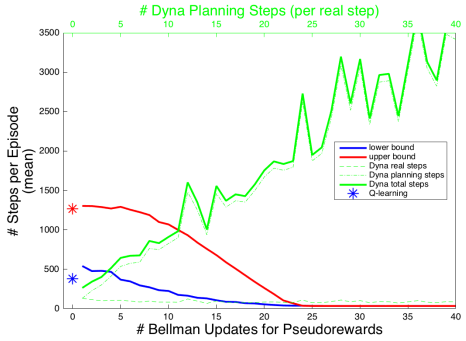
The number of real steps alone taken by the Dyna agent do not converge as low as the MBPA agent. With sufficiently precise pseudorewards, the MBPA agent, on the other hand, can learn the shortest path on the very first episode. Specifically, 24 Bellman updates are required for this, because the start state is 24 steps away from the goal state; after 24 iterations of the Bellman equation, optimal state values have propagated back from the goal state to the start state.

Next, we calculated the actual time required to learn the shortest path. While the pseudoreward method may take fewer steps to reach the goal than Dyna, it does not necessarily mean that it is faster; planning steps (which use scalar operations to update Q-values) are about two orders of magnitude quicker than Bellman updates (which require matrix multiplication). However, Figure 3b shows that MBPA is still faster than Dyna. The fastest learning occurs when 24 iterations of the Bellman equation are used; any more than this is unnecessary and the CPU time increases.

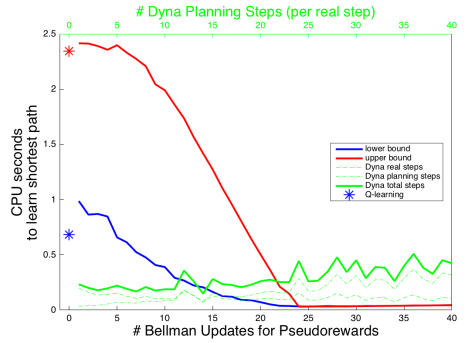
Simulation 2: Mountain car problem

Methods

Simulation 2 explored learning in a standard mountain car environment (Moore, 1990). The agent begins in a valley between two mountains with the goal of reaching the top of the right mountain. The agent must learn to apply force such that it oscillates between the mountain slopes, building momentum until it reaches the top. States consisted of discretized locations and velocities. Actions consisted of discretized forces applied tangentially to the direction of movement. The agent



(a)



(b)

Figure 3: (a) MBPA requires fewer steps to reach the goal than Dyna during maze learning. (b) MBPA learns the shortest path more quickly than Dyna with maze learning.

used Q-learning during 200 learning episodes, where each episode ended when the car reached the goal state or 1,000 steps were taken. When the agent reached the goal it was conferred a reward of one. An ε-greedy decision policy was used where $\varepsilon = 0.01 \times 0.99^{i-1}$, where i is the episode number. As before, MBPA was compared with Dyna.

Results

Simulation 2 showed a similar pattern of results as Simulation 1. The upper-bound and lower-bound estimates of state values converged to optimal values within 48 iterations of the Bellman equation because 48 is the furthest possible number of steps away from the goal. The total number of Dyna steps (real steps plus planning steps) far exceeds the number of steps using MBPA, and the number of real steps alone does not converge as low as the number of steps taken with MBPA.

The CPU time required to learn the shortest path was similarly faster for MBPA. Although Dyna requires more steps to learn, because its computations are scalar-based Q-updates, it is relatively quick, whereas Bellman approximation requires more costly matrix multiplication. Still, MBPA learns faster.

Discussion

We have introduced MBPA, a new method for cooperatively integrating MF and MB RL. This method relies on BRTDP to iteratively estimate state values that converge monotonically to values under the optimal policy. These approximate values are used to calculate pseudorewards according to the shaping theorem, such that the reward function is altered but the optimal policy is invariant. This modified reward function is used for MF learning. Our simulations demonstrate that this method performs comparably to and even better than the Dyna algorithm, a popular cooperative RL method.

