
Lecture 7: Computational Cognitive Modeling

Reinforcement Learning (pt. 2-3.0)

email address for instructors:
instructors-ccm-spring2018@nyucll.org

course website:
<https://brendenlake.github.io/CCM-site/>

Today: Evaluating the world when you DO know something about it

- **Model-based RL** - learning about states and actions
- **Cognitive science/Neuroscience of RL**

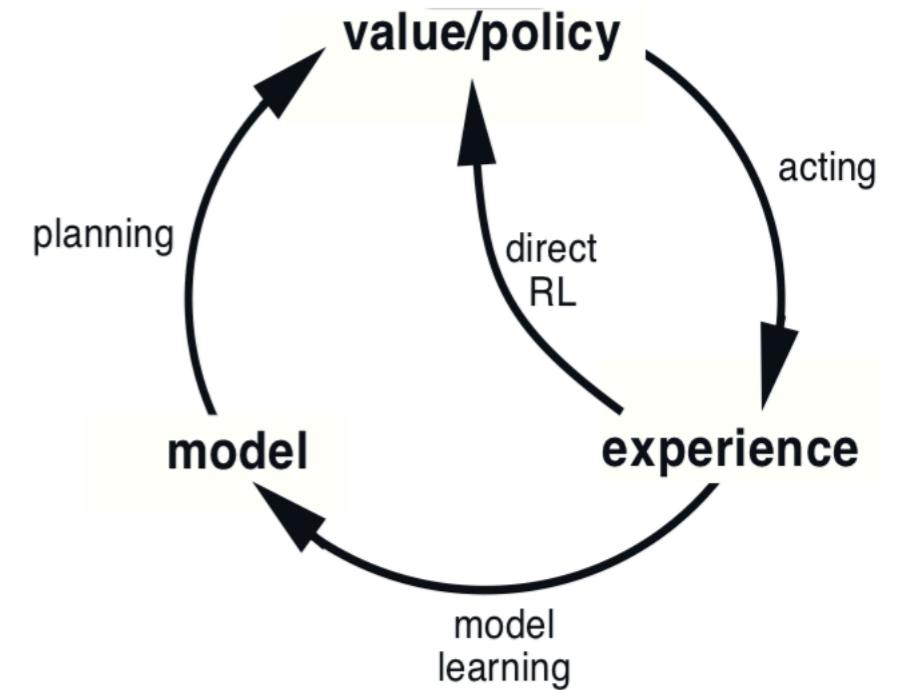
Planning or Model-based Learning

Temporal-Difference/Monte-carlo methods we have considered so far are called “model-free” because they don’t maintain or learn information about the state transitions and rewards explicitly.

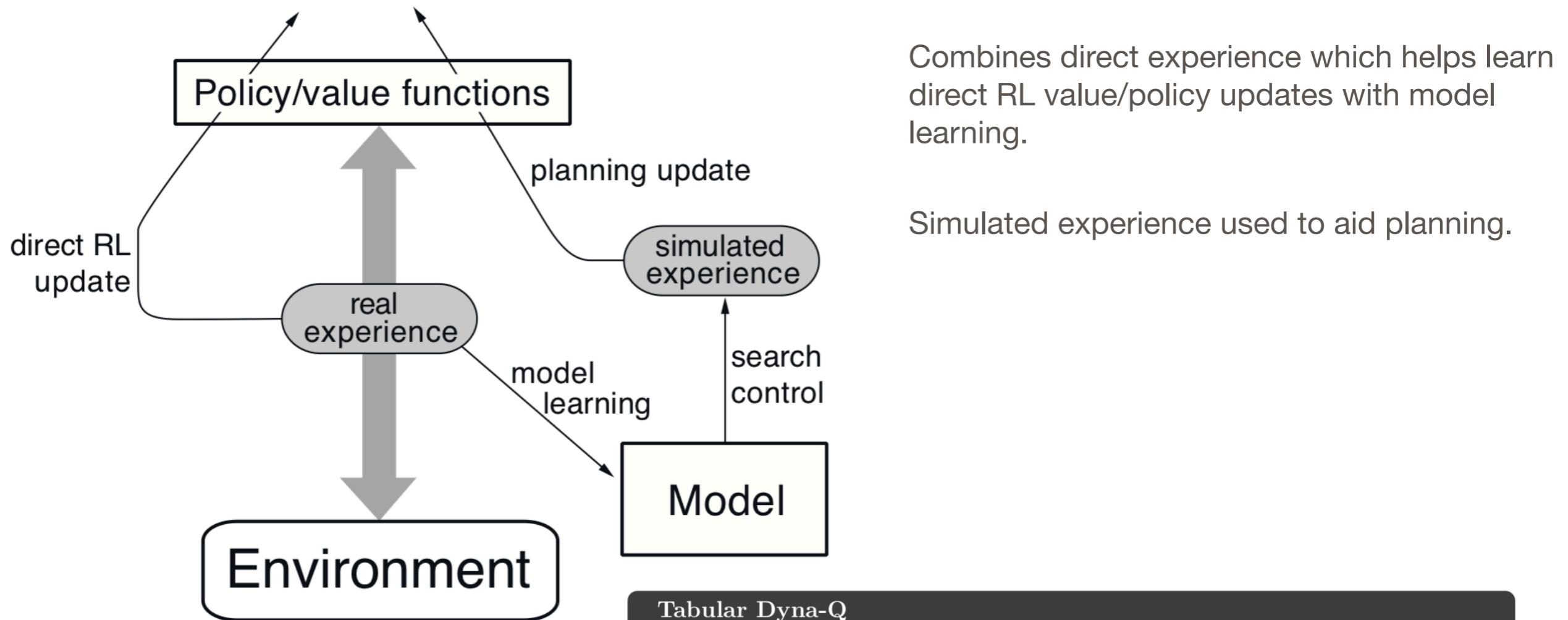
In contrast, Dynamic Programming needs a model of the environment in terms of the transition probabilities and rewards.

If you have a model you can distinguish **planning from learning**.

Planning - predict what will happen in the future... what state will I end up in, what reward will I get?



The DYNA algorithm



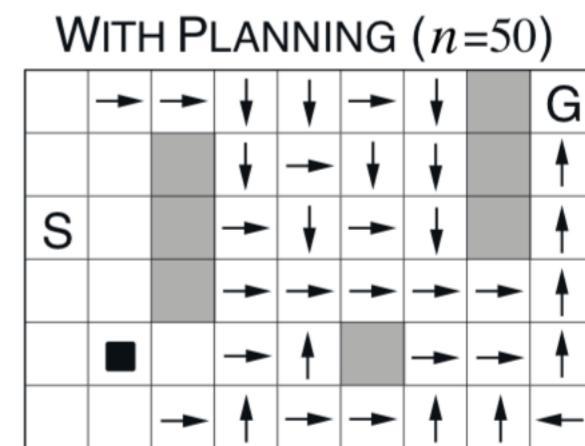
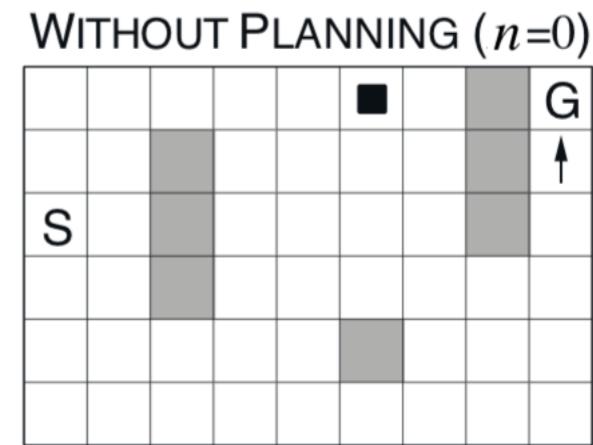
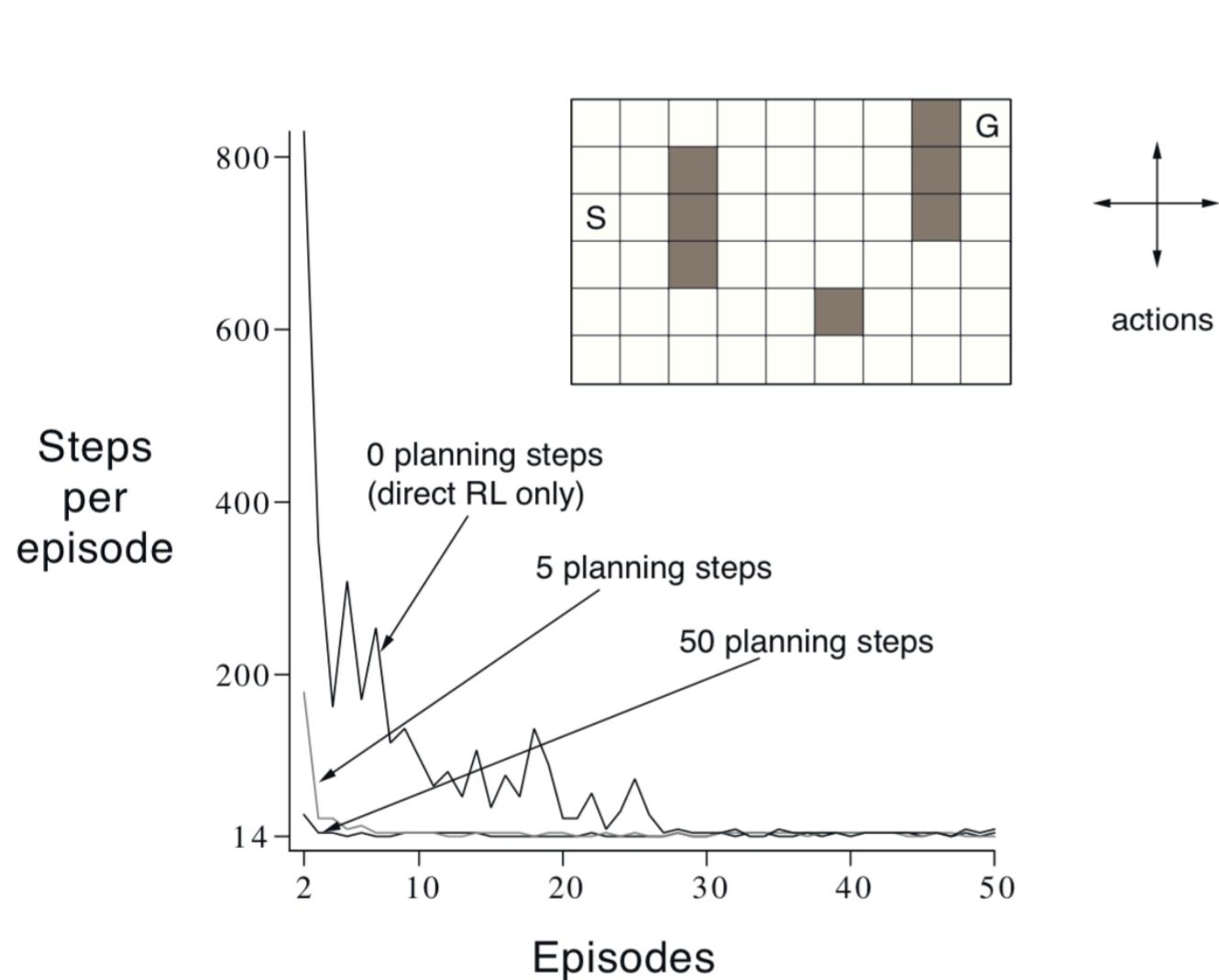
Tabular Dyna-Q

Initialize $Q(s, a)$ and $Model(s, a)$ for all $s \in \mathcal{S}$ and $a \in \mathcal{A}(s)$

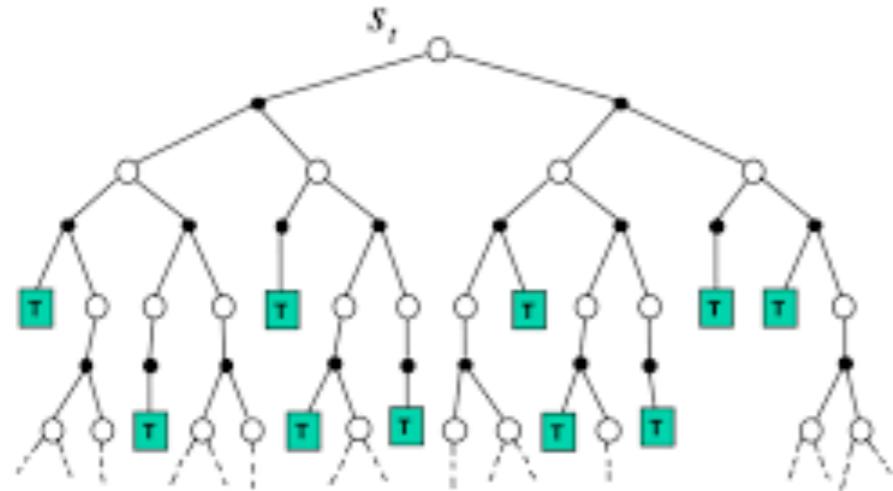
Loop forever:

- (a) $S \leftarrow$ current (nonterminal) state
- (b) $A \leftarrow \varepsilon\text{-greedy}(S, Q)$
- (c) Take action A ; observe resultant reward, R , and state, S'
- (d) $Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$
- (e) $Model(S, A) \leftarrow R, S'$ (assuming deterministic environment)
- (f) Loop repeat n times:
 $S \leftarrow$ random previously observed state
 $A \leftarrow$ random action previously taken in S
 $R, S' \leftarrow Model(S, A)$
 $Q(S, A) \leftarrow Q(S, A) + \alpha [R + \gamma \max_a Q(S', a) - Q(S, A)]$

The DYNA algorithm



Online planning

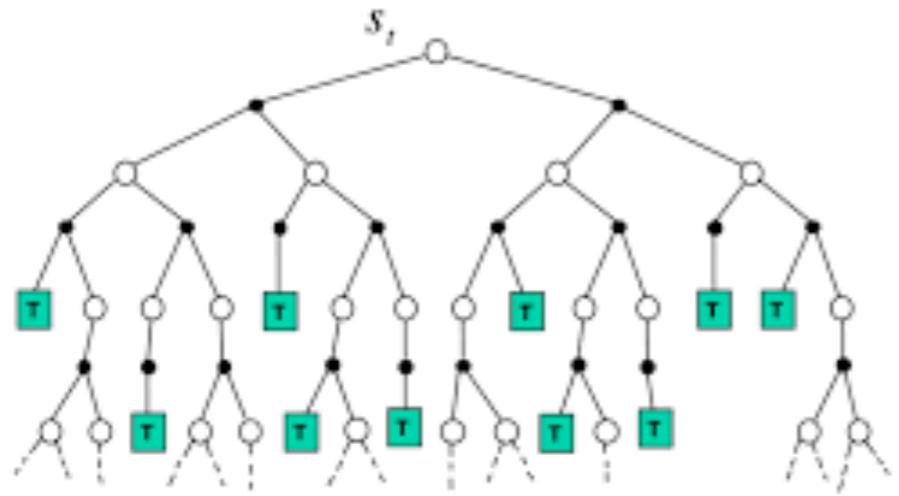


From each state, if we have a good model of environment we can simulate forward many times to determine the value of the current state or action.

However, the branching factor of this tree can be a major problem.

One recent innovation is **Monte Carlo Tree Search (MCTS)**

Monte Carlo Tree Search



$$\bar{x}_i \pm \sqrt{\frac{2 \ln n}{n_i}}$$

- \bar{x}_i : the mean payout for machine i
- n_i : the number of plays of machine i
- n : the total number of plays
- Pick option with highest confidence bound

- In a standard Monte Carlo process, a large number of random simulations are run, in this case, from the board position that you want to find the best move for.
- The downside to this method, though, is that for any given turn in the simulation, there may be many possible moves, but only one or two that are good.
- Instead of doing many purely random simulations, UCT works by doing many multi-phase playouts where each position is itself viewed as a multi-armed bandit.

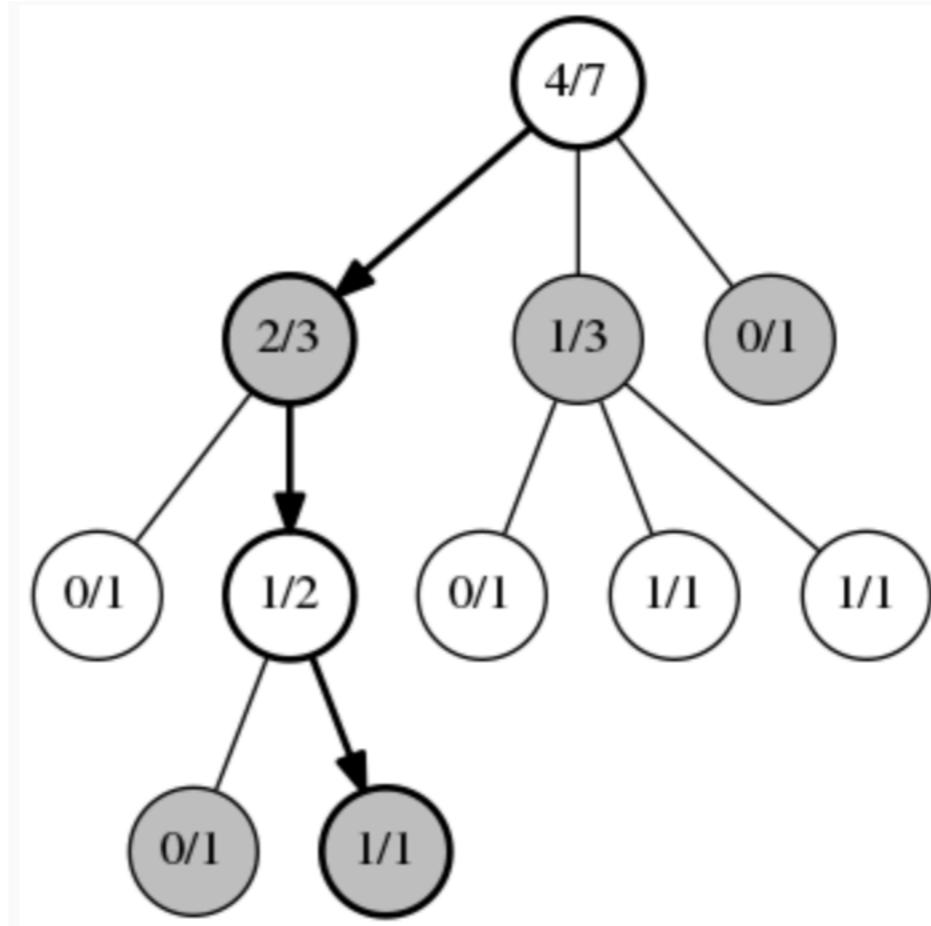
UCB1

Monte Carlo Tree Search

$$\bar{x}_i \pm \sqrt{\frac{2 \ln n}{n_i}}$$

- \bar{x}_i : the mean payout for machine i
- n_i : the number of plays of machine i
- n : the total number of plays

UCB1



Selection

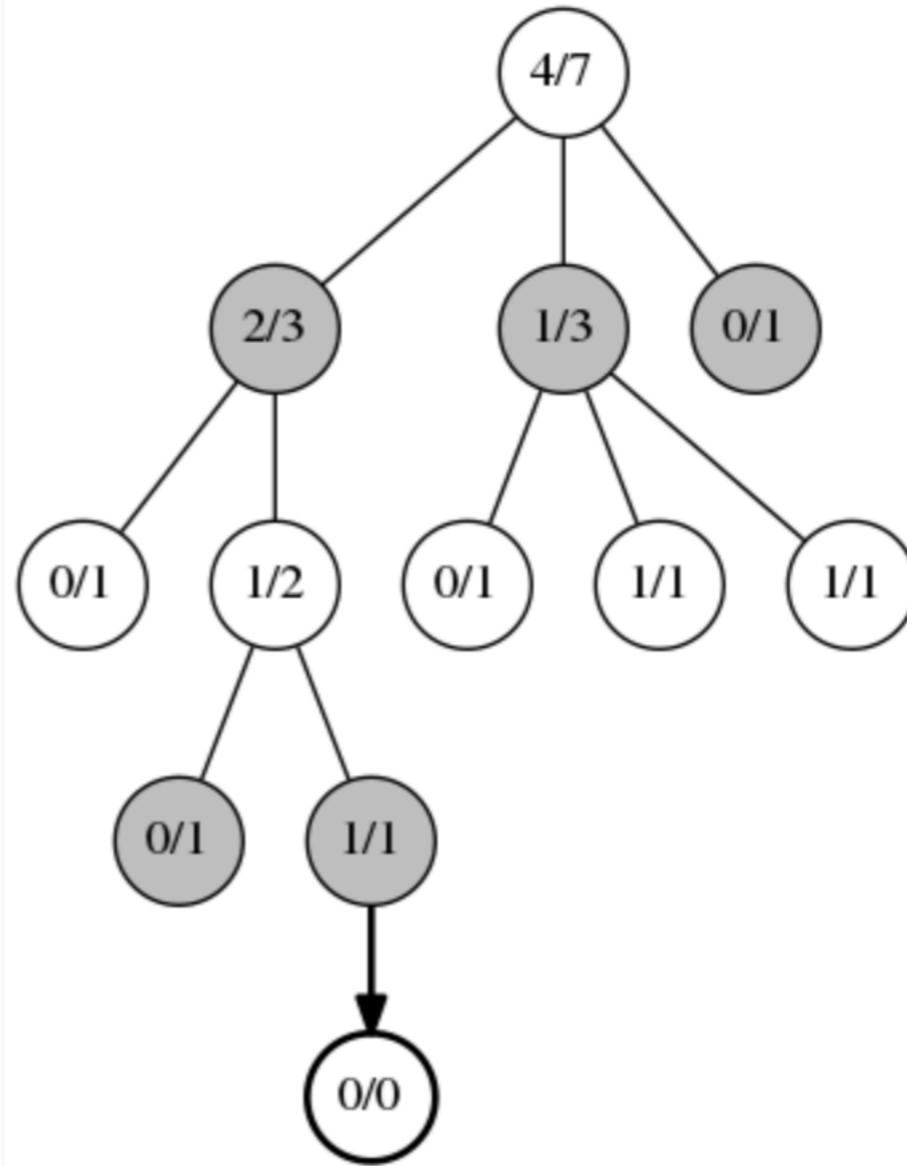
Here the positions and moves selected by the UCB1 algorithm at each step are marked in bold. Note that a number of playouts have already been run to accumulate the statistics shown. Each circle contains the number of wins / number of times played.

