
Is prioritized sweeping the better episodic control?

Johanni Brea¹

Abstract

Episodic control has been proposed as a third approach to reinforcement learning, besides model-free and model-based control, by analogy with the three types of human memory. i.e. episodic, procedural and semantic memory. But the theoretical properties of episodic control are not well investigated. Here I show that in deterministic tree Markov decision processes, episodic control is equivalent to a form of prioritized sweeping in terms of sample efficiency as well as memory and computation demands. For general deterministic and stochastic environments, prioritized sweeping performs better even when memory and computation demands are restricted to be equal to those of episodic control. These results suggest generalizations of prioritized sweeping to partially observable environments, its combined use with function approximation and the search for possible implementations of prioritized sweeping in brains.

1. Introduction

Single experiences can drastically change subsequent decision making. Discovering a new passage in my home town, for example, may immediately affect my policy to navigate and shorten path lengths. A software developer may discover a tool that increases productivity and never go back to the old workflow. And on the level of the policy of a state, Vasco Da Gamma's discovery of the sea route to India had an almost immediate and long-lasting effect for the Portuguese.

Given the importance of such single experiences, it is not surprising that humans and some animal use an episodic(-like) memory system devoted to single experiences (Lengyel & Dayan, 2008; Clayton et al., 2007). But how exactly should episodic memory influence decision making? Lengyel &

Dayan (2008) proposed "episodic control", where

[..] each time the subject experiences a reward that is considered large enough (larger than expected a priori) it stores the specific sequence of state-action pairs leading up to this reward, and tries to follow such a sequence whenever it stumbles upon a state included in it. If multiple successful sequences are available for the same state, the one that yielded maximal reward is followed.

This form of episodic control was found to be beneficial in the initial phase of learning for stochastic tree Markov Decision Processes (tMDP) (Lengyel & Dayan, 2008) and in the domain of simple video games (Blundell et al., 2016). But it is unclear, if there are conditions under which episodic control converges to a (nearly) optimal solution and how fast it does so.

An alternative way to link episodic memory to decision making is to compute Q-values at decision time based on a set of retrieved episodes using a mechanism similar to N-step backups (Gershman & Daw, 2017). In fact, if all episodes starting in a given state-action pair are retrieved the result is equivalent to TD-learning with N-step backups and decaying learning rates. More interesting is the case where the retrieved episodes are cleverly selected, but how this selection mechanism should work exactly, remains an open question.

While speed of learning in the sense of sample efficiency is one important aspect of a learning algorithm, a second criterion is computational efficiency (Lengyel & Dayan, 2008). Without this second desideratum, the canonical choice would be model-based reinforcement learning, where the agent additionally learns how its perception of the environment changes in response to actions and how reward depends on state-action pairs. Model-based reinforcement learning is well known for high sample efficiency (Sutton & Barto, 2018). But it is often considered to be computationally challenging. In fact, one demanding way to use the model is to perform a forward search before every decision, similar to chess players or board game playing algorithms (Silver et al., 2017) that plan the next moves. Instead of forward search at decision time, however, one can use backward search whenever an important discovery is

¹ School of Computer and Communication Sciences and Brain Mind Institute, School of Life Sciences, cole polytechnique fdrale de Lausanne, CH-1015 Lausanne. Correspondence to: Johanni Brea <johanni.brea@epfl.ch>.

made. Prioritized sweeping (Moore & Atkeson, 1993; Peng & Williams, 1993), implements this backward search efficiently, in particular with small backups (Seijen & Sutton, 2013). Prioritized sweeping converges to the optimal policy for arbitrary Markov Decision Processes (MDP) (Seijen & Sutton, 2013). Furthermore, for deterministic environments or at the initial phase of learning in stochastic environments, prioritized sweeping relies in a similar way on single experiences as episodic control.

In this paper I address the question of whether prioritized sweeping or episodic control are preferable in terms of sample efficiency and computational efficiency. I introduce a variant of prioritized sweeping with model reset at the end of each episode and show that it has the same computational complexity as episodic control, equivalent sample efficiency in deterministic tree Markov Decision Processes and better sample efficiency in general deterministic MDPs or stochastic MDPs. Thus, it appears as a third and promising candidate algorithm to link episodic memory to decision making.