The relationship between MF and MB RL has received much attention in cognitive neuroscience research. For example, some work has focused on the neural arbitration between the two systems (Daw et al., 2005), while other work has proposed an underlying unification of both systems (Miller, Shenhav, & Ludvig, 2016). At the computational level of analysis (Marr, 1982), it has been suggested (Boureau et al., 2015) that negotiating these two systems could be understood as a metacognitive solution to a problem of resource rationality (Griffiths et al., 2015). From this perspective, cooperative RL can be understood as a means for transferring computationally intensive MB knowledge to the MF system for quick and easy implementation. The metacognitive optimization in MBPA is the tradeoff between MB state value approximation and the effectiveness of training the MF system. Our results show that certain degrees of state value approximation minimize the computation needed to learn, which results in less computation than Dyna. This point of minimal computation in Figure 3 represents the optimal resource rational solution.

MBPA links MF and MB RL cooperatively by shaping the reward function via pseudoreward. Another interesting example of the interplay between habits and goals is in moral decision-making. It has been suggested that the dual-system approach to moral psychology is well described by the distinction between MF and MB RL, with the former describing the emotional, instinctive, action-oriented, habitual system and the later mapping onto the cognitive, rational, outcome-oriented, and goal-based system (Crockett, 2013; Cushman, 2013). MBPA may provide a direct link between these two systems, with the MB cognitive system producing particular emotions that function as pseudorewards, shaping the MF emotional system. For example, when one's moral behavior deviates from one's moral compass, leading to an untoward outcome, remorse could be generated to correct the action-oriented propensity that produced the misguided behavior.

Another interesting application of cooperative RL and metacognitive rationality is in the formation of sub-goals, or "options" (Huys et al., 2015). Sub-goals offer a way of simplifying complex, high-dimension environments into simpler strategies and MF heuristic-based decisions. MBPA could offer a mechanism for approximating state values of a complex decision space, and transferring this knowledge to a simpler, computationally cheap MF system through reward shaping.

By providing a new way to link MF and MB RL, MBPA offers a new way to think about how the two systems might interact. As discussed earlier, Dyna is readily likened to using MB imagination to train a MF system, which has a natural psychological interpretation. What might be an analog of MBPA in human cognition? In particular, how might pseu-

dorewards – which provide the critical link between systems – manifest cognitively? For any given task or goal-directed behavior, certain emotions often have the effect of altering the reward landscape and functioning as pseudorewards. MBPA proposes that some emotions may represent the (approximate) values of states that are stored in a model, and then used to train MF learning by adding bonuses (positive emotions) or punishments (negative emotions) to certain actions. It is already known that emotions influence MF learning, as in the case of fear conditioning or positive reinforcement (Fields, Hjelmstad, Margolis, & Nicola, 2007; Maren, 2001). These emotions are usually elicited by some external factor; what we are suggesting with MBPA is that the emotions can be produced internally, using an already-learned model of the environment, such as high-level goals.

While we are primarily suggesting that MBPA may function in human cognition, it is interesting to note that humans employ analogous strategies externally. "Temptation bundling" is a strategy whereby a positive association is used to override a negative one (Milkman, Minson, & Volpp, 2013). For example, if one feels averse to exercise but has the goal to get fit, one may watch a favorite television show while on the treadmill; the aversive reward landscape of exercise is positively shaped through deliberate coupling with rewarding behavior. The positive emotions generated from the TV show function as pseudorewards to guide the person along a more optimal route toward exercise and fitness.

Finally, because our interest is in describing the relationship between acquired MB knowledge and goals with MF behaviors or habits, our MBPA algorithm uses a MB system with knowledge of the casual structure of the environment. MBPA can alternatively learn a model of the environment rather than inherit it, but it is not designed to excel under such circumstances. Conversely, Dyna can inherit such knowledge, but it actually performs worse than the Dyna agent presented here, because replay becomes less efficient. "Prioritized sweeping" (Moore & Atkeson, 1993) is a more efficient version of Dyna, but for the problems we present, we found that MBPA still performs faster and in fewer steps.

Dual-process theories are abundant in psychology, but there is a dearth of computational theories for understanding the precise relationship between dual systems. MBPA offers a new way to understand interaction between MB and MF systems, with natural cognitive and affective psychological interpretations. We hope that MBPA will inspire new questions to be pursued experimentally.

References

Aarts, H., & Dijksterhuis, A. (2000). Habits as knowledge structures: automaticity in goal-directed behavior. *Journal of personality and social psychology*, *78*(1), 53.