Monte Carlo Tree Search

$$\bar{x}_i \pm \sqrt{\frac{2 \ln n}{n_i}}$$

- \bar{x}_i : the mean payout for machine i
- n_i : the number of plays of machine i
- n : the total number of plays

UCB1



Expansion

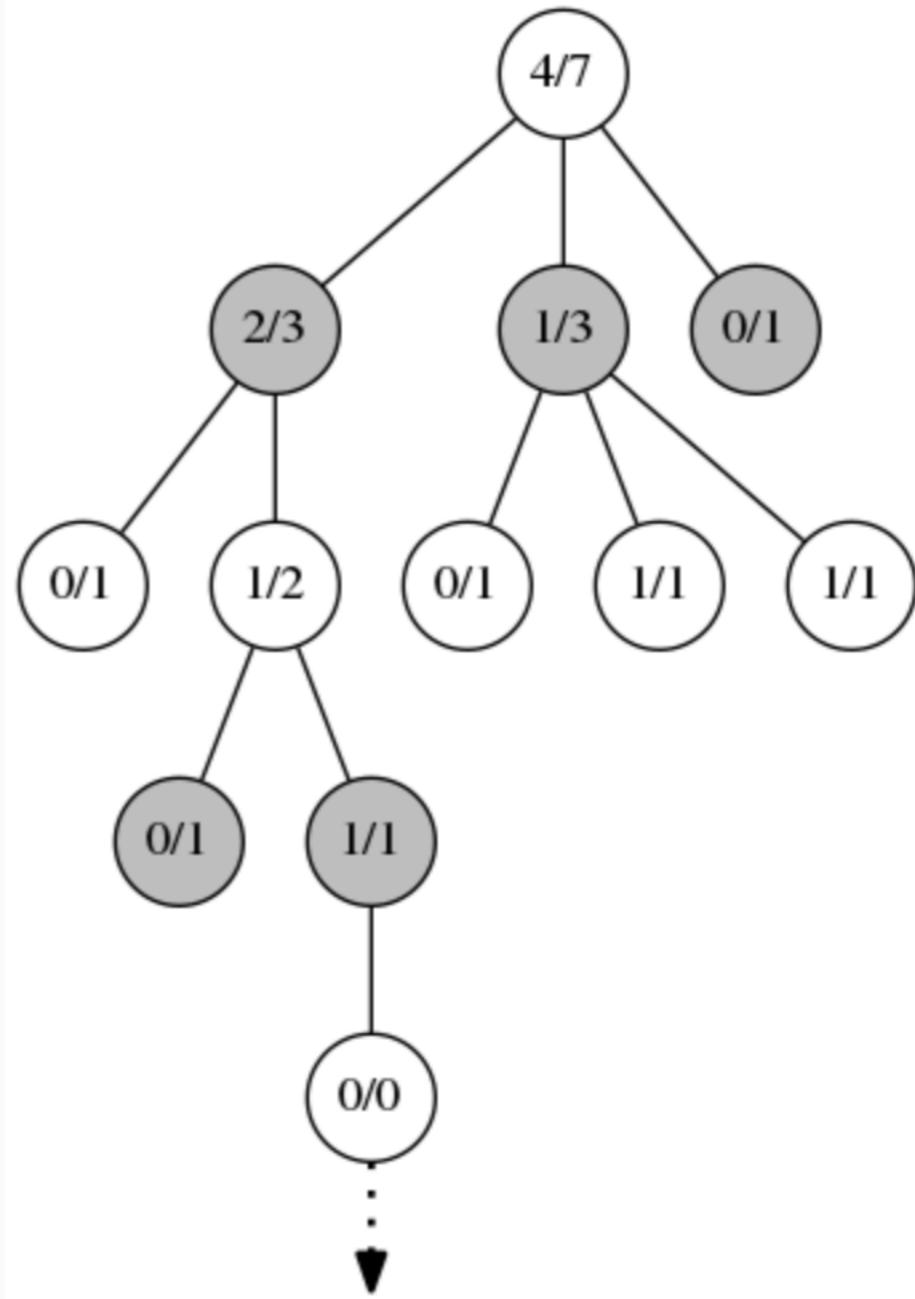
The position marked 1/1 at the bottom of the tree has no further statistics records under it, so we choose a random move and add a new record for it (bold), initialized to 0/0.

Monte Carlo Tree Search

$$\bar{x}_i \pm \sqrt{\frac{2 \ln n}{n_i}}$$

- \bar{x}_i : the mean payout for machine i
- n_i : the number of plays of machine i
- n : the total number of plays

UCB1



Simulation

Once the new record is added, the Monte Carlo simulation begins, here depicted with a dashed arrow. Moves in the simulation may be completely random, or may use calculations to weight the randomness in favor of moves that may be better.

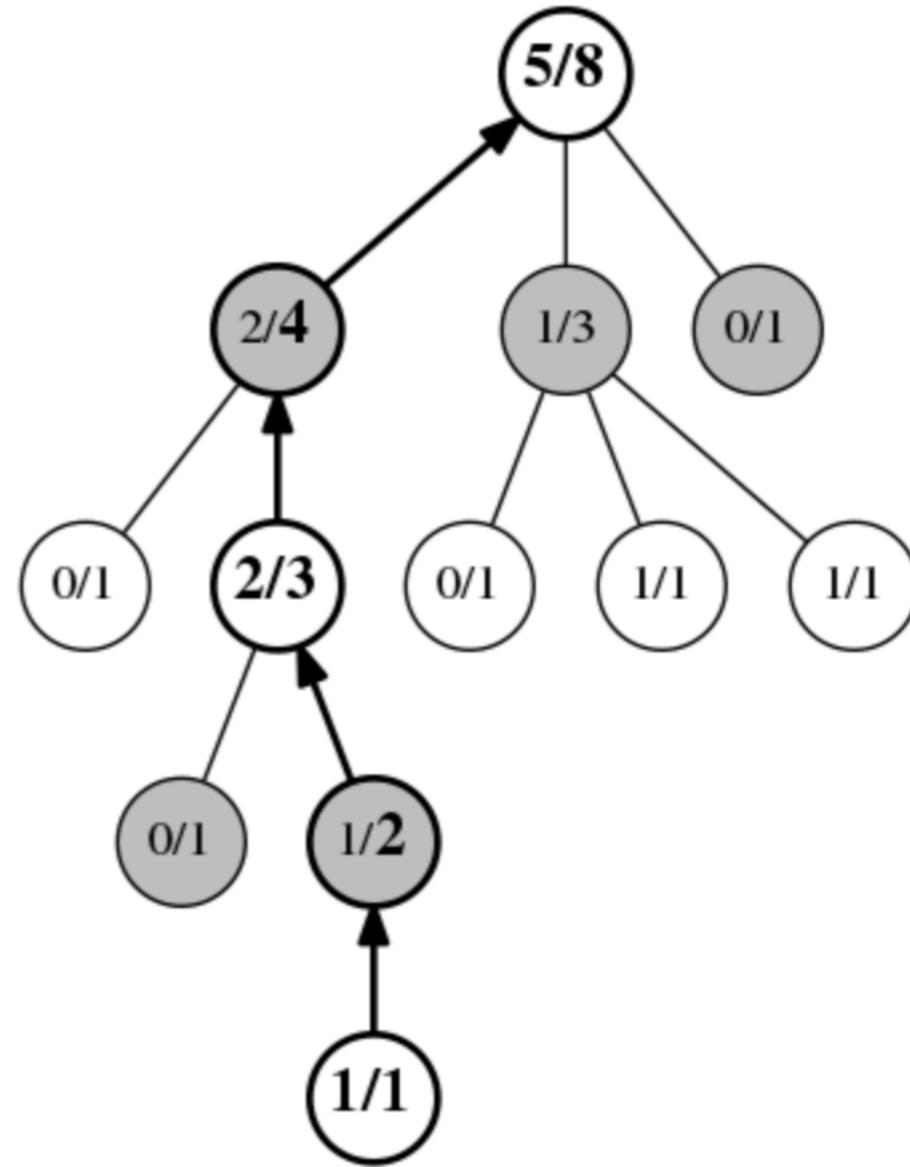
Monte Carlo Tree Search

- As more and more playouts are run, the tree of statistics grows in memory and the move that will finally be chosen will converge towards the actual optimal play, though that may take a very long time, depending on the game.

$$\bar{x}_i \pm \sqrt{\frac{2 \ln n}{n_i}}$$

- \bar{x}_i : the mean payout for machine i
- n_i : the number of plays of machine i
- n : the total number of plays

UCB1



Back-Propagation

After the simulation reaches an end, all of the records in the path taken are updated. Each has its play count incremented by one, and each that matches the winner has its win count incremented by one, here shown by the bolded numbers.

Three levels of description (*David Marr, 1982*)

Computational

Why do things work the way they do?
What is the goal of the computation?
What are the unifying principles?



Algorithmic

What representations can implement such computations?
How does the choice of representations determine the algorithm?

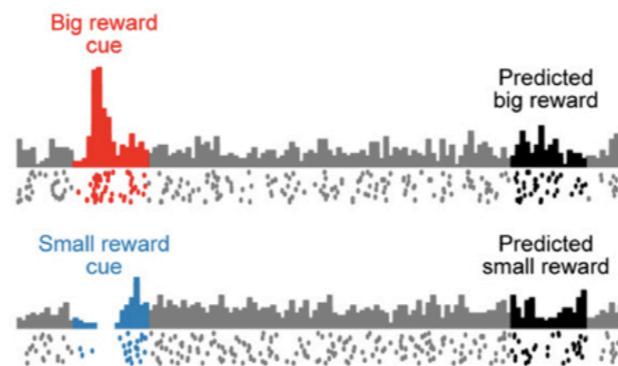
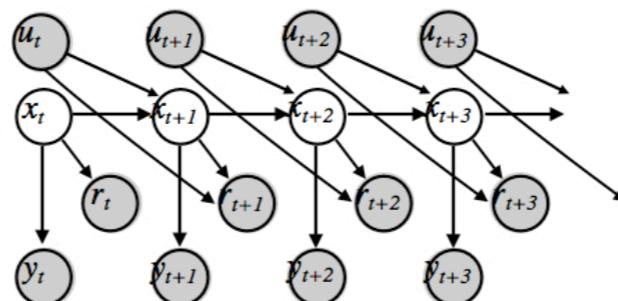
Implementational

How can such a system be built in hardware?
How can neurons carry out the computations?

maximize:

$$R_t = r_{t+1} + r_{t+2} + \dots + r_T$$

Bellman

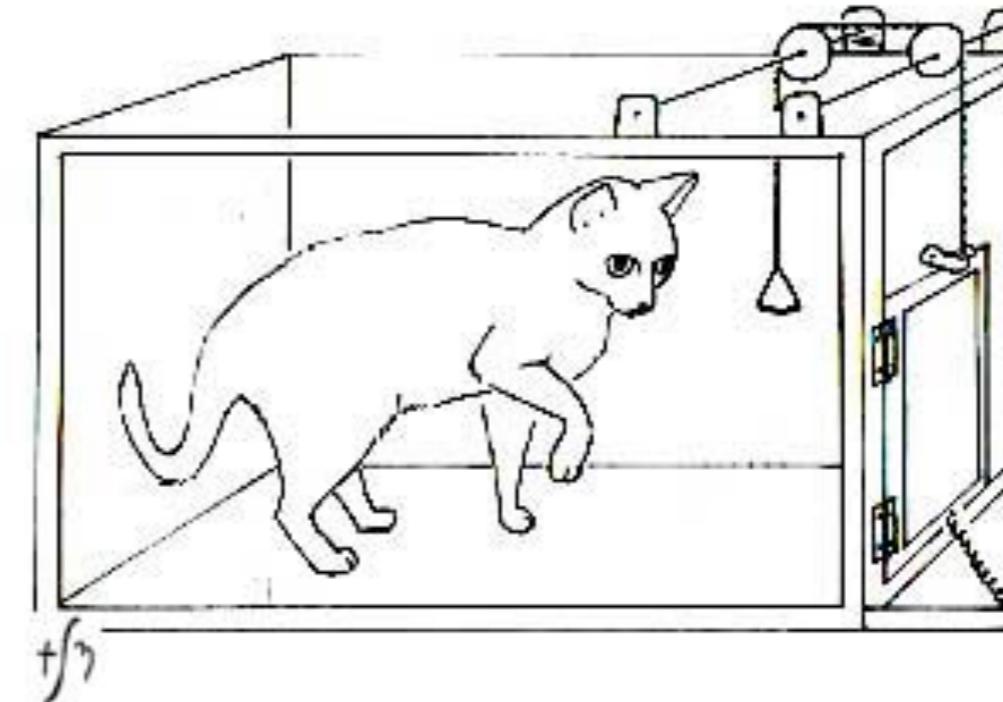
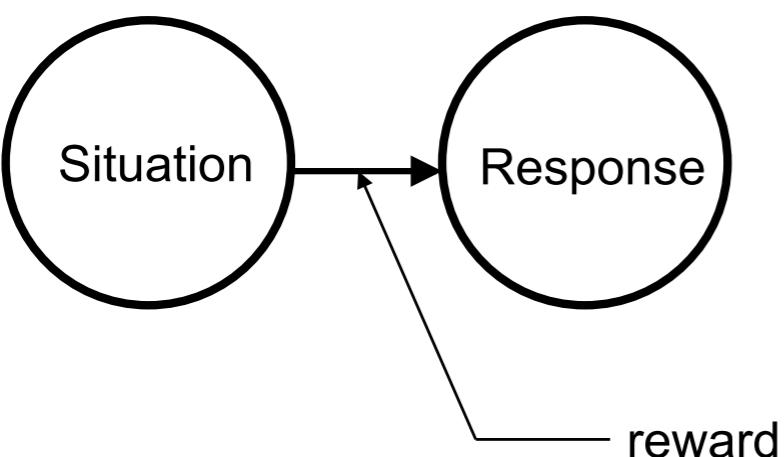


Dynamic programming,
TD methods, Monte
Carlo

Neural firing patterns,
prediction errors,
system level
neuroscience

Edward Thorndike

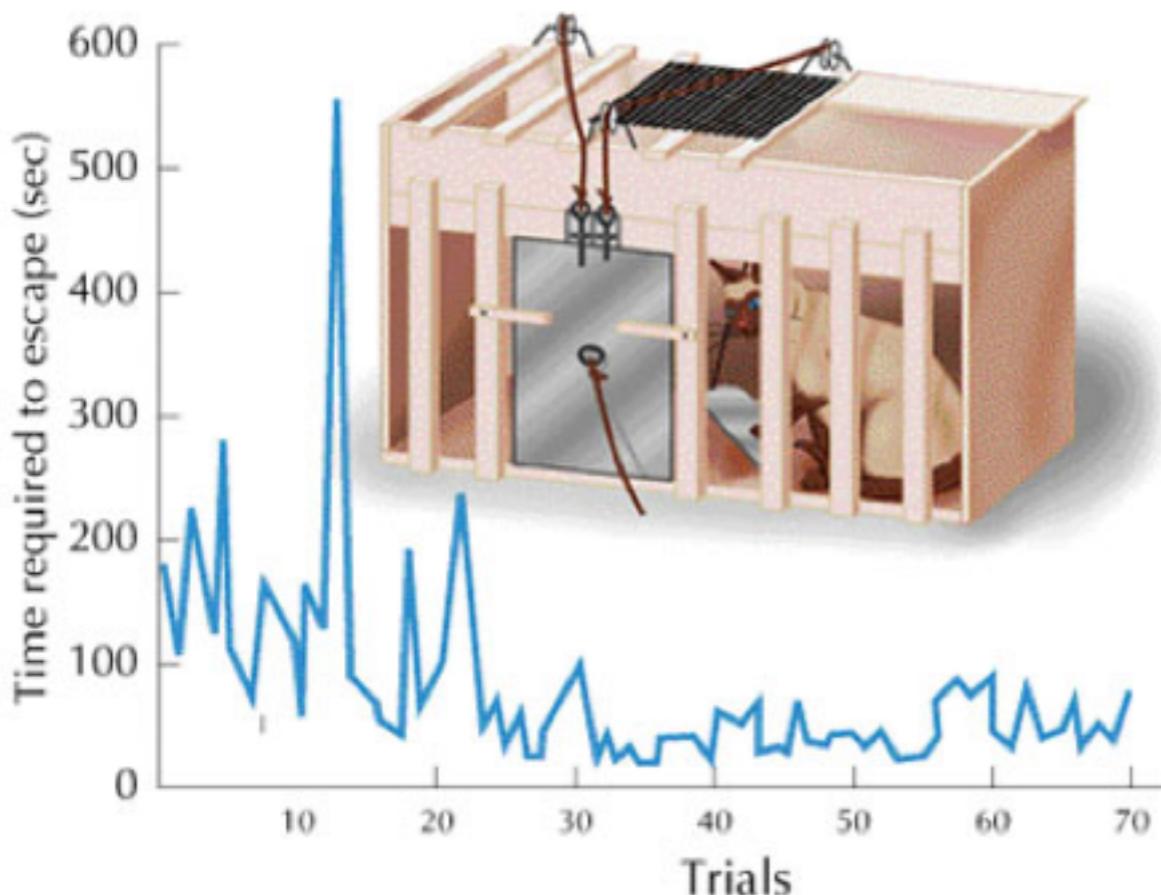
Law of Effect



“Of several responses made to the same situation, those which are accompanied or closely **followed by satisfaction** to the animal **will**, other things being equal, **be more firmly connected with the situation**, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur” (Thorndike, 1911)

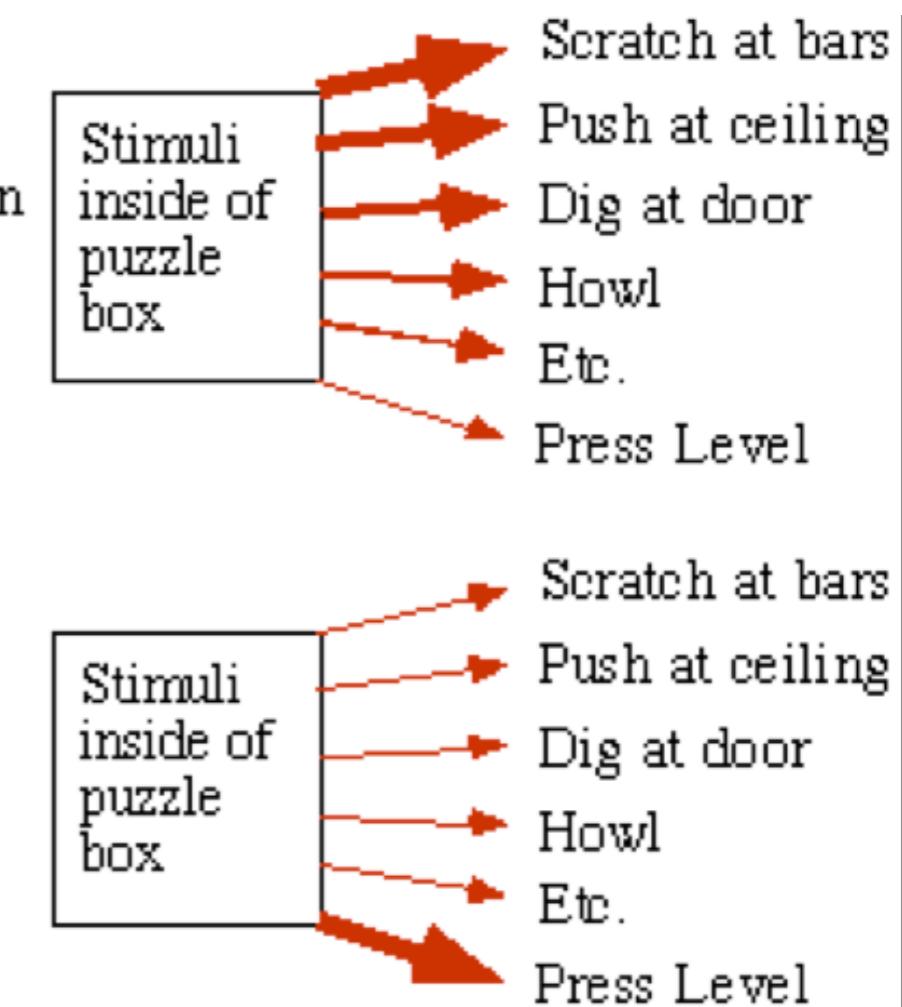
Edward Thorndike

Law of Effect



First trial in
puzzle box

After many
trials in
puzzle box



classical conditioning



- prediction
- ... revealed by behavior
- ... shaped by learning

Maximizing Reward

The learning algorithm which approximates the full Bellman Solution (temporal difference):

$$V_t = E[r_{t+1}] + V_{t+1}$$

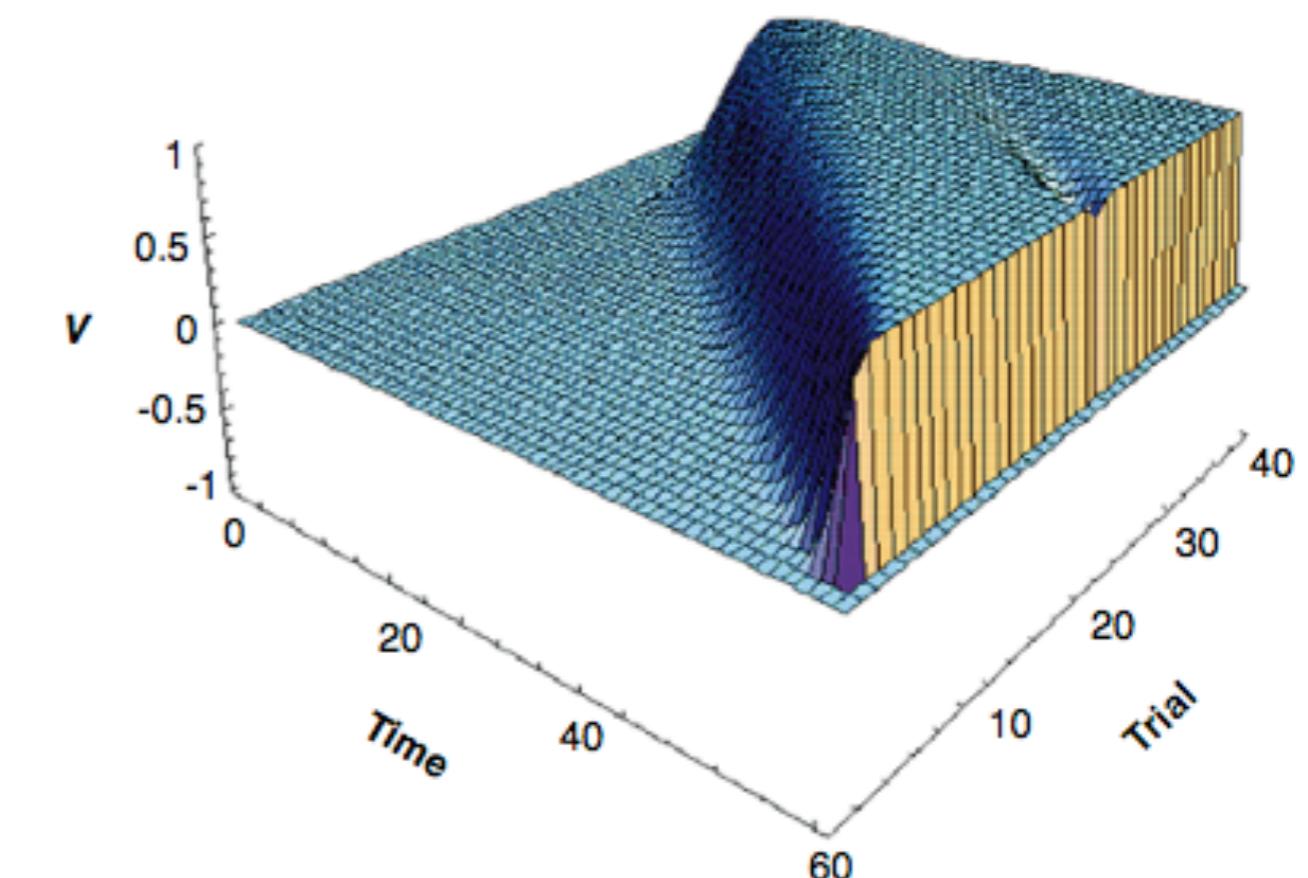
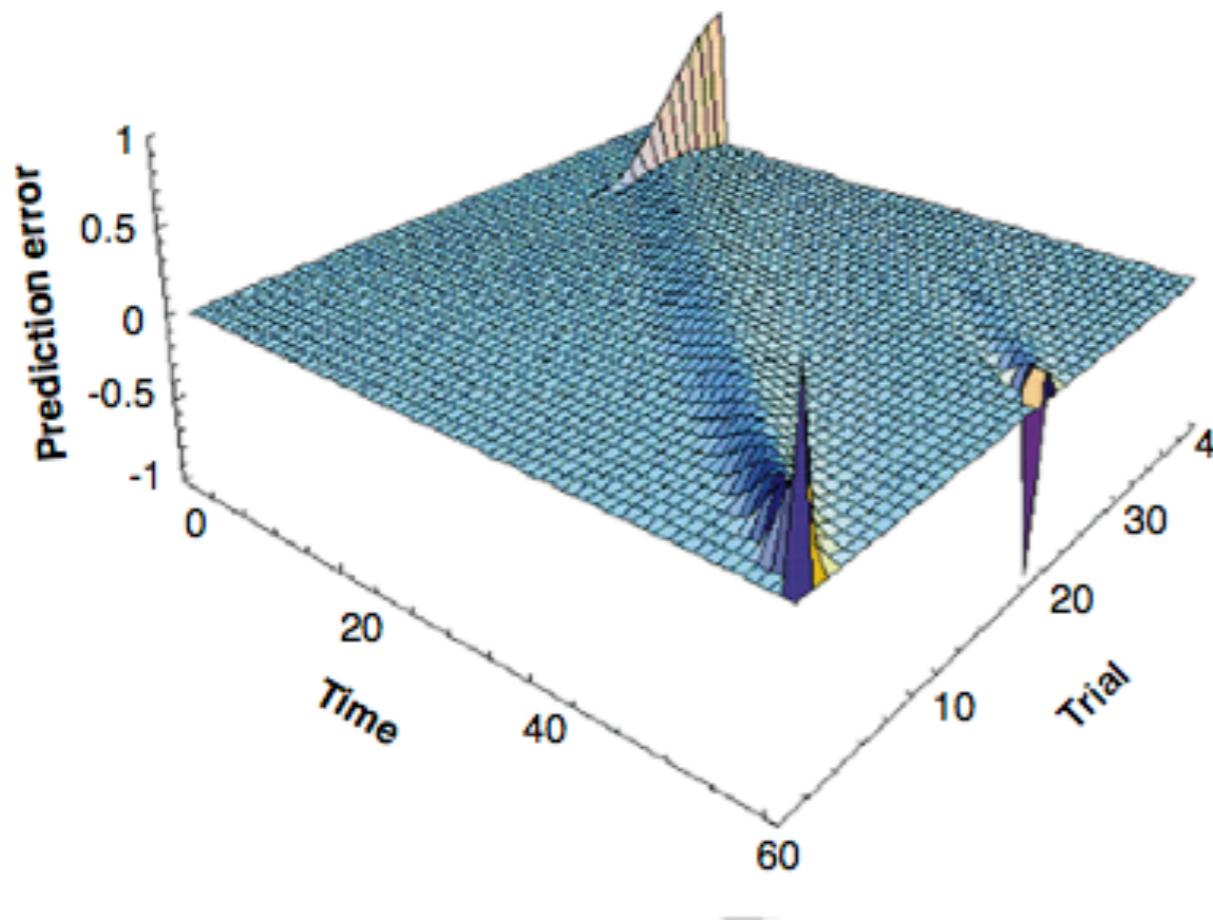
$$V_t \leftarrow V_t + \eta(r_{t+1} + V_{t+1} - V_t)$$

$$V_t \leftarrow (1 - \eta)V_t + \eta(r_{t+1} + V_{t+1})$$

*Compare with **Rescorla-Wagner***

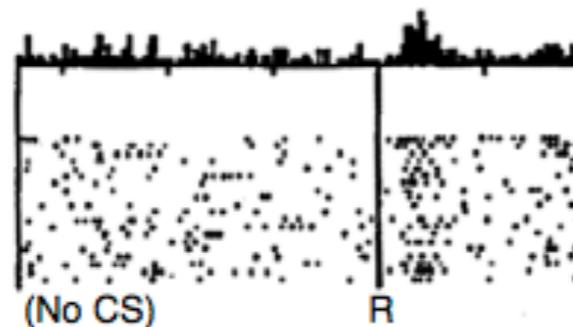
$$V_t \leftarrow V_t + \eta(r_{t+1} + V_t)$$

Temporal-difference learning and “backing up”

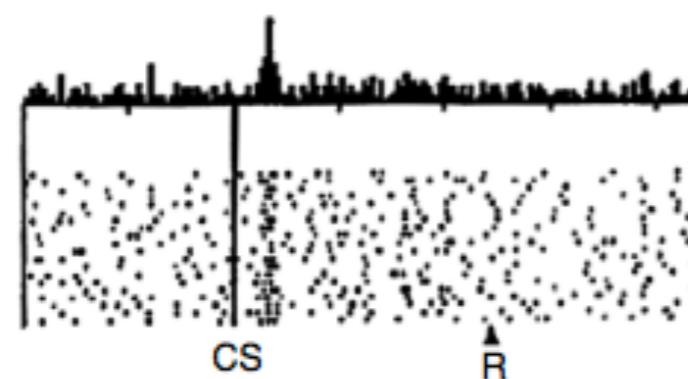


**Do dopamine neurons report an error
in the prediction of reward?**

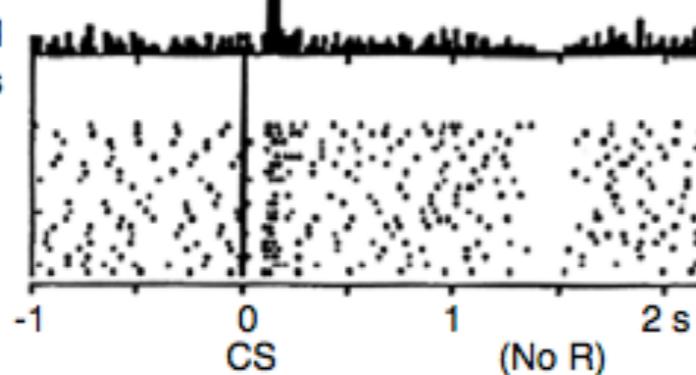
No prediction
Reward occurs



Reward predicted
Reward occurs



Reward predicted
No reward occurs

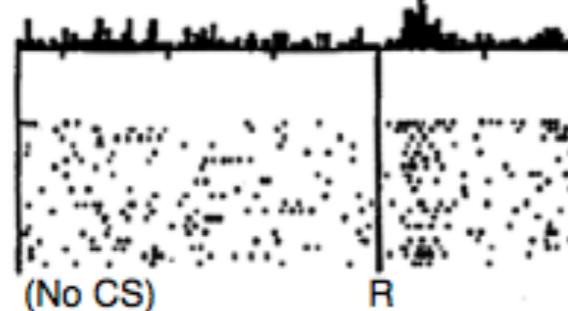


**Dopamine
neurons do it
too...**

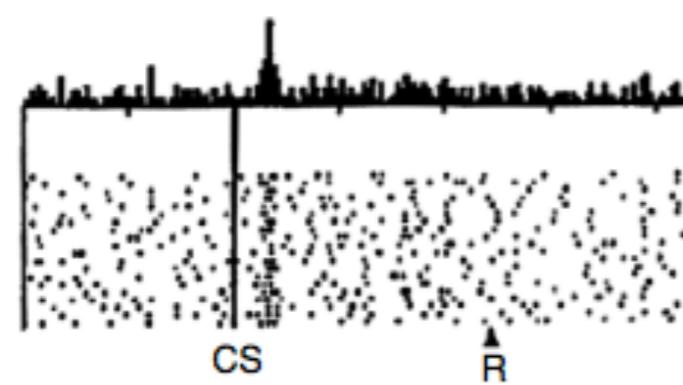
Dopamine neurons do it too...

Do dopamine neurons report an error
in the prediction of reward?

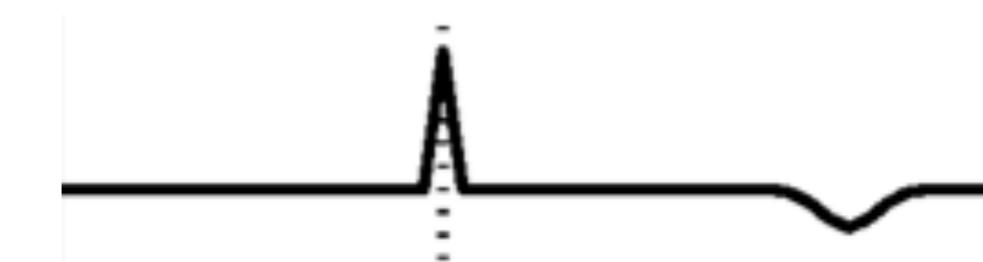
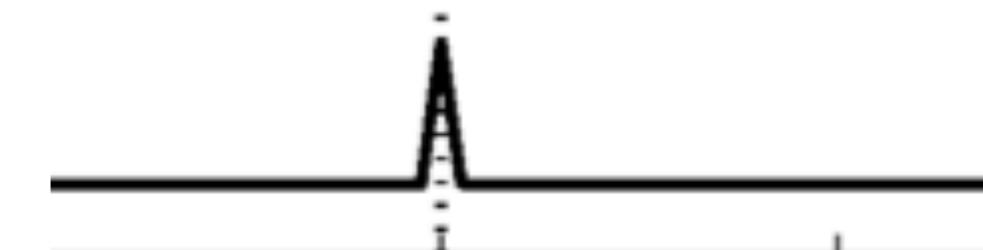
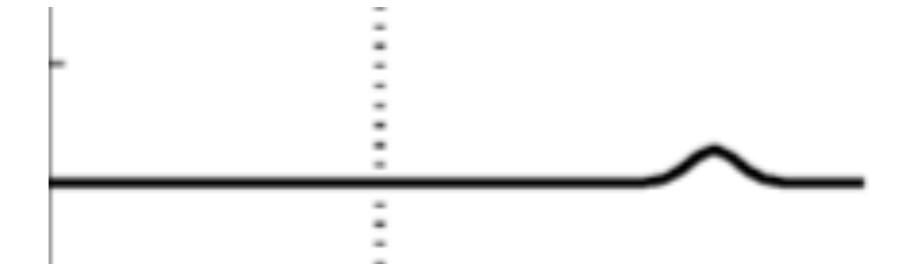
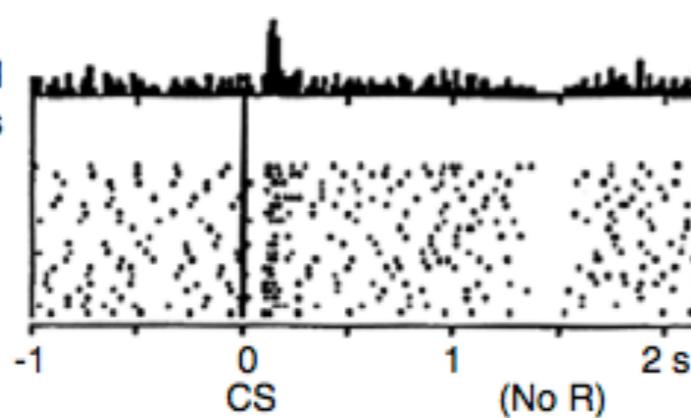
No prediction
Reward occurs



Reward predicted
Reward occurs

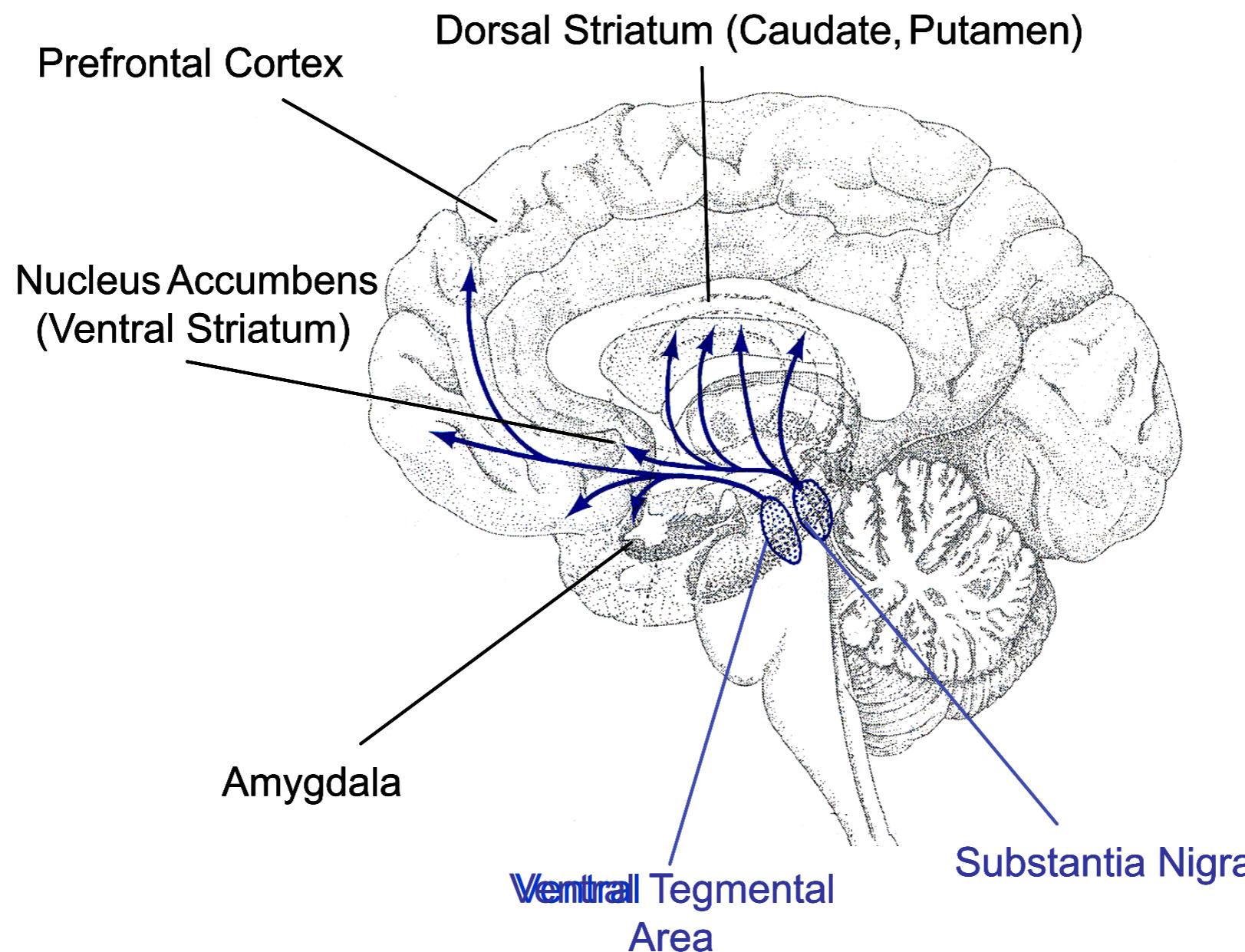


Reward predicted
No reward occurs



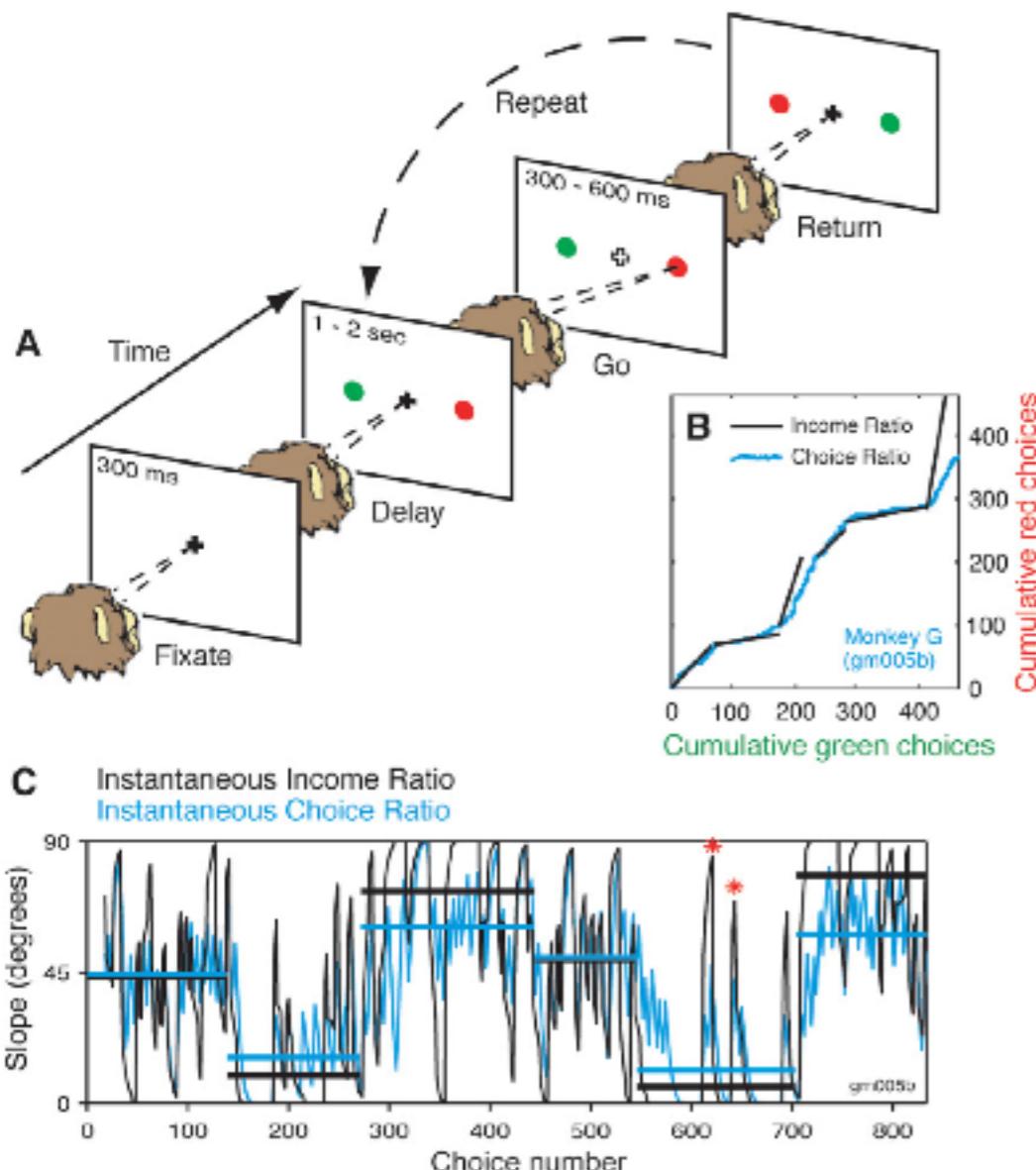
*Also correlates with magnitude of reward, degree
of intermittent reinforcement, etc...*

The role of dopamine in reward learning



Matching Behavior and the Representation of Value in the Parietal Cortex

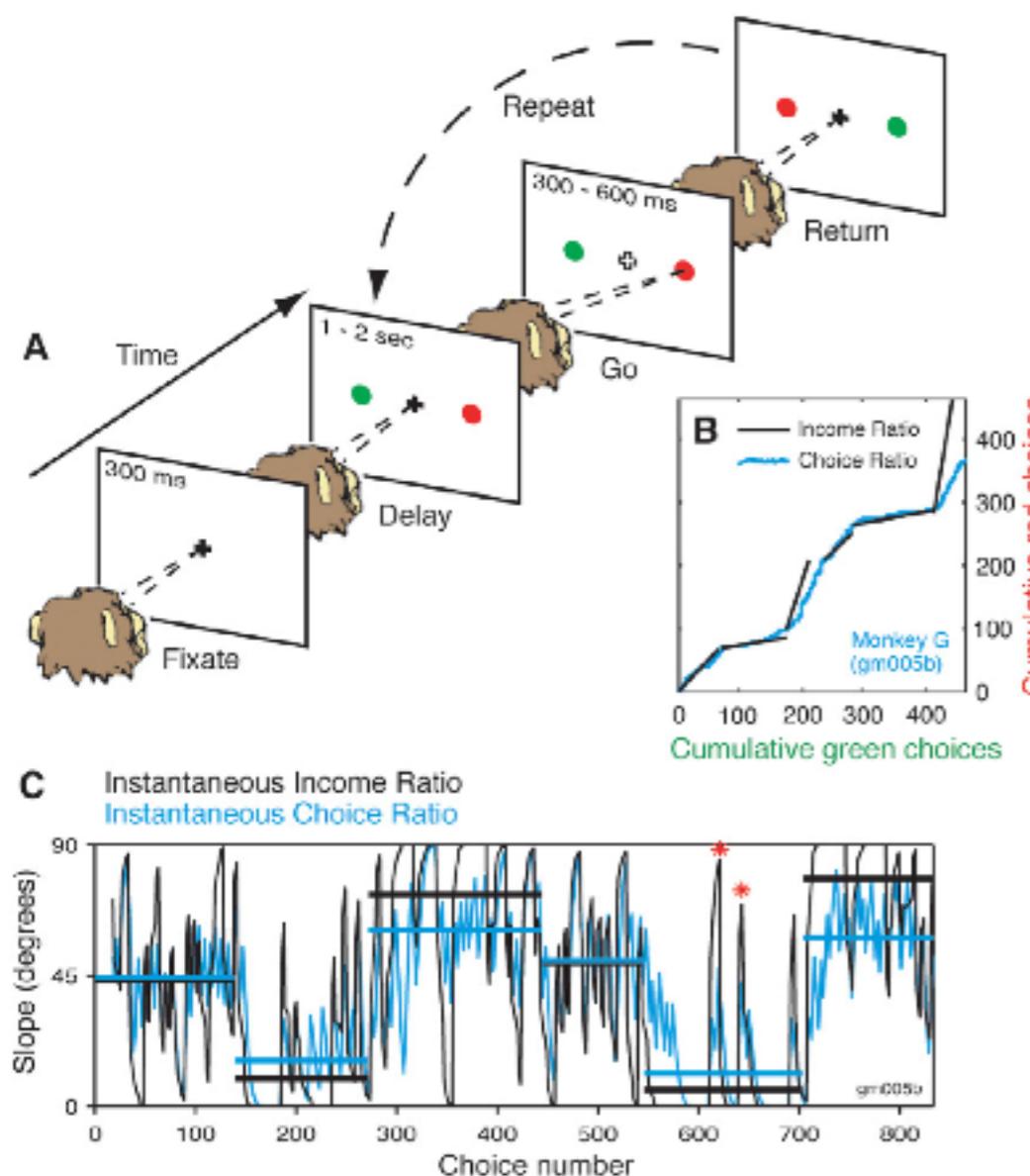
Leo P. Sugrue,* Greg S. Corrado, William T. Newsome



- Rhesus monkeys make eye fixations to one of two locations which get rewarded at different rates
- Dynamic foraging task since the reward from each option “drifts” in time (Poisson arrival times, but one are baited are always available... like VI schedule, makes matching the optimal strategy)
- Ever 200 or so trials, the reward magnitudes or rates change suddenly causing discontinuous shifts in the reward rates

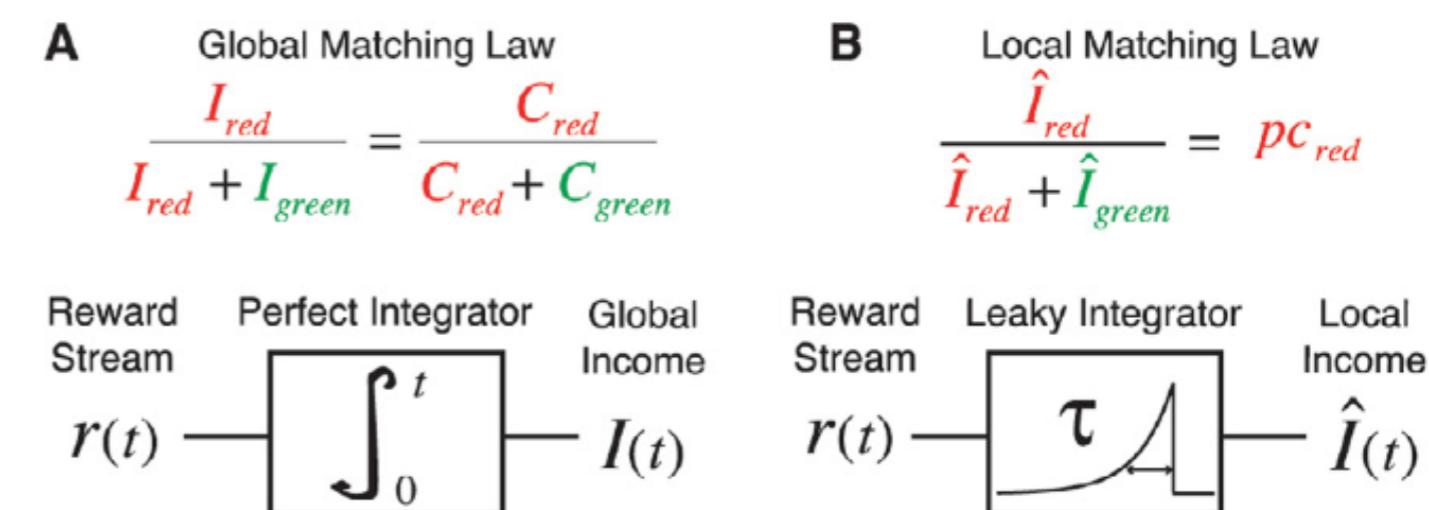
Matching Behavior and the Representation of Value in the Parietal Cortex

Leo P. Sugrue,* Greg S. Corrado, William T. Newsome



Learning

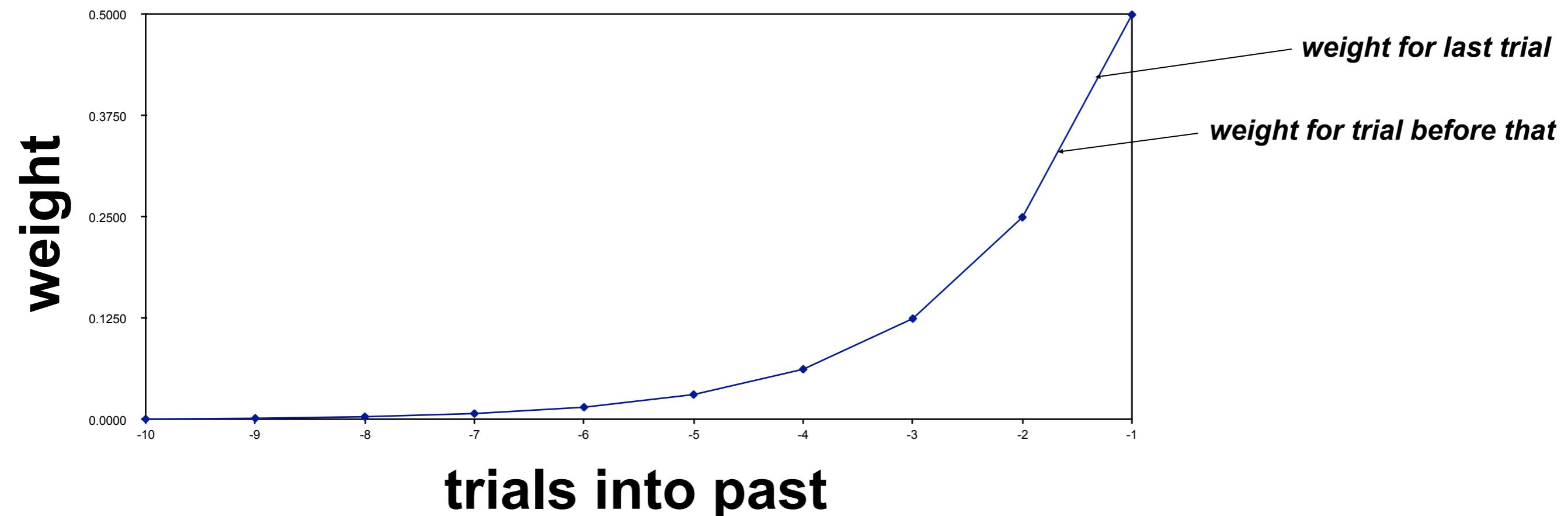
- Optimal integration of the reward history would be incapable of this quick adaptation
- Instead, suggest some “leaky integrator” which is pooling information in time (recency biased)



- Distinct from traditional matching law which integrates “globally”

Matching Behavior and the Representation of Value in the Parietal Cortex

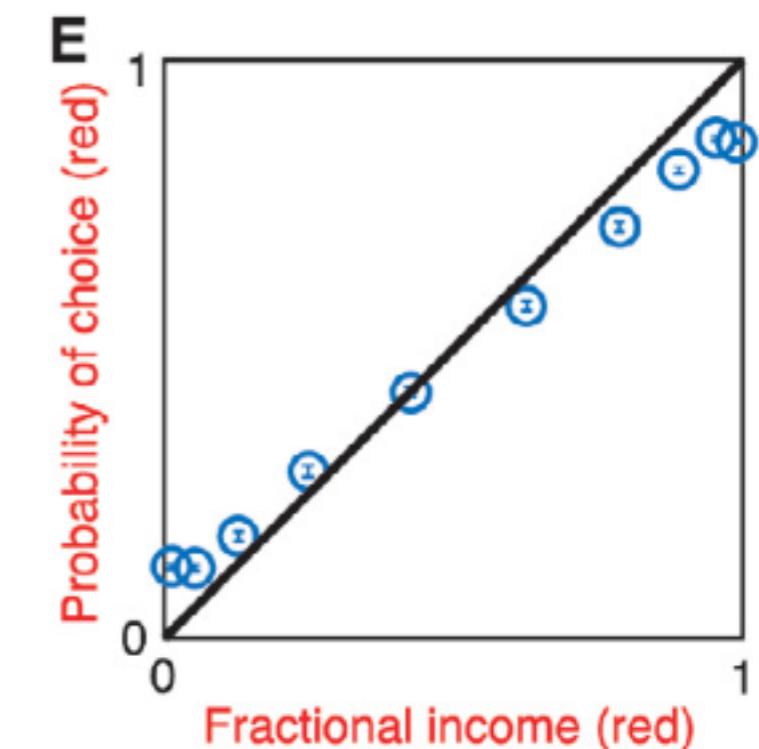
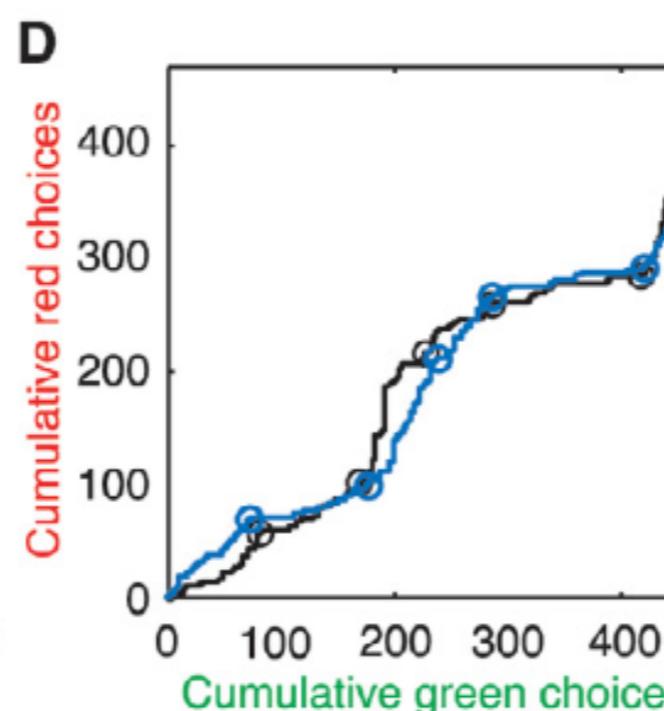
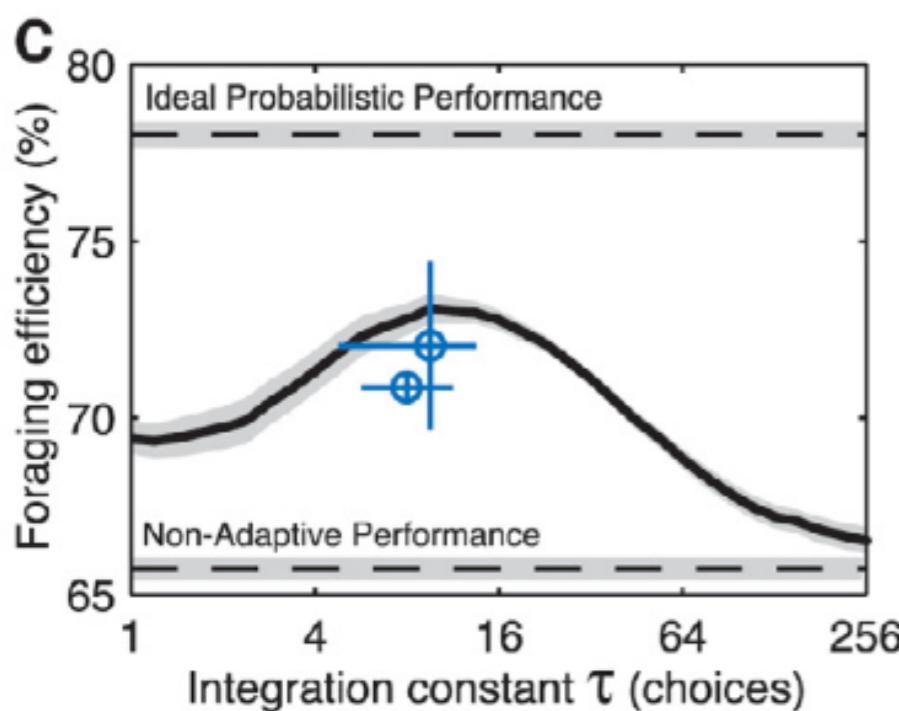
Leo P. Sugrue,* Greg S. Corrado, William T. Newsome



Matching Behavior and the Representation of Value in the Parietal Cortex

Leo P. Sugrue,* Greg S. Corrado, William T. Newsome

How do the monkeys do?



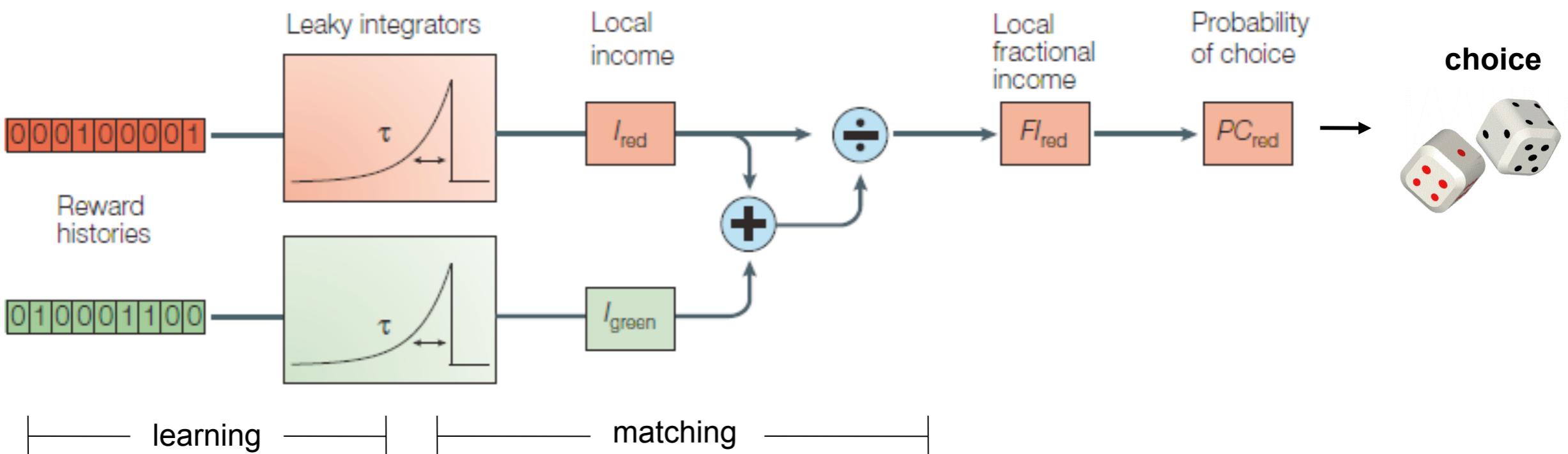
Near optimal setting of tau

The monkey's match!

Matching Behavior and the Representation of Value in the Parietal Cortex

Leo P. Sugrue,* Greg S. Corrado, William T. Newsome

The model

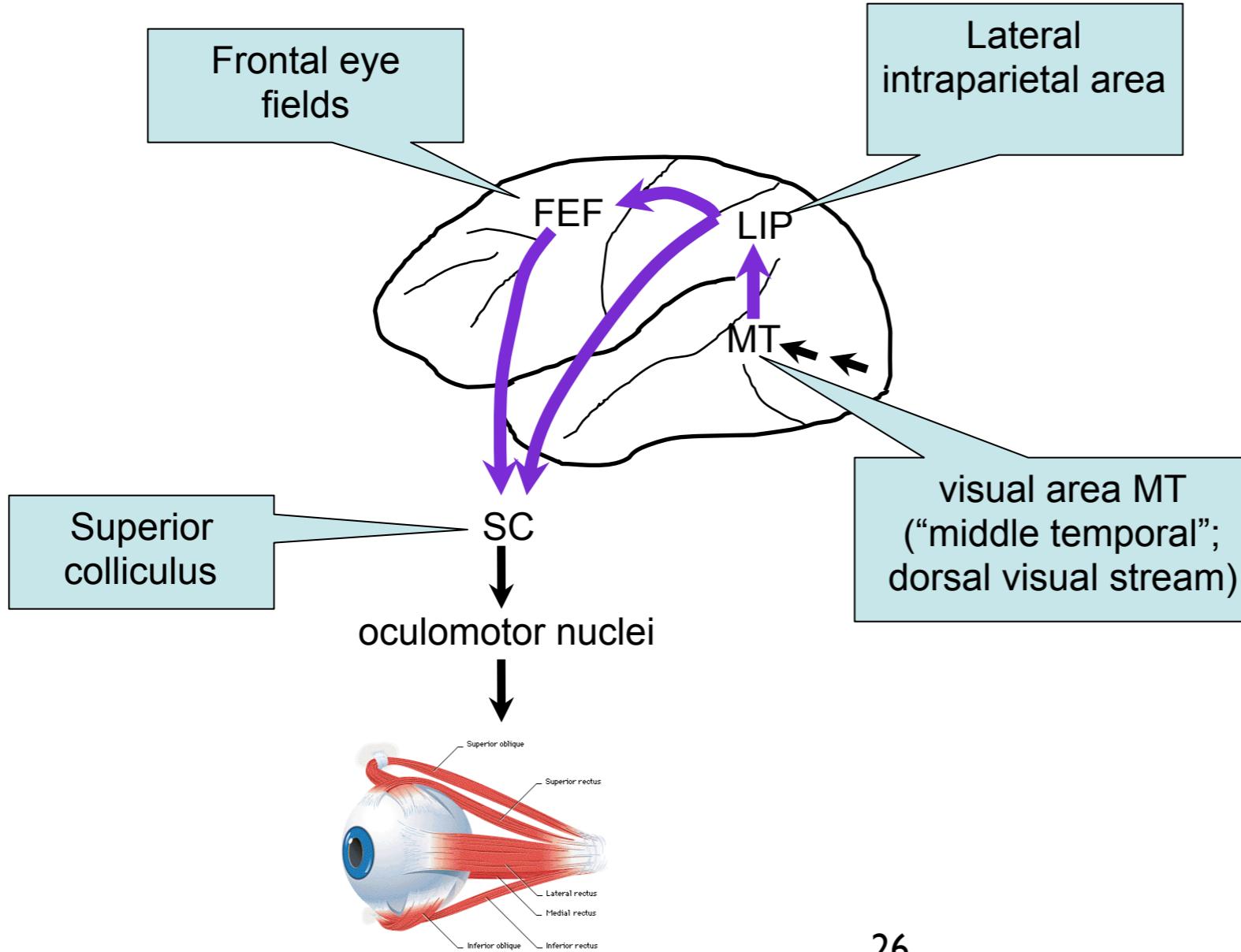


Matching Behavior and the Representation of Value in the Parietal Cortex

Leo P. Sugrue,* Greg S. Corrado, William T. Newsome

Recording from target-selective neuron is LIP

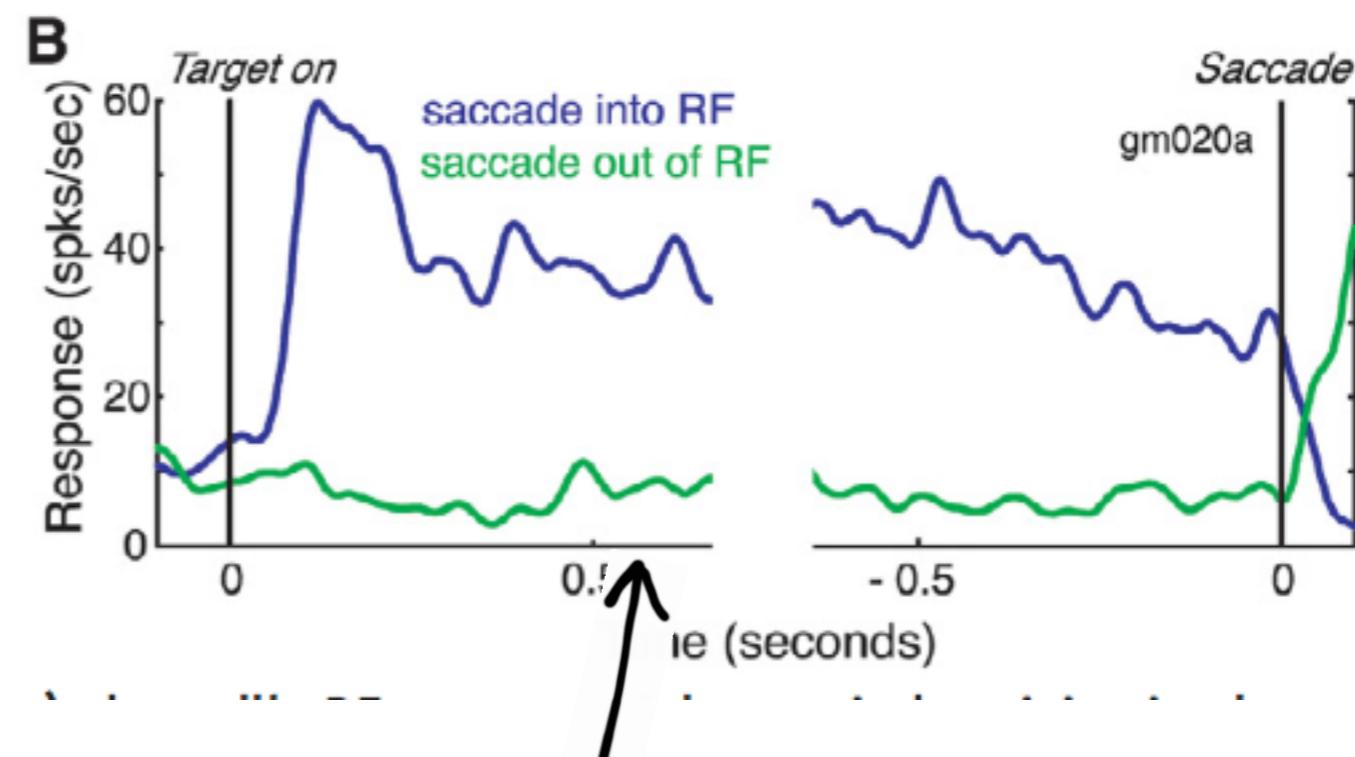
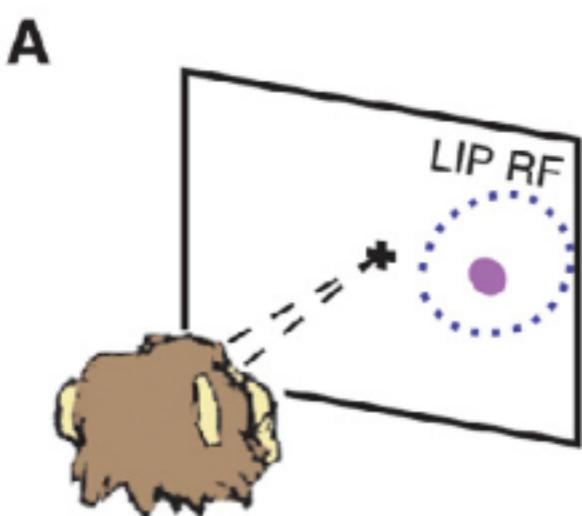
- Region involved in planning and executing saccades



Matching Behavior and the Representation of Value in the Parietal Cortex

Leo P. Sugrue,* Greg S. Corrado, William T. Newsome

Recording from target-selective neuron is LIP



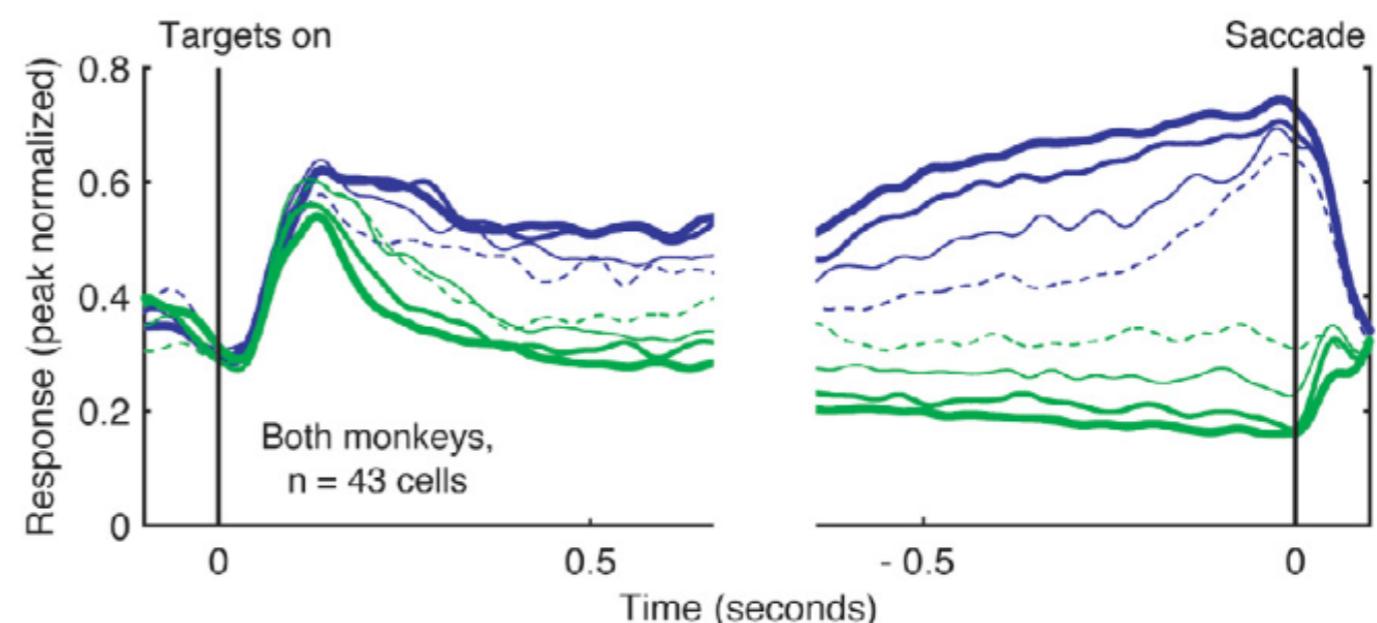
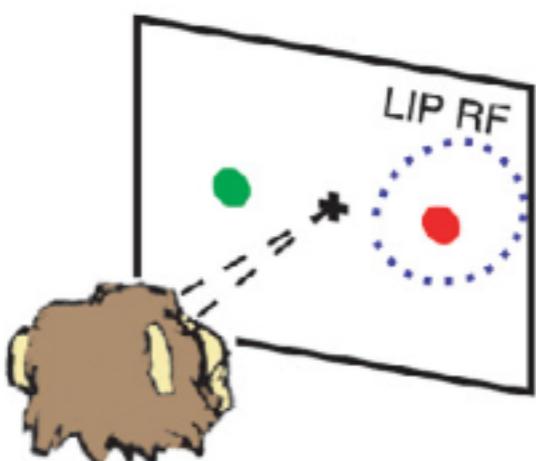
- Selected on the basis of delayed saccade task to find RF selective cells (like a forced choice version of matching game)

Spatially selective response

Matching Behavior and the Representation of Value in the Parietal Cortex

Leo P. Sugrue,* Greg S. Corrado, William T. Newsome

Recording from target-selective neuron is LIP

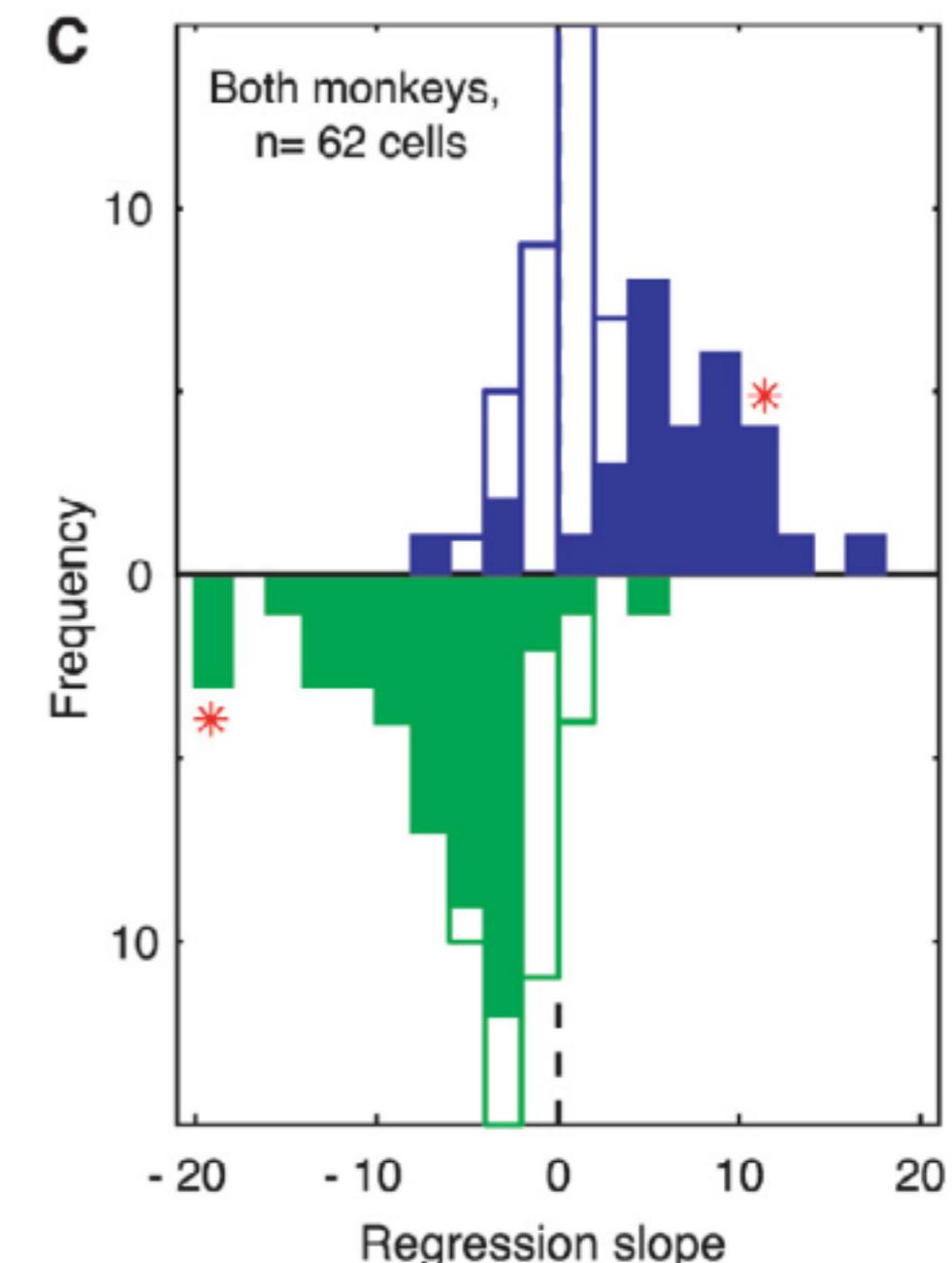
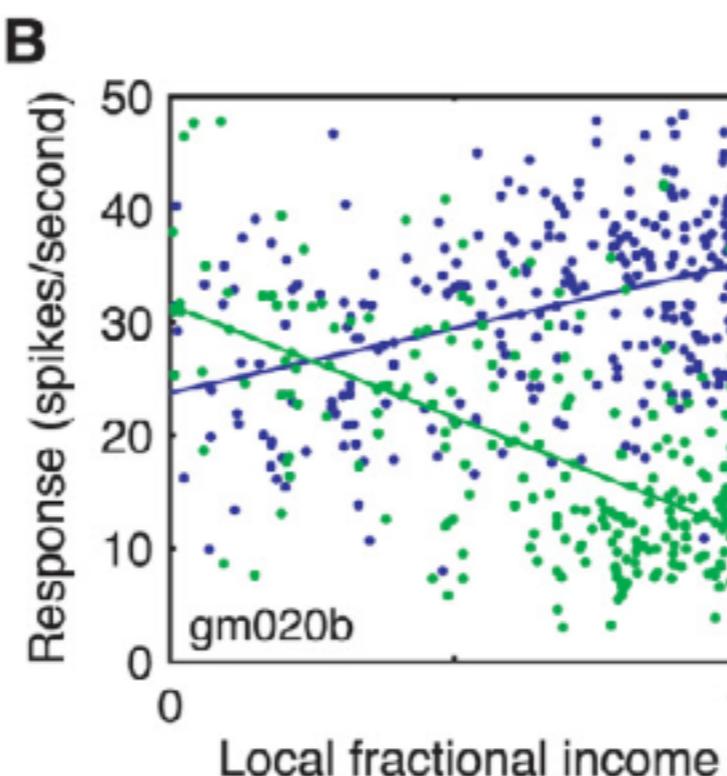
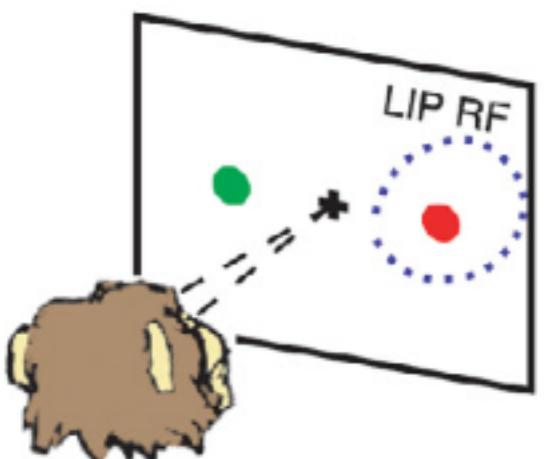


- Cells modulated by trial-to-trial utility of target in the receptive field

Matching Behavior and the Representation of Value in the Parietal Cortex

Leo P. Sugrue,* Greg S. Corrado, William T. Newsome

Recording from target-selective neuron is LIP



- Cells modulated by trial-to-trial utility of target in the receptive field

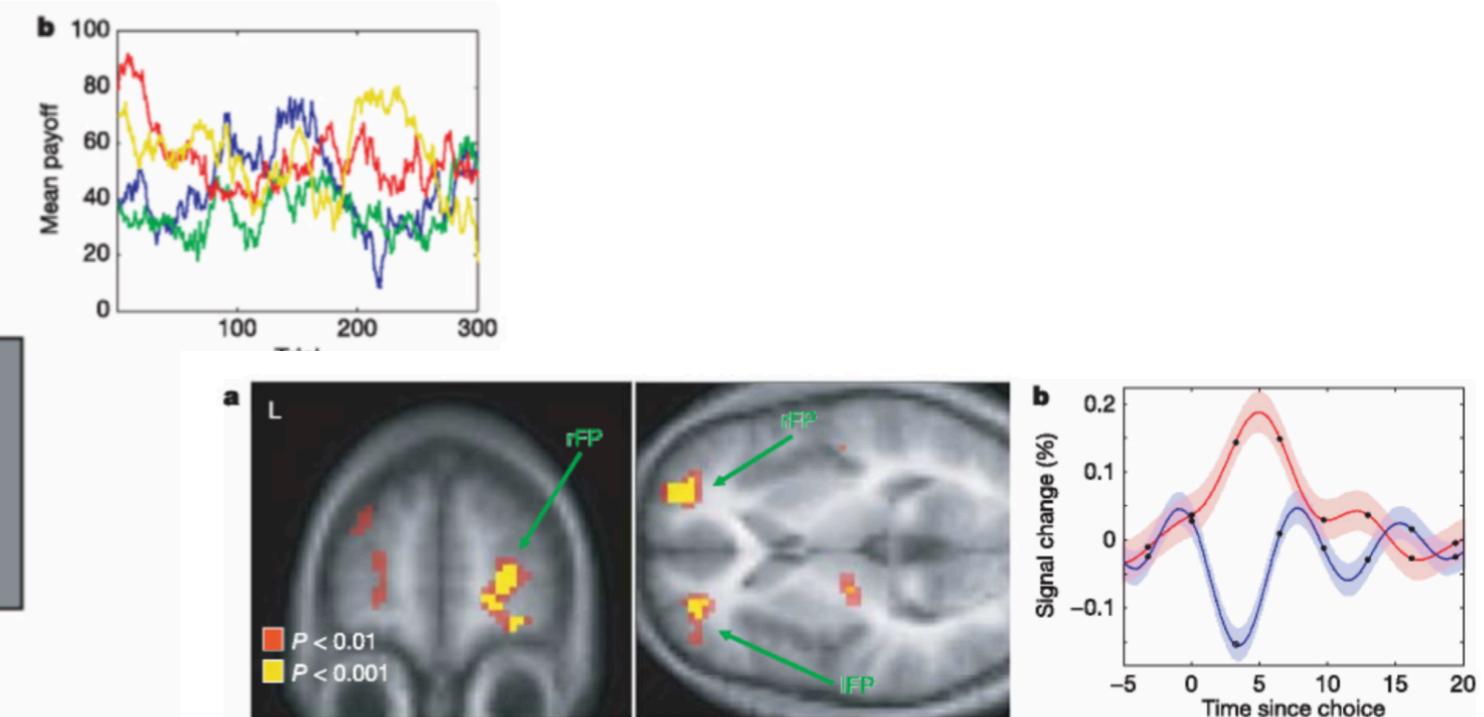
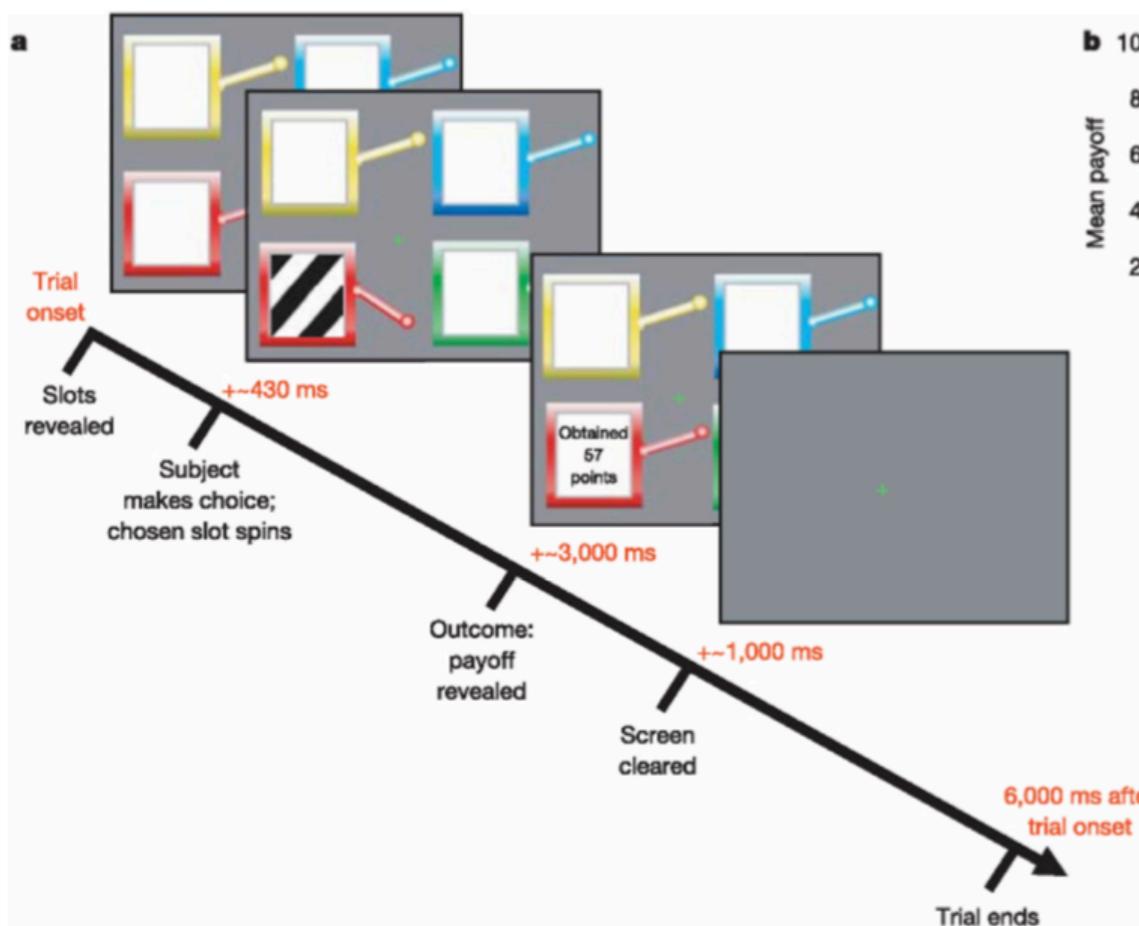


Figure 3.
Exploration-related activity in frontopolar cortex. a, Regions of left and right frontopolar cortex (IFP, rFP) showing significantly increased activation on exploratory compared with exploitative trials. Activation maps (yellow, $P < 0.001$; red, $P < 0.01$) are superimposed on a subject-averaged structural scan. The coordinates of activated areas are [−27, 48, 4, peak $z = 3.49$] for IFP and [27, 57, 6, peak $z = 4.13$] for rFP. b, rFP BOLD time courses averaged over 1,515 exploratory (red line) and 2,646 exploitative (blue line) decisions. Black dots indicate the sampling frequency (although, because sample alignment varied from trial to trial, time courses were upsampling). Coloured fringes show error bars (representing s.e.m.).

Published in final edited form as:

Nature. 2006 June 15; 441(7095): 876–879. doi:10.1038/nature04766.

Cortical substrates for exploratory decisions in humans

Nathaniel D. Daw, John P. O'Doherty, Peter Dayan, Ben Seymour, and Raymond J. Dolan

Tolman, Latent Learning, and Cognitive Maps

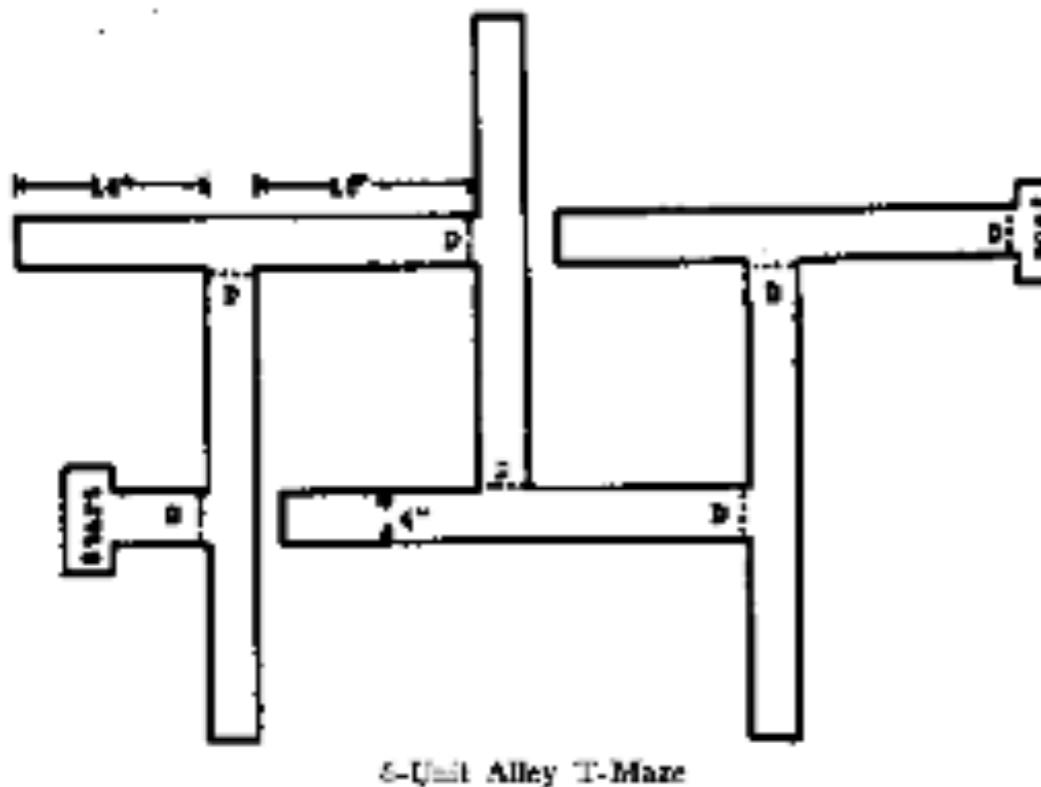
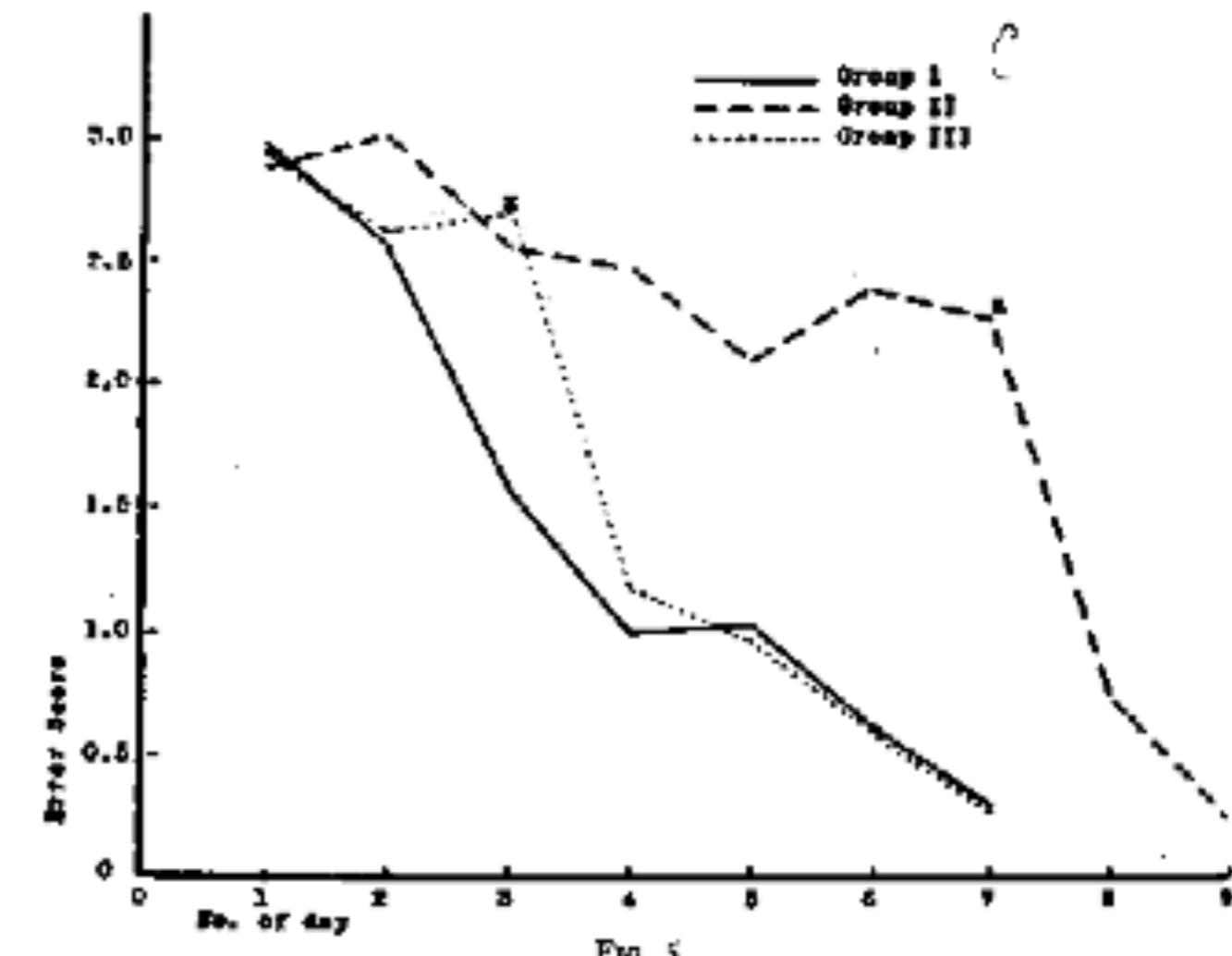


FIG. 4

(From H. C. Blodgett, The effect of the introduction of reward upon the maze performance of rats. *Univ. Calif. Publ. Psychol.*, 1929, 4, No. 8, p. 117.)

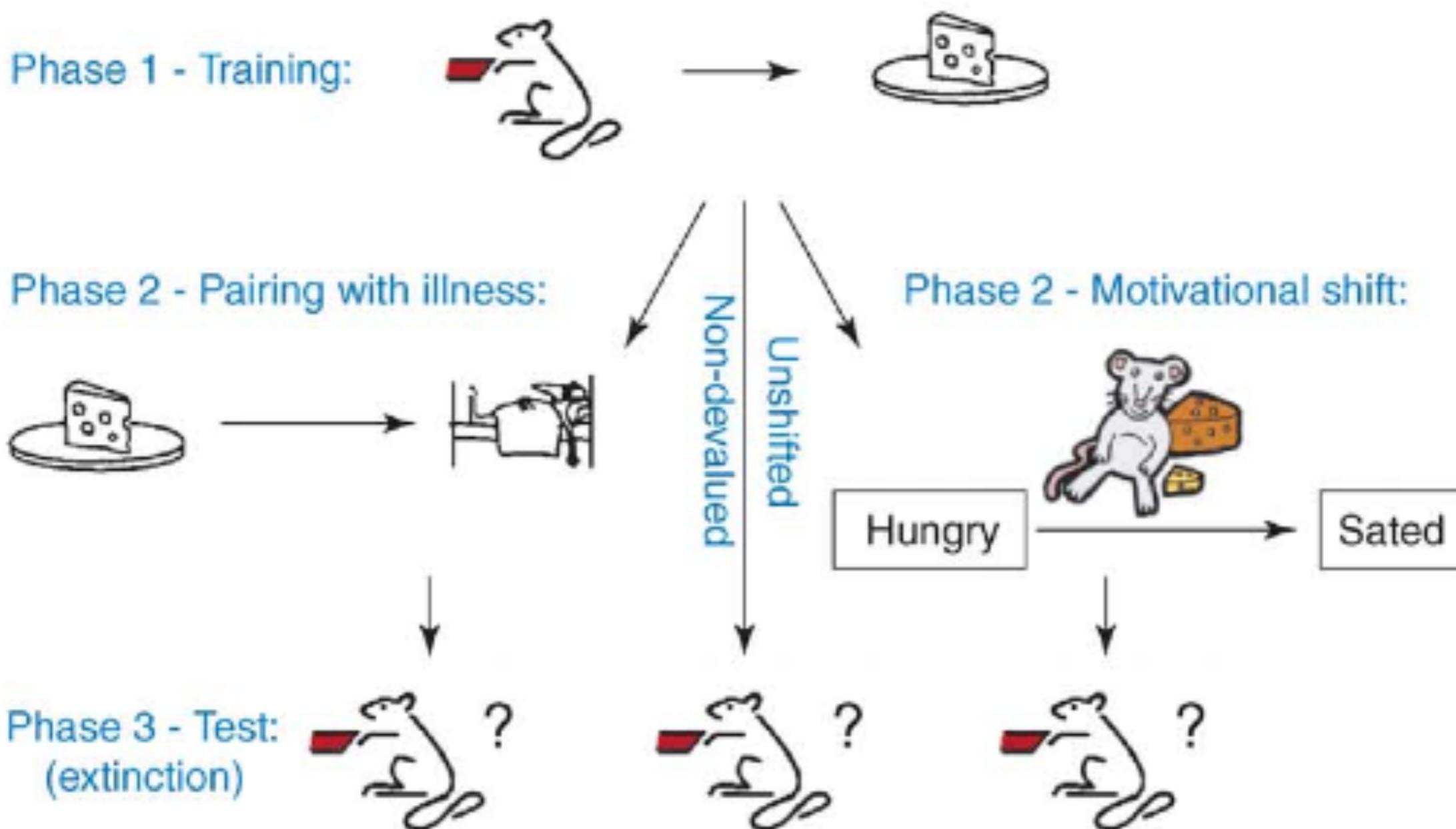


(From H. C. Blodgett, The effect of the introduction of reward upon the maze performance of rats. *Univ. Calif. Publ. Psychol.*, 1929, 4, No. 8, p. 120.)

Do animals plan?

- S-R theory:
 - Parsimonious - same theory for Pavlovian conditioning (CS/CR pairing)
 - But it overlook the critical contingency in instrumental conditioning between the response and outcome
- alternative: R-O theory (or A-O)
 - proponents: Rescorla, Dickinson, et al.
 - same spirit as Tolman (know ‘map’ of contingencies and desires, and can use this to plan... maybe INFORMATION PROCESSING!)

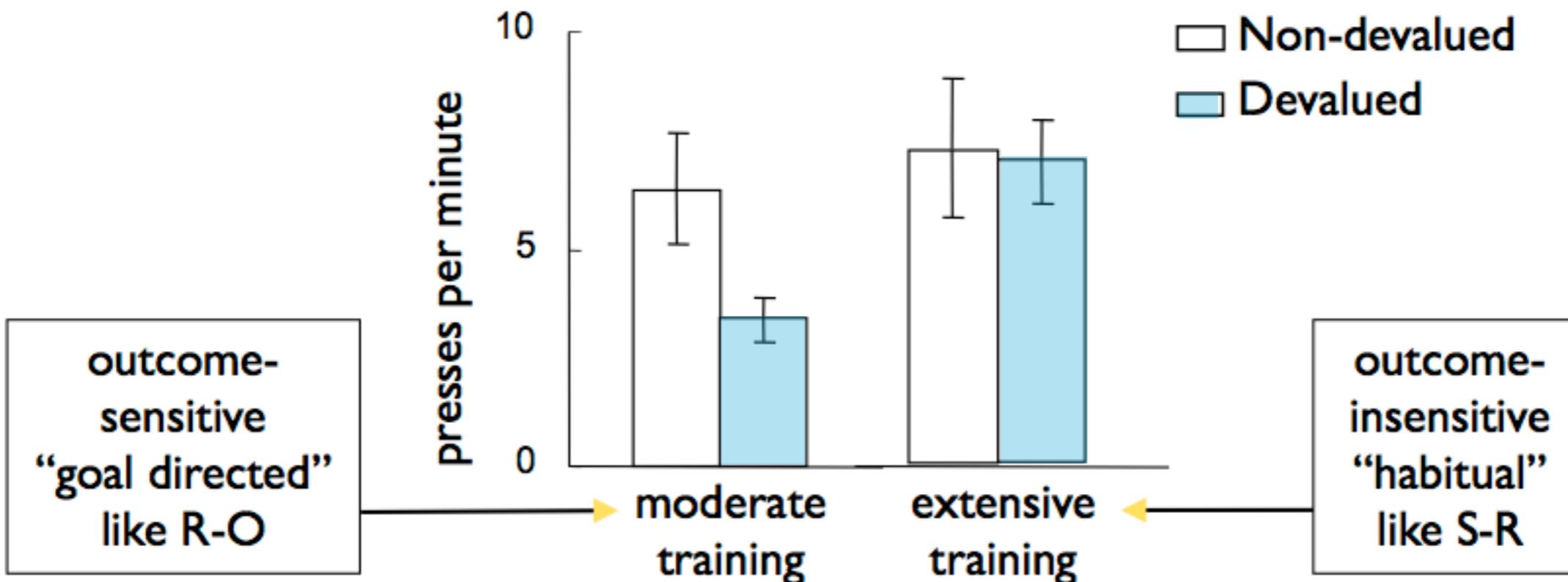
A good test: Outcome devaluation



TRENDS in Cognitive Sciences

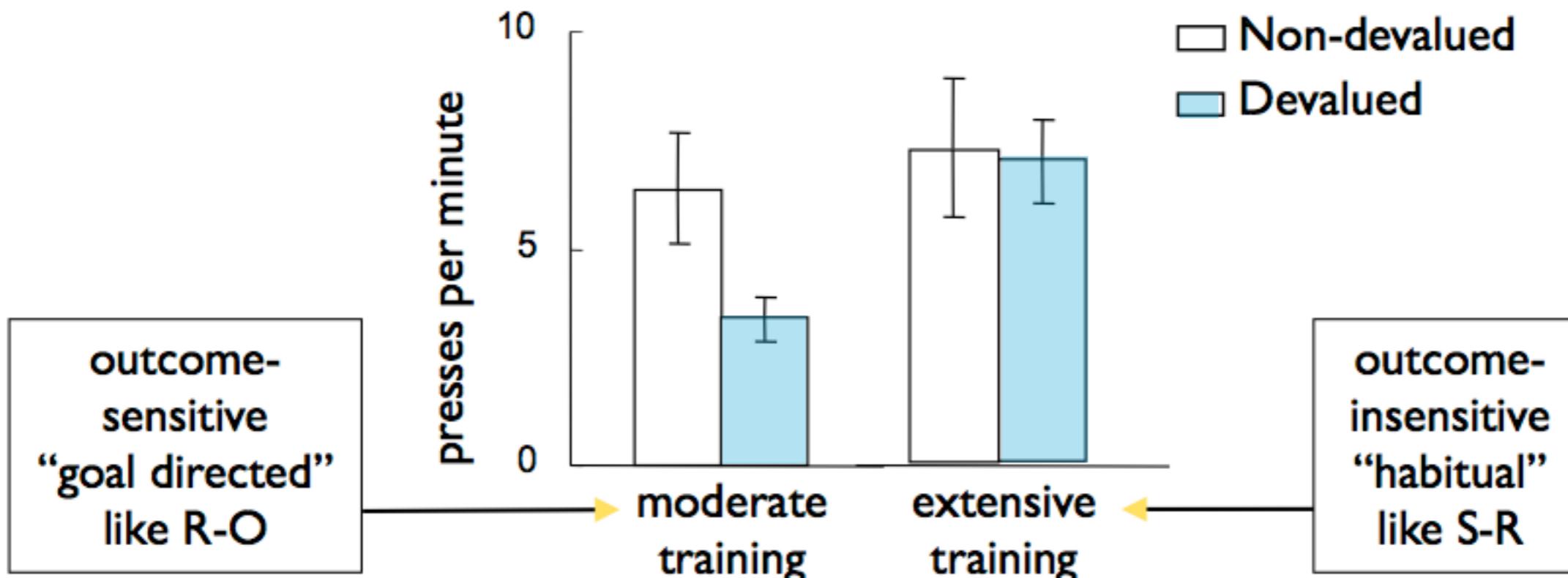
A good test: Outcome devaluation

Holland (2004) see also Dickinson article for examples



A good test: Outcome devaluation

Holland (2004) see also Dickinson article for examples



Suggests some measure of reward or outcome **expectancy** (shifts to habitual control with extensive training)

Goal directed

vs.

Habitual Behavior

“Cognitive”

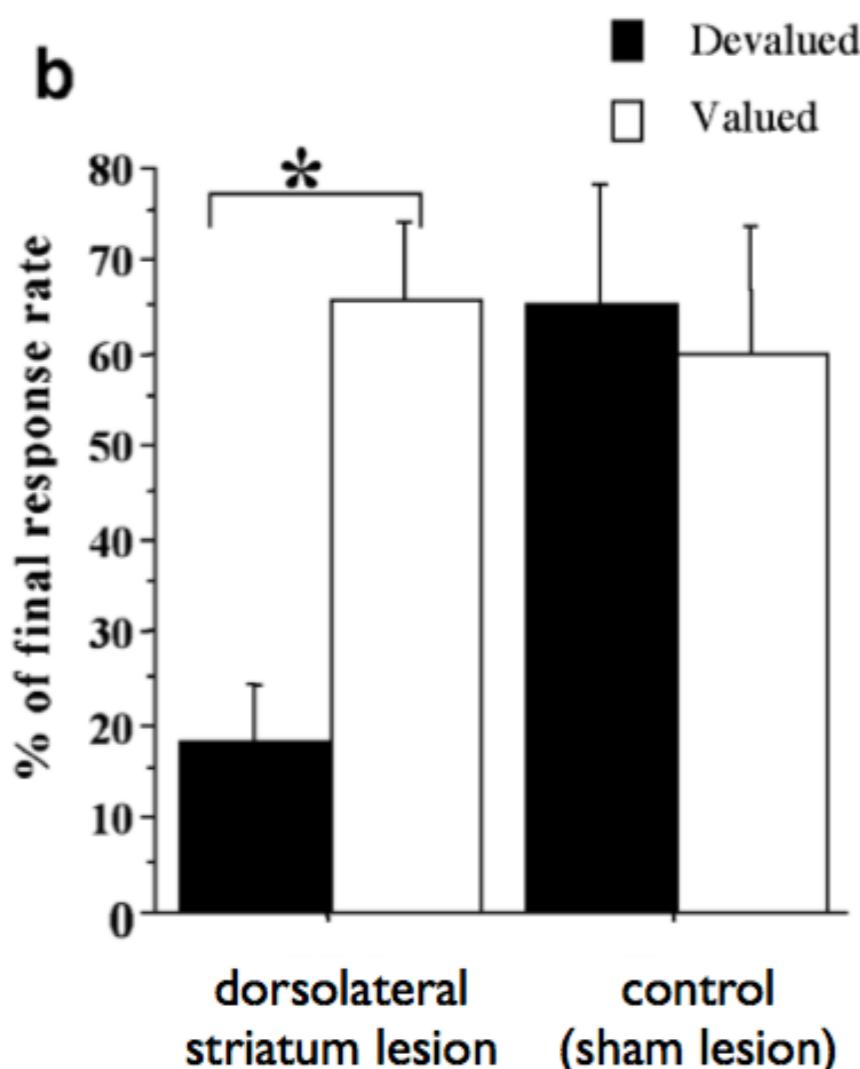
vs.

“Associationist”

Devaluation and Lesions

- Systematic patterns of behavior in devaluation studies suggest distinct neural pathways involved in either the goal-directed or habitual system

overtrained rats

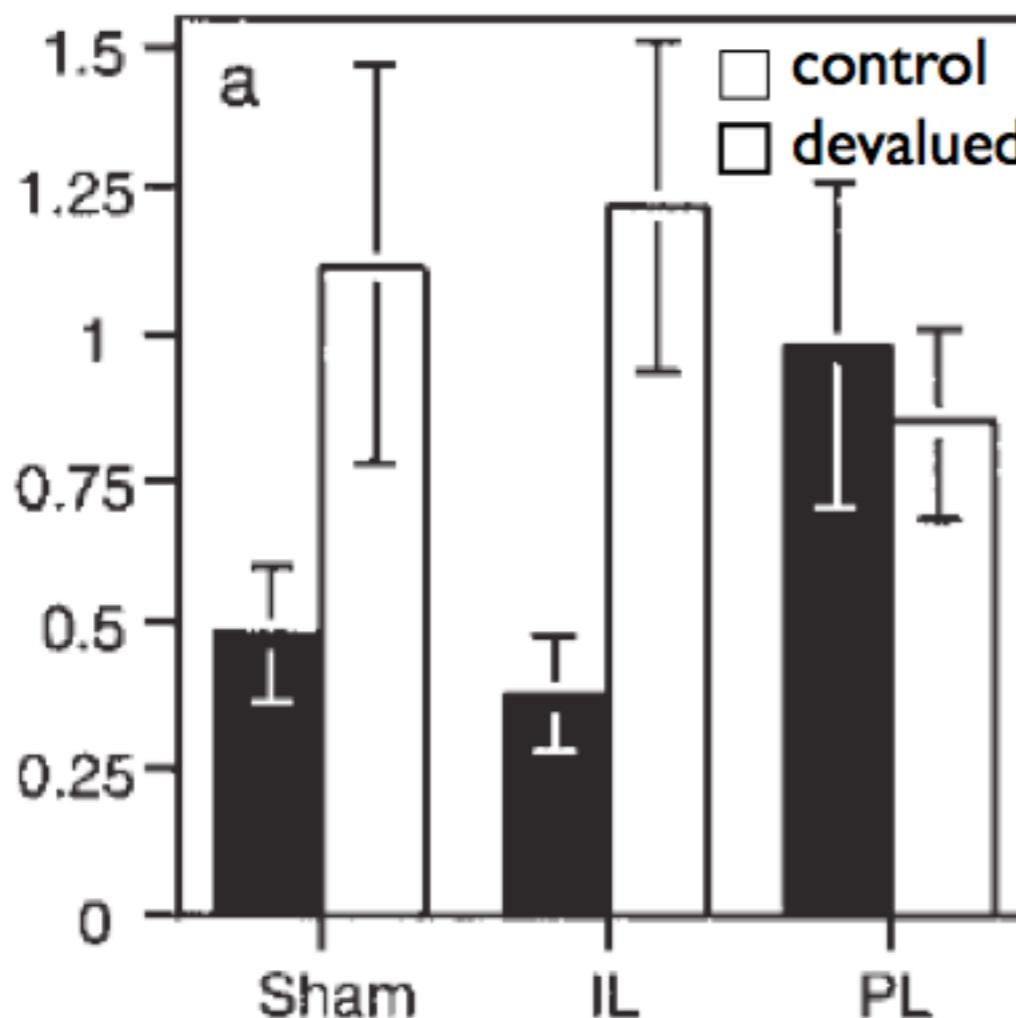


- Animals with lesion to Dorsolateral Striatum (DLS) *never develop habits* despite extensive training
- Also happens with drugs that deplete dopamine in DLS
- Also happens with lesions to infralimbic division of PFC (same corticostriatal loop)

Devaluation and Lesions

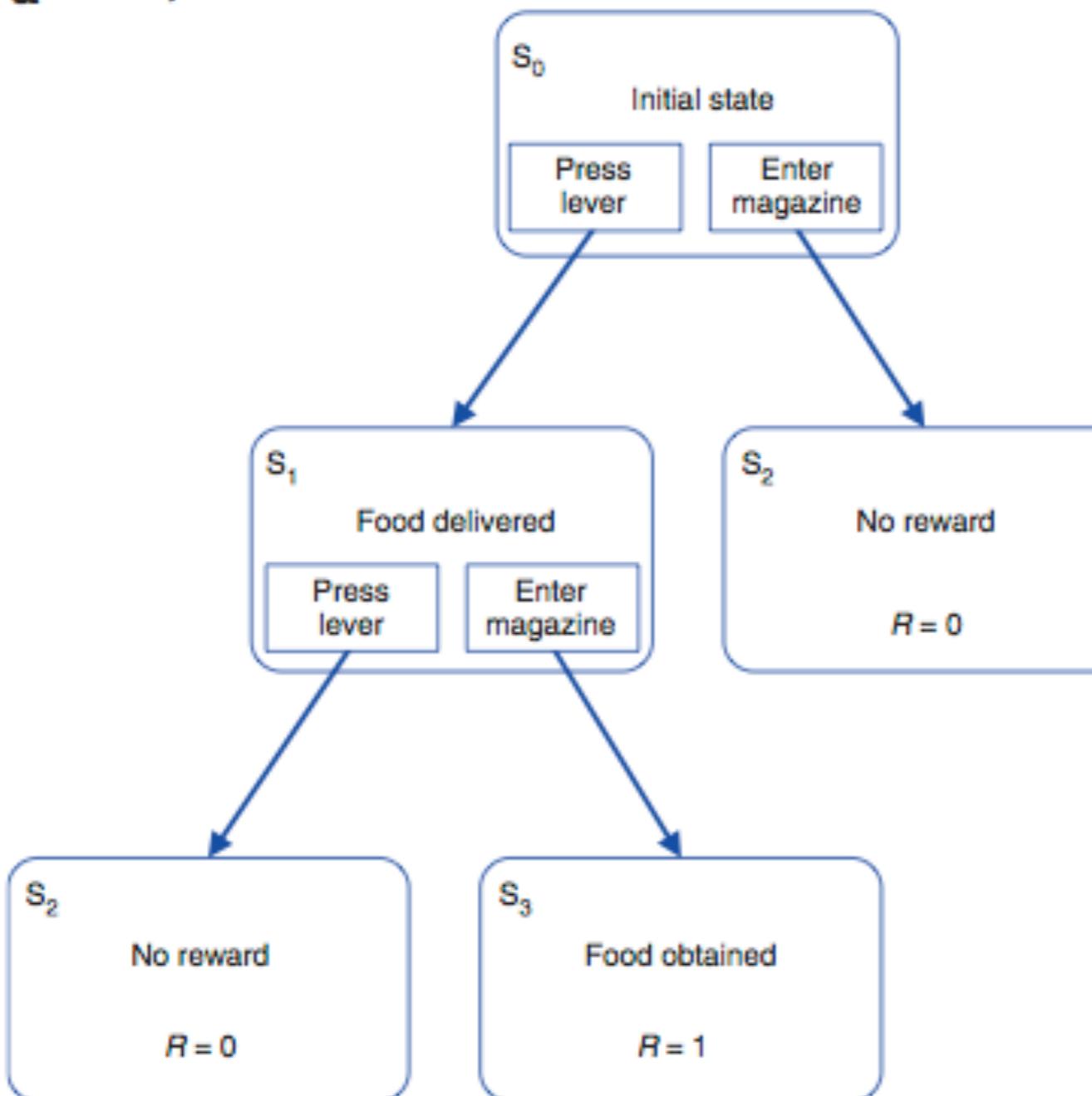
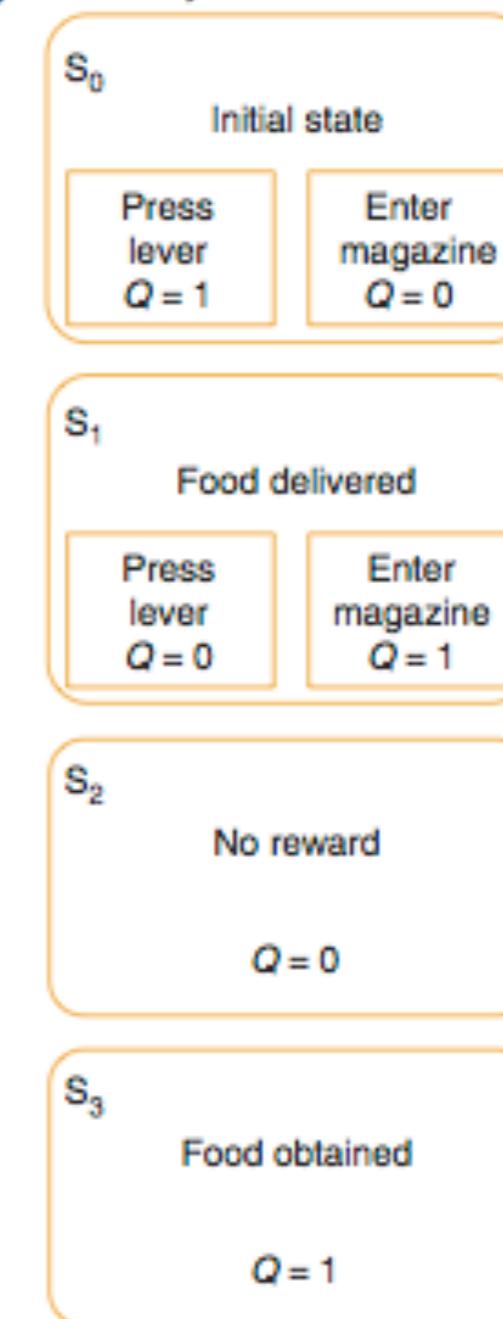
- Systematic patterns of behavior in devaluation studies suggest distinct neural pathways involved in either the goal-directed or habitual system

moderate training



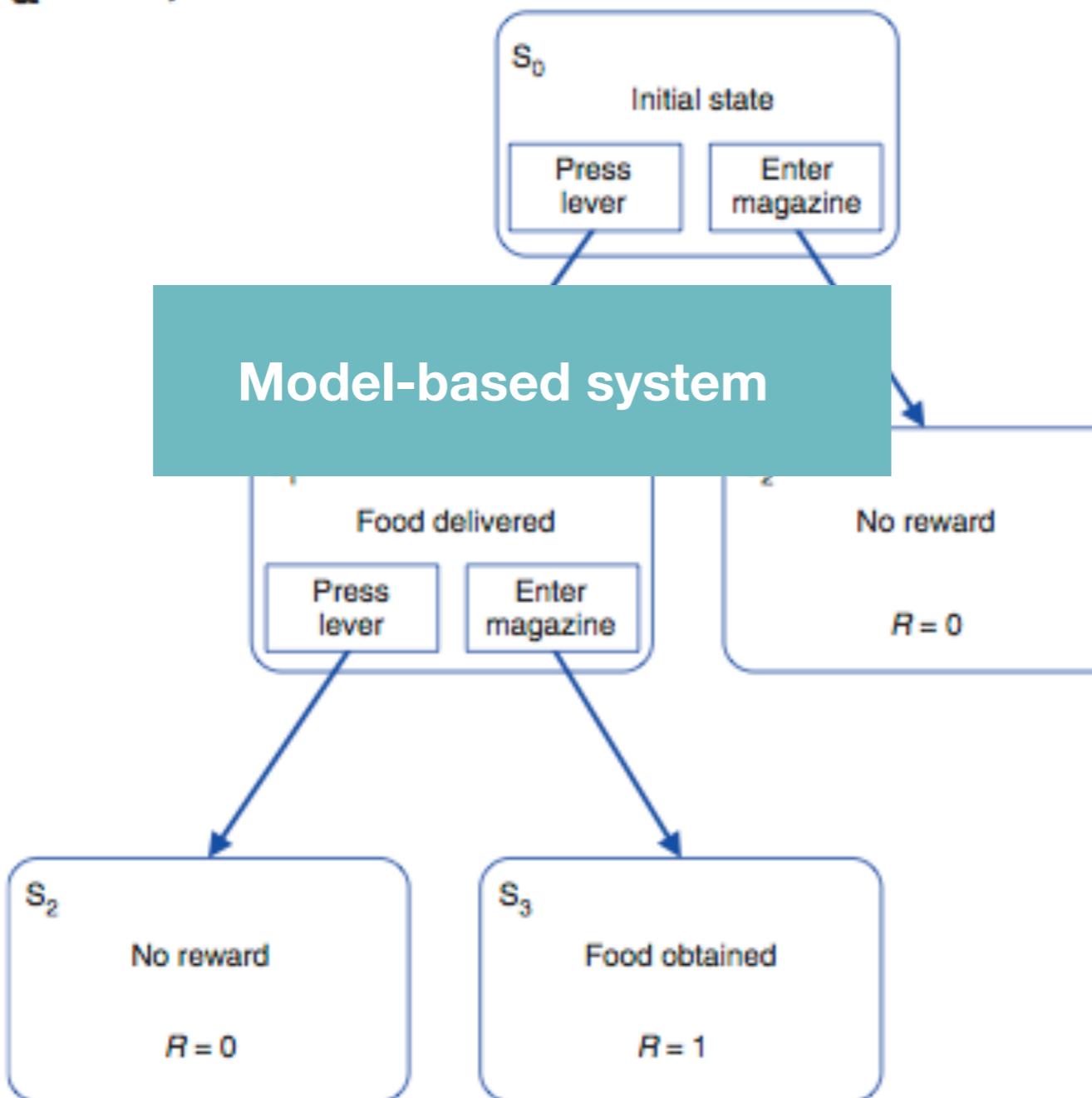
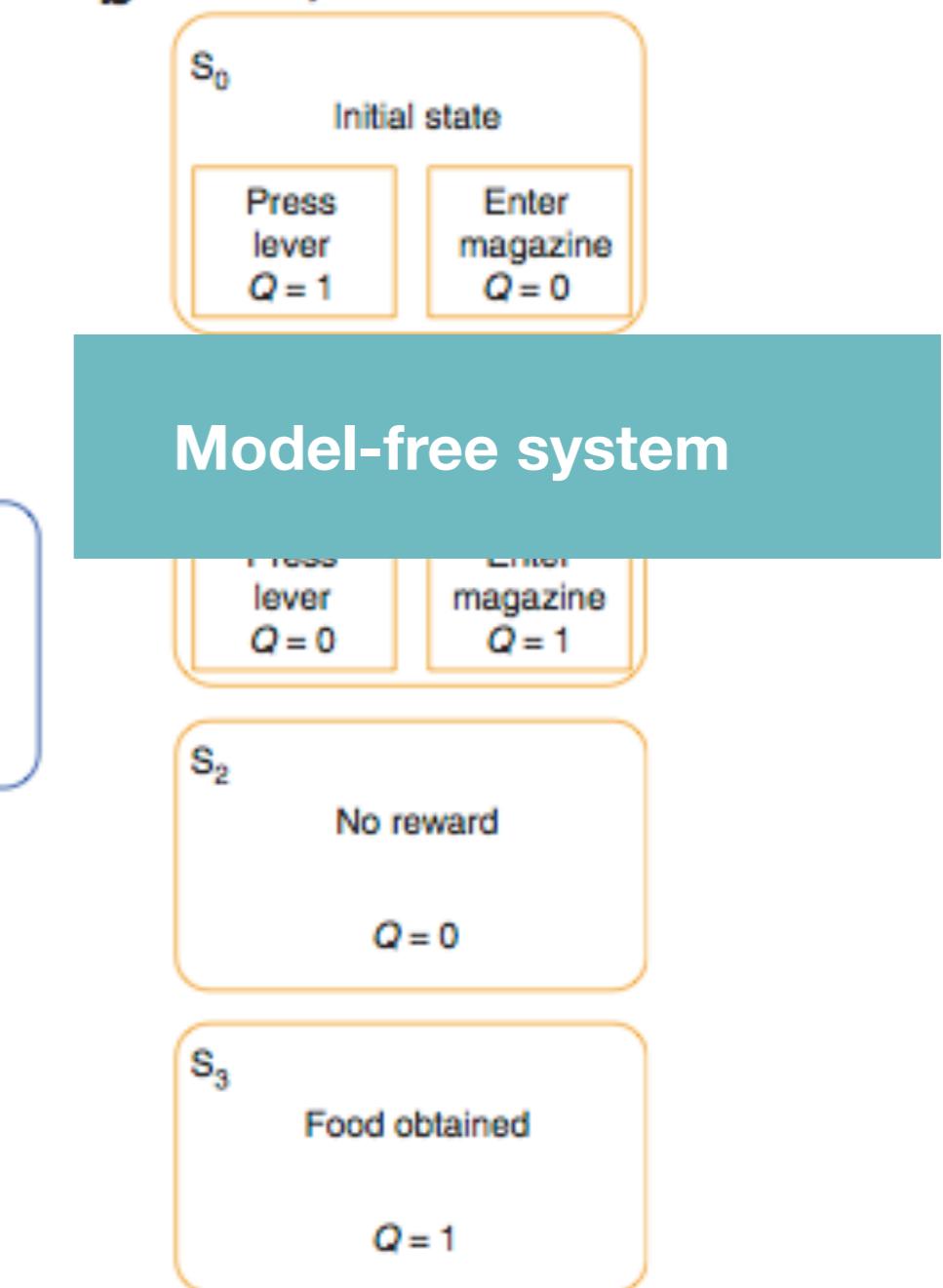
- Prelimbic (PL) PFC lesions cause animals to leverpress *habitually* even with only moderate training (along with dmPFC and mediodorsal thalamus which are in same loop).

Key idea

a Tree System**b** Cache system

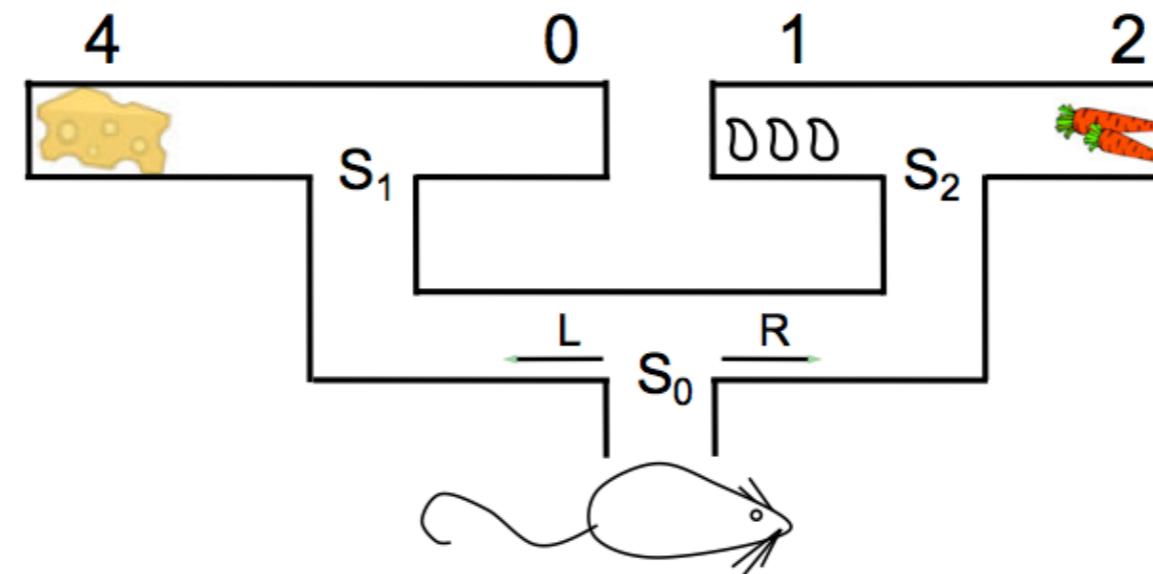
from Daw, Niv, Dayan, 2005

Key idea

a Tree System**b** Cache system

from Daw, Niv, Dayan, 2005

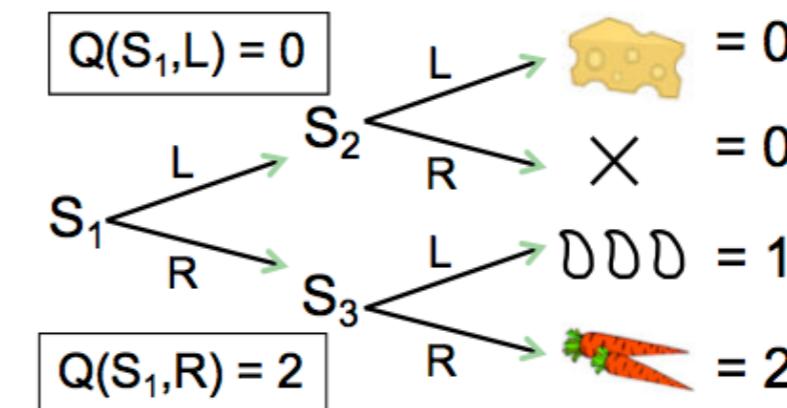
strategy I: model based RL



learn model of task through experience
 (= cognitive map)

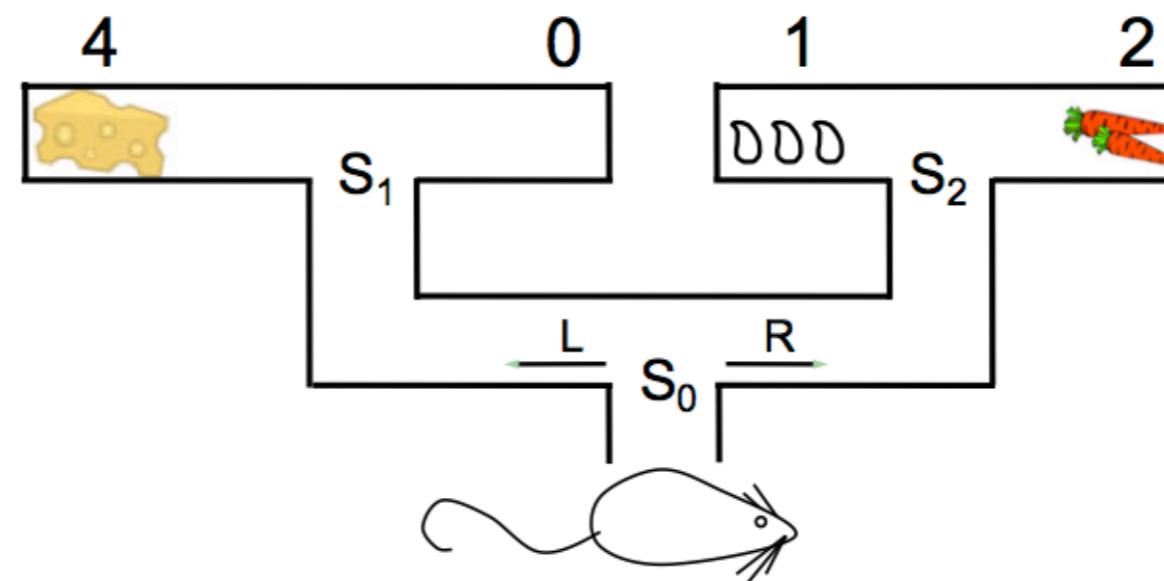
compute Q values by “looking ahead” in
the map

computationally costly, but also flexible
(immediately sensitive to change)



from Yael Niv

strategy II: model free RL



- Shortcut: store long-term values
 - then simply retrieve them to choose action
- Can learn these from experience
 - without building or searching a model
 - incrementally through prediction errors
 - dopamine dependent SARSA/Q-learning or Actor/Critic

Stored:

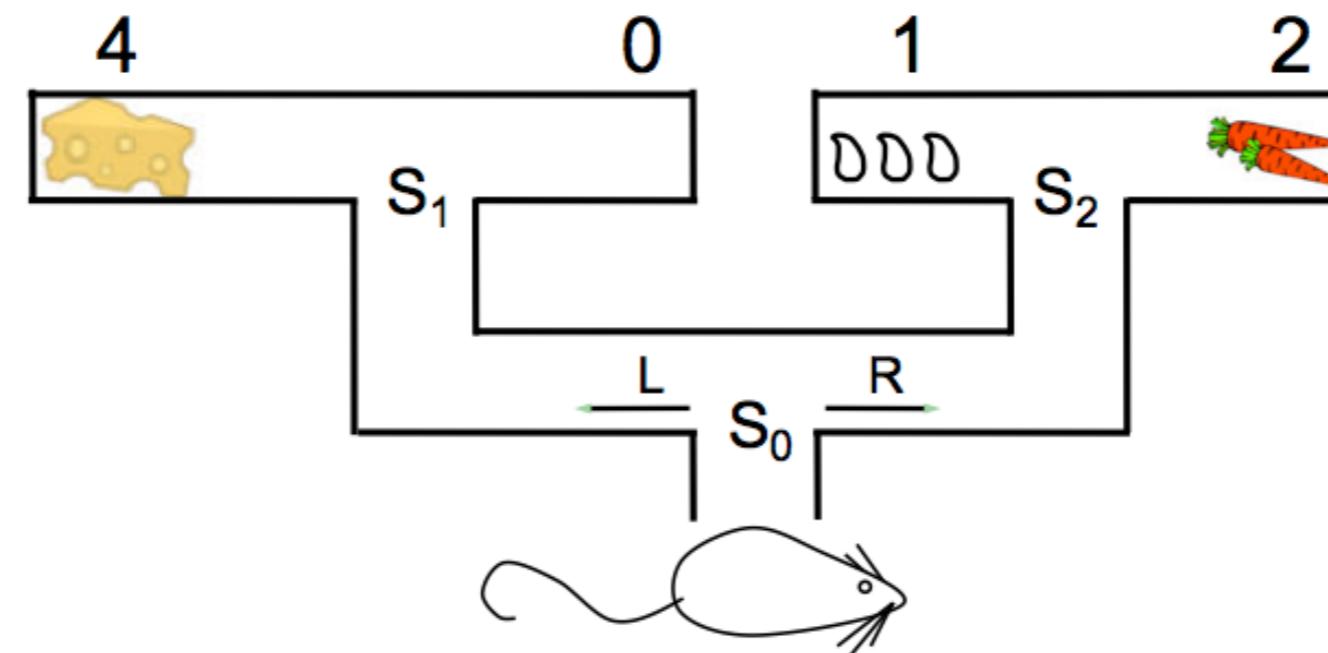
$$\begin{aligned} Q(S_0, L) &= 4 \\ Q(S_0, R) &= 2 \end{aligned}$$

$$\begin{aligned} Q(S_1, L) &= 4 \\ Q(S_1, R) &= 0 \end{aligned}$$

$$\begin{aligned} Q(S_2, L) &= 1 \\ Q(S_2, R) &= 2 \end{aligned}$$

from Yael Niv

strategy II: model free RL



- choosing actions is easy so behavior is quick, reflexive (S-R)
- but needs a lot of experience to learn
- and inflexible, need relearning to adapt to any change (habitual)

Stored:

$$\begin{aligned} Q(S_1, L) &= 4 \\ Q(S_1, R) &= 0 \end{aligned}$$

$$\begin{aligned} Q(S_0, L) &= 4 \\ Q(S_0, R) &= 2 \end{aligned}$$

$$\begin{aligned} Q(S_2, L) &= 1 \\ Q(S_2, R) &= 2 \end{aligned}$$

from Yael Niv

two big questions

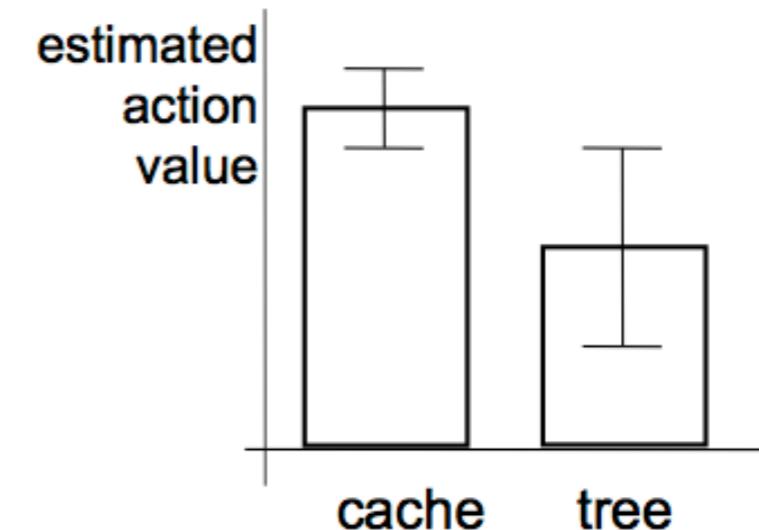
- Why should the brain use two different strategies/controllers in parallel?
- If it uses two: how can it arbitrate between the two when they disagree (*new decision making problem...*)



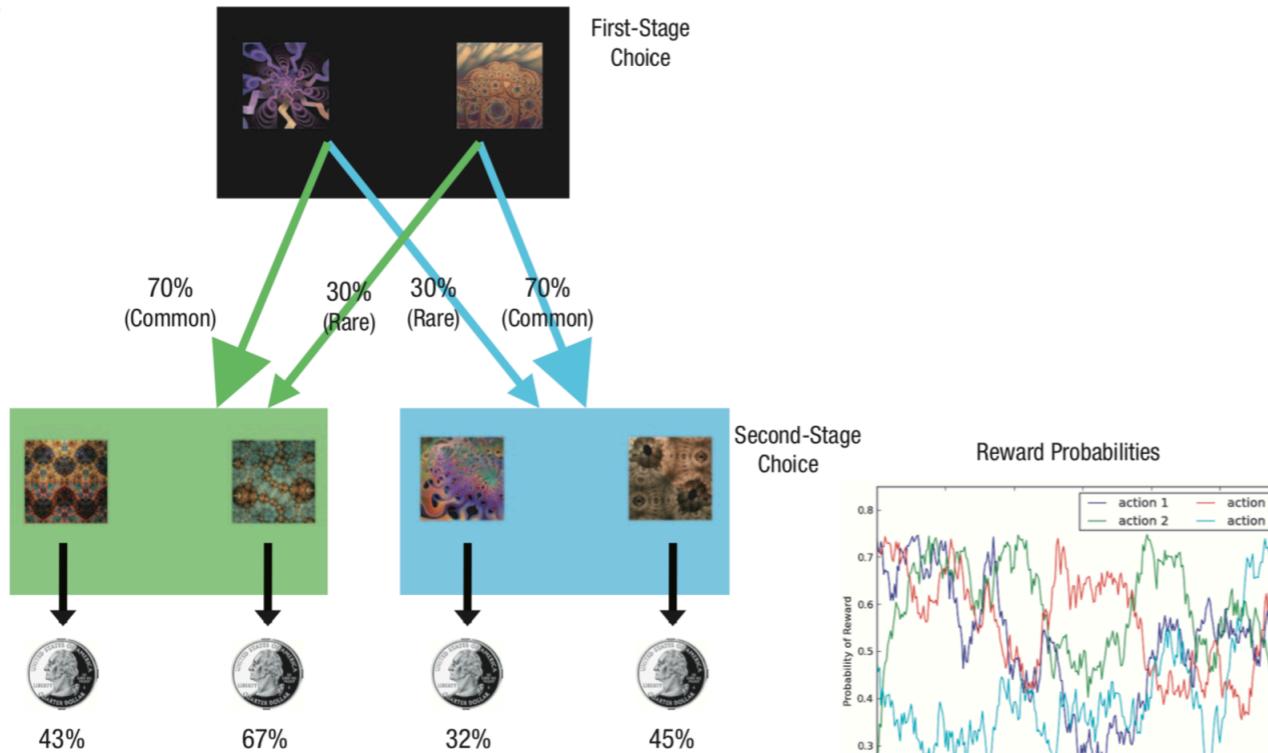
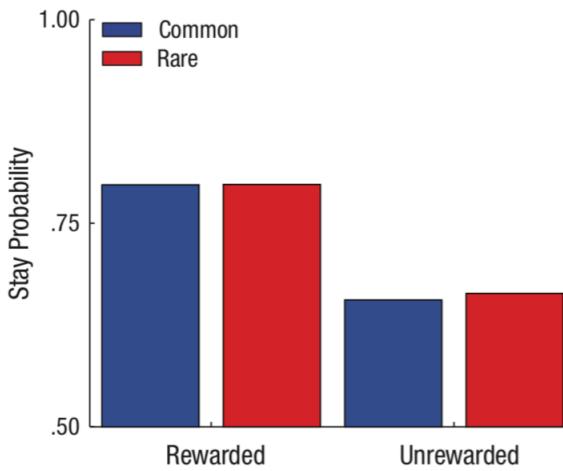
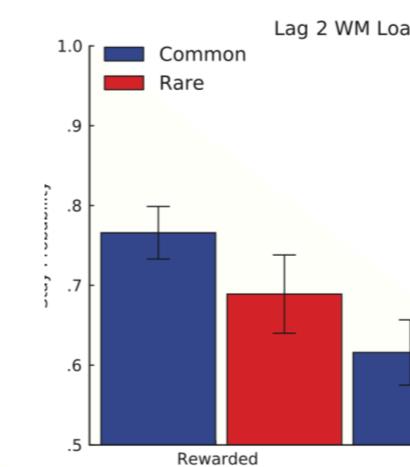
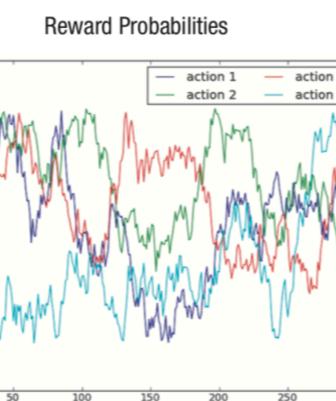
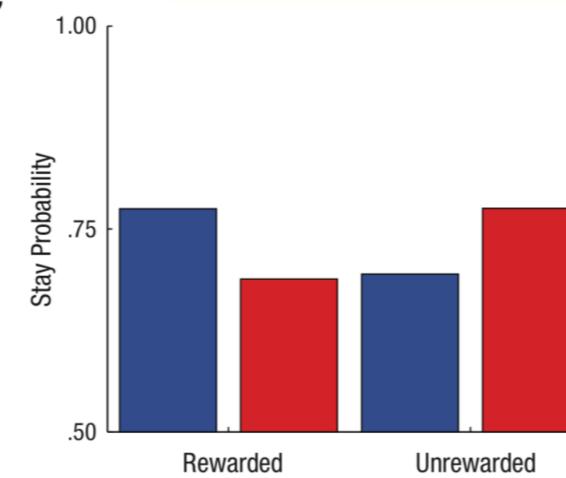
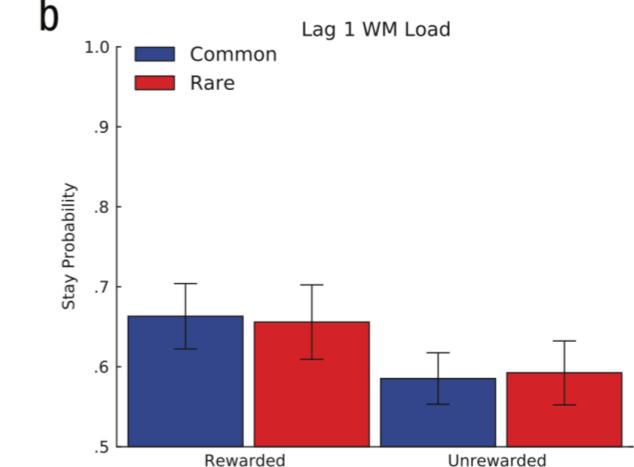
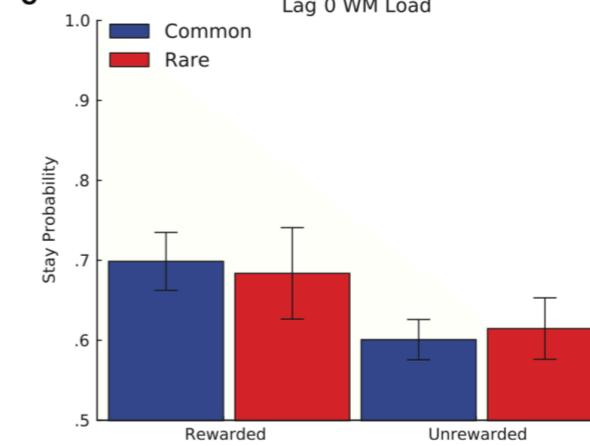
from Yael Niv

answers

- each system is best in different situations (use each one when it is most suitable/most accurate)
 - goal-directed (forward search) - good with limited training, close to the reward (don't have to search ahead too far)
 - habitual (cache) - good after much experience, distance from reward not so important
- arbitration: trust the system that is more confident in its recommendation
 - different sources of uncertainty in the two systems
 - don't always choose the highest value
 - uncertainty is different from risk



from Yael Niv

a**b****c****b****c**

Working memory load interferes with online planning making human act more like model-free learners.

The Curse of Planning: Dissecting Multiple Reinforcement-Learning Systems by Taxing the Central Executive

A. Ross Otto¹, Samuel J. Gershman², Arthur B. Markman¹, and Nathaniel D. Daw³

¹Department of Psychology, University of Texas at Austin; ²Department of Psychology and Princeton Neuroscience Institute, Princeton University; and ³Department of Psychology and Center for Neural Science, New York University

Three levels of description (*David Marr, 1982*)

Computational

Why do things work the way they do?
What is the goal of the computation?
What are the unifying principles?



Algorithmic

What representations can implement such computations?
How does the choice of representations determine the algorithm?

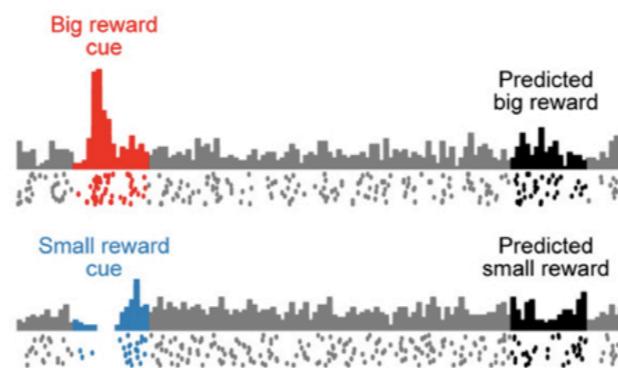
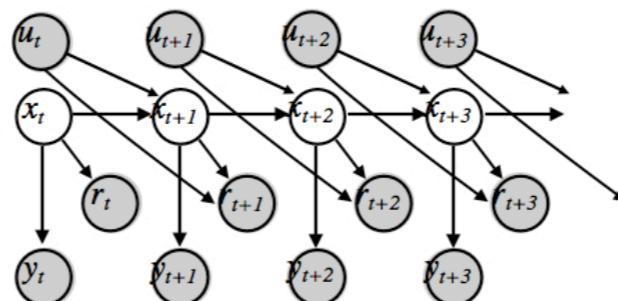
Implementational

How can such a system be built in hardware?
How can neurons carry out the computations?

maximize:

$$R_t = r_{t+1} + r_{t+2} + \dots + r_T$$

Bellman



Dynamic programming,
TD methods, Monte
Carlo

Neural firing patterns,
prediction errors,
system level
neuroscience

Next time: Bayes



Slide Credits

Nathaniel Daw (exploration/gittins)
Alex Rich
Gillian Hayes (TD methods/explore)
Rich Sutton (general approach)
Andy Barto (general approach)