2. Review of prioritized sweeping

Prioritized sweeping (Moore & Atkeson, 1993; Peng & Williams, 1993; Seijen & Sutton, 2013) is an efficient method for doing approximate full backups in reinforcement learning. A typical setting of reinforcement learning consists of an agent who observes state s_t , performs action a_t and receives reward r_t at each time step $t \in \mathbb{N}$. It is assumed that the state transitions and reward emissions are governed by a stationary stochastic process with probabilities $T_{sas'} = P(s_{t+1} = s' | s_t = s, a_t = a)$ and $R_{sa} = \mathbb{E}[r_t | s_t = s, a_t = a]$. The agent's goal is to find a policy $a = \pi^*(s)$ that maximizes the expected future discounted reward $V_s^\pi = \mathbb{E}[\sum_{s=0}^T \gamma^s r_s | s_0 = s, \pi]$ for each state s , where $\gamma \in [0, 1]$ is a discount factor. With Q -values $Q_{sa}^\pi = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a, \pi]$, we see that the optimal policy π^* should satisfy the Bellman equations $V_s^{\pi^*} = \max_a Q_{sa}^{\pi^*}$. Dropping for notational simplicity the conditioning on the policy and using the definition of the Q -values, this can be written as

$$Q_{sa} = R_{sa} + \gamma \sum_{s'} T_{sas'} \max_{a'} Q_{s'a'} . \quad (1)$$

From this equation, we see that a change in $Q_{s'a'}$ affects the values Q_{sa} of possible predecessor states s , if and only if $Q_{s'a'} = \max_{a''} Q_{s'a''}$ before or after the change.

In reinforcement learning the agent is assumed to not know the true values of the parameters R_{sa} and $T_{sas'}$. The idea of prioritized sweeping is to maintain an estimate of $T_{sas'}$ and update the Q -values by interaction with the environment and cleverly prioritized background planning. The observation below Equation 1 that limits the impact of Q -value

Algorithm 1 Prioritized sweeping (Seijen & Sutton, 2013)

```

1: For all  $a, s$  initialize  $Q_{sa}, V_s, U_s$ , all variables needed
   to estimate  $T_{sas'}$  and an empty priority queue  $PQueue$ 
2: for each episode do
3:   Observe  $s$ 
4:   for each step in an episode do
5:     Take action  $a$  given by policy  $\pi(Q, s)$ 
6:     Observe  $r, s'$ 
7:     Update  $Q_{sa}$  and  $T_{sas''}$  (for all  $s''$ ) given  $s, a, s', r$ 
8:      $V_s \leftarrow \max_b Q_{sb}$ 
9:     Add  $s$  to  $PQueue$  with some priority  $p$ 
10:     $s \leftarrow s'$ 
11:    loop {for some steps or on background thread}
12:      Remove highest priority state  $s^*$  from  $PQueue$ 
13:       $\Delta V \leftarrow V_{s^*} - U_{s^*}$ 
14:       $U_{s^*} \leftarrow V_{s^*}$ 
15:      for all  $(\bar{s}, \bar{a})$  with  $T_{\bar{s}\bar{a}s^*} > 0$  do
16:         $Q_{\bar{s}\bar{a}} \leftarrow Q_{\bar{s}\bar{a}} + \gamma T_{\bar{s}\bar{a}s^*} \Delta V$ 
17:         $V_{\bar{s}} \leftarrow \max_b Q_{\bar{s}b}$ 
18:        Add  $\bar{s}$  to  $PQueue$  with some priority  $p$ 
19:      end for
20:    end loop
21:  end for
22: end for

```

changes has an important implication in the reinforcement learning setting: if an agent, who acts in a large, familiar environment, discovers a novel value for R_{sa} or $T_{sas'}$, this may change the Q -values of some predecessor states of s , but generally it does not lead to a change of all Q -values. Hence, a model-based approach that recomputes all Q -values, like value iteration, is computationally inefficient in the reinforcement learning setting. More precisely, let us assume Equation 1 holds for all state-action pairs (s, a) at a given moment in time. Next, the agent experiences in state s and for action a an unexpected immediate reward or an unexpected transition that suggest a change $\Delta Q_{sa} \neq 0$. If this change also alters the value $\Delta V_s \neq 0$, all Q -values of predecessor states \bar{s} with $T_{\bar{s}\bar{a}s} > 0$ should be updated to $Q_{\bar{s}\bar{a}} \leftarrow Q_{\bar{s}\bar{a}} + \gamma T_{\bar{s}\bar{a}s} \Delta V$ (c.f. 1 line 16). This process of updating Q - and V -values of predecessor states continues until there are no changes in V -values anymore or the time budget is spent. The order in which predecessor states are updated can be heuristically prioritized, e.g. $p = |V_s - U_s|$ in lines 9 and 18 of 1 (Seijen & Sutton, 2013), such that large changes propagate faster, but also other prioritization heuristics are possible and sometimes advisable.

Empirically it is found that a small and constant number of Q -value updates in the order of their priority is sufficient to learn considerably faster, i.e. requiring fewer interac-

tions with the environment, than model-free methods like TD-learning or policy gradient learning (Seijen & Sutton, 2013). This increase in sample efficiency comes at the cost of slightly higher computation demands and a memory complexity that is higher in general, but equal for deterministic environments. The memory complexity of prioritized sweeping is dominated by storing the transition table $T_{sas'}$, which, generally, has complexity $\mathcal{O}(AS^2)$, with A the number of possible actions and S the size of the state space. However, for deterministic environments, or stochastic environments where the number of possible successor states scales as $\mathcal{O}(1)$ (i.e. many entries of the transition table are 0), the memory complexity of prioritized sweeping is $\mathcal{O}(AS)$, equal to the one of model-free methods.

Prioritized sweeping is an instance of the Dyna architecture, where models, i.e. estimates of $T_{sas'}$, are explicitly learned through interactions with the environment and then used to update the Q -values by background planning, i.e. without interaction with the environment ("trying things in your head" Sutton (1991); Sutton & Barto (2018)). Importantly, at decision time, this form of model-based reinforcement learning has the same computational complexity as any model-free method that relies on learned Q -values.

2.1. Domain adaptation of prioritized sweeping

The explicit dependence on the transition table $T_{sas'}$ allows for easily incorporating prior knowledge about the domain, an advantage that is rarely mentioned in discussions of prioritized sweeping. In stationary and stochastic environments, the updates in line 7 of 1 depend on visit counts N_{sa} of state- action pairs (s, a) and transition counts $N_{sas'}$ (such that $T_{sas'} = N_{sas'}/N_{sa}$). Q_{sa} can be efficiently updated, without the need to recompute the sum in Equation 1 (Seijen & Sutton, 2013). For non-stationary stochastic environments it may be better to do recency-weighted updates via leaky integration, i.e. after experiencing the transition (s, a, s') the transition table is updated according to $T_{sas''} \leftarrow (1 - \kappa)T_{sas''}$ for all $s'' \neq s'$ and $T_{sas'} \leftarrow (1 - \kappa)T_{sas'} + \kappa$ for some $\kappa \in (0, 1)$ that could be estimated by the agent and does not need to be the same for all $T_{sas'}$. For non-stationary deterministic environments, experiencing the transition (s, a, s') should result in $T_{sas''} \leftarrow 0$ for all $s'' \neq s'$ and $T_{sas'} \leftarrow 1$. And in non-stationary environments with mixed degrees of stochasticity, the agent could keep track of the volatility of transitions to determine whether to treat them as intrinsically stochastic or non-stationary deterministic.

3. Results

3.1. Episodic control is equivalent to prioritized sweeping for deterministic tree Markov Decision Processes

For deterministic Markov decision processes with $s' = T(s, a)$ and rewards $P(r_t | s_t = s, a_t = a) = R_{sa}$ the Bellman equations reduce to

$$Q(s, a) = R_{sa} + \gamma \max_{a'} Q(T(s, a), a'), \quad (2)$$

where the alternative notation $Q(s, a) = Q_{sa}$ is used for readability. In tree Markov Decision Processes (c.f. Figure 2A) each state has only one predecessor state and thus prioritized sweeping never branches during the backups. For the first episode $\{s_1^{(1)}, a_1^{(1)}, r_1^{(1)}, \dots, s_d^{(1)}, a_d^{(1)}, r_d^{(1)}\}$ in a deterministic tMDP of depth d , prioritized sweeping may proceed as follows: no states are added to the priority queue until the end of the episode at which point the final state $s_d^{(1)}$ is added to the priority queue and $d - 1$ backup steps (iterations of the loop starting on line 11 in 1) are performed. After this, the Q -values are given by

$$Q(s_t^{(1)}, a_t^{(1)}) = G_t^{(1)} \equiv \sum_{\tau=t}^d \gamma^{\tau-t} r_\tau^{(1)}, \quad (3)$$

if the initialization was $Q_{sa} = V_s = U_s = 0$. Note that an unconventional but efficient use of the priority queue is made here: if states were also added to the priority queue during the episode, the final result would be the same, if more backup steps would be allowed (up to $\sum_{k=1}^{d-1} k = d \cdot (d - 1)/2$). When in episode i at time step t a state-action pair $(s_t^{(i)}, a_t^{(i)})$ is re-experienced, the backups of prioritized sweeping result in a change of the Q -value that is equivalent to

$$Q(s_t^{(i)}, a_t^{(i)}) \leftarrow \max \left\{ Q(s_t^{(i)}, a_t^{(i)}), G_t^{(i)} \right\}, \quad (4)$$

i.e. the learning rule of episodic control used by Blundell et al. (2016), which is a formalization of the proposition by Lengyel & Dayan (2008) quoted in the introduction.

To see why prioritized sweeping leads to updates equivalent to Equation 4, we note that $Q(s_t^{(i)}, a_t^{(i)})$ will change when a novel action $a_{t+1}^{(i)}$ is chosen* in the subsequent state $s_{t+1}^{(i)} = T(s_t^{(i)}, a_t^{(i)})$ and $G_{t+1}^{(i)} > Q(s_{t+1}^{(i)}, a')$ for all $a' \neq a_{t+1}^{(i)}$ and thus $\Delta V = G_{t+1}^{(i)} - G_{t+1}^{(j^*)} > 0$, where j^* is the episode number with the previously highest return from state $s_{t+1}^{(i)}$. Hence, by virtue of line 16 in 1, $Q(s_t^{(i)}, a_t^{(i)}) \leftarrow G_t^{(i)}$. If this leads to $\Delta V > 0$ for state

*That is, $a_{t+1}^{(i)} \neq a_{t+1}^{(j)}, \forall j < i$ with $s_{t+1}^{(i)} = s_{t+1}^{(j)}$

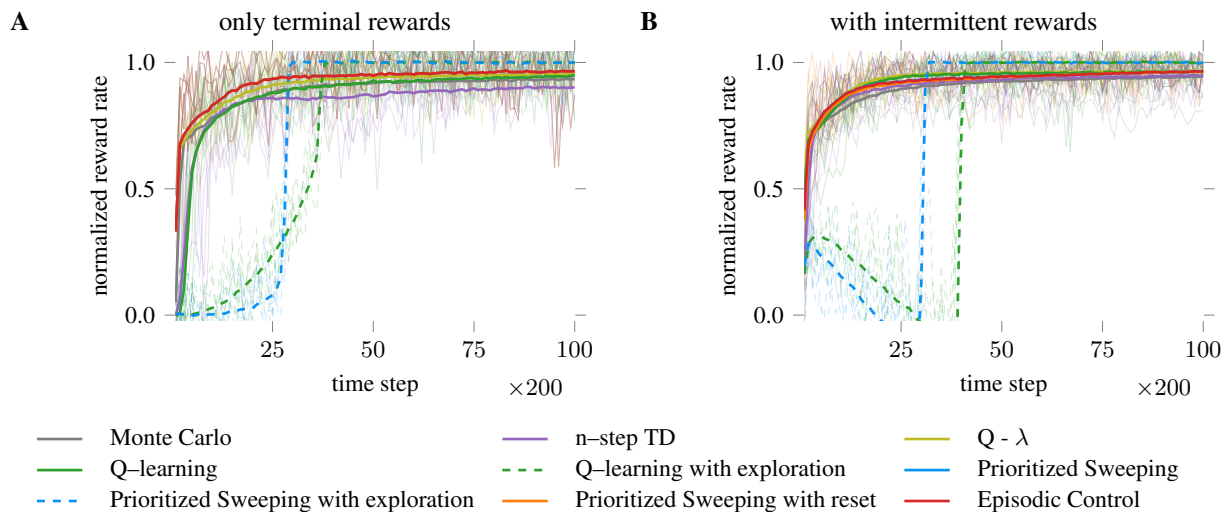


Figure 1. Deterministic tree Markov Decision Processes **A** The curve of the average normalized reward rate of episodic control (red) almost entirely covers the curves for prioritized sweeping (blue) and prioritized sweeping with model resets (orange), indicating that they perform equally well on deterministic tMDPs with 4 actions, depth 5 and reward only in terminal states. The curves of N-step TD (dark blue) and Monte Carlo learning (gray) are very similar, as they are up to the learning rate schedule equivalent in this case; being on-policy methods, they perform slightly worse. Because the Q-learner (green) bootstraps only when re-experiencing a given state-action pair, it learns slower, even with eligibility traces (Q- λ , yellow). Methods with a strong exploration bias caused by large initial Q-values (“optimistic initialization”, dashed curves) sacrifice initial reward for optimal exploitation at a later stage of learning. **B** With intermittent rewards the differences between the algorithms almost vanish, except for the ones with a strong exploration bias. See subsection 3.3 for details.

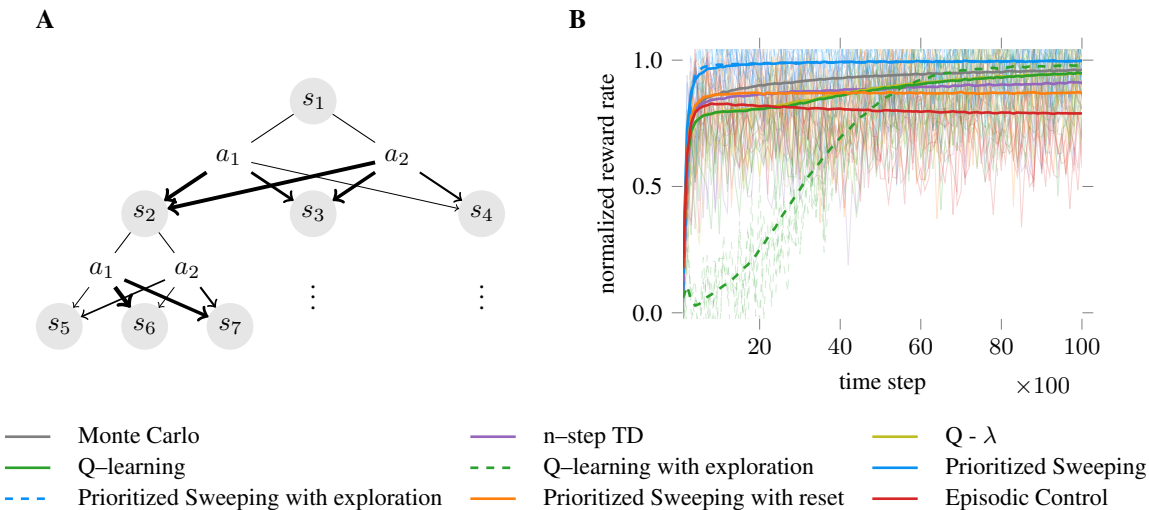


Figure 2. Stochastic tMDPs. **A** Example of a stochastic tree MDP with depth 2, 2 actions and branching factor 3. The thickness of the arrow indicates the relative probability of transitioning to the subsequent state when choosing a given action, e.g. $T_{s_1 a_2 s_2} > T_{s_1 a_2 s_4}$. Rewards are in $[0, 1]$. **B** Learning curves for a stochastic tMDP with depth 4, 4 actions and branching factor 2. See subsection 3.3 for details.

$s_t^{(i)}$, prioritized sweeping further propagates changes backwards along the path the agent took in this episode until $V_{s^*} = U_{s^*}$ in line 17 of 1, after which the backup loop stops.

In Figure 1 we see that episodic control (red curves) performs equivalently to prioritized sweeping (solid blue curves, covered by the red curves). Moreover, we see that episodic control and prioritized sweeping learn faster in tMDPs than alternative methods.

For deterministic tMDPs an implementation of prioritized sweeping does not need to maintain the transition function $T(s, a)$ beyond the end of an episode. It has thus no influence on the performance, if the estimates of the model parameters are reset after each episode. I call this algorithm prioritized sweeping with model reset. In Figure 1 the performance curves of prioritized sweeping with model reset after each episode (orange curves, see also subsection 3.3) also match the ones of episodic control (red curves). With model reset, the memory requirements are identical to those of episodic control. There is also no additional cost in maintaining a priority queue, if only the last state of the episode is added to the priority queue on line 9 in 1, since the queue will afterwards contain only one item at each moment in time: the predecessor of the currently backed up state.

We assumed above that the Q -values are initialized in a way such that for all novel actions a' , i.e. actions that were never chosen before in state s^* , $Q(s^*, a') < \min R_{sa}$. Exploration is not dramatically hampered