Barto, A. G., Bradtke, S. J., & Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, *72*(1), 81–138.

Boureau, Y.-L., Sokol-Hessner, P., & Daw, N. D. (2015). Deciding how to decide: Self-control and meta-decision making. *Trends in cognitive sciences*, *19*(11), 700–710.

Crockett, M. J. (2013). Models of morality. *Trends in cognitive sciences*, *17*(8), 363–366.

Cushman, F. (2013). Action, outcome, and value a dual-system framework for morality. *Personality and social psychology review*, *17*(3), 273–292.

Daw, N. D., & Dayan, P. (2014). The algorithmic anatomy of model-based evaluation. *Phil. Trans. R. Soc. B*, *369*(1655), 20130478.

Daw, N. D., Niv, Y., & Dayan, P. (2005). Uncertainty-based competition between prefrontal and dorsolateral striatal systems for behavioral control. *Nature neuroscience*, *8*(12), 1704–1711.

Dayan, P., & Berridge, K. C. (2014). Model-based and model-free pavlovian reward learning: revaluation, revision, and revelation. *Cognitive, Affective, & Behavioral Neuroscience*, *14*(2), 473–492.

Dolan, R. J., & Dayan, P. (2013). Goals and habits in the brain. *Neuron*, *80*(2), 312–325.

Fields, H. L., Hjelmstad, G. O., Margolis, E. B., & Nicola, S. M. (2007). Ventral tegmental area neurons in learned appetitive behavior and positive reinforcement. *Annu. Rev. Neurosci.*, *30*, 289–316.

Gershman, S. J., Markman, A. B., & Otto, A. R. (2014). Retrospective revaluation in sequential decision making: A tale of two systems. *Journal of Experimental Psychology: General*, *143*(1), 182.

Gläscher, J., Daw, N., Dayan, P., & O'Doherty, J. P. (2010). States versus rewards: dissociable neural prediction error signals underlying model-based and model-free reinforcement learning. *Neuron*, *66*(4), 585–595.

Griffiths, T. L., Lieder, F., & Goodman, N. D. (2015). Rational use of cognitive resources: Levels of analysis between the computational and the algorithmic. *Topics in cognitive science*, *7*(2), 217–229.

Huys, Q. J., Lally, N., Faulkner, P., Eshel, N., Seifritz, E., Gershman, S. J., ... Roiser, J. P. (2015). Interplay of approximate planning strategies. *Proceedings of the National Academy of Sciences*, *112*(10), 3098–3103.

Maren, S. (2001). Neurobiology of pavlovian fear conditioning. *Annual review of neuroscience*, *24*(1), 897–931.

Marr, D. (1982). Vision. san francisco: W. h. *H. Freeman*.

McMahan, H. B., Likhachev, M., & Gordon, G. J. (2005). Bounded real-time dynamic programming: Rtdp with monotone upper bounds and performance guarantees. In *Proceedings of the 22nd international conference on machine learning* (pp. 569–576).

Milkman, K. L., Minson, J. A., & Volpp, K. G. (2013). Holding the hunger games hostage at the gym: An evaluation of temptation bundling. *Management science*, *60*(2), 283–299.

Miller, K., Shenhav, A., & Ludvig, E. (2016). Habits without values. *bioRxiv*, 067603.

Moore, A. W. (1990). *Efficient memory-based learning for robot control*. Unpublished doctoral dissertation, University of Cambridge, Cambridge, UK.

Moore, A. W., & Atkeson, C. G. (1993). Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, *13*(1), 103–130.

Ng, A. Y., Harada, D., & Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml* (Vol. 99, pp. 278–287).

Otto, A. R., Gershman, S. J., Markman, A. B., & Daw, N. D. (2013). The curse of planning dissecting multiple reinforcement-learning systems by taxing the central executive. *Psychological science*, 0956797612463080.

Sutton, R. S. (1991). Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, *2*(4), 160–163.

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

Sutton, R. S., Barto, A. G., & Williams, R. J. (1992). Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems*, *12*(2), 19–22.

Thorndike, E. L. (1933). A proof of the law of effect. *Science*.

Tolman, E. C. (1948). Cognitive maps in rats and men. *Psychological review*, *55*(4), 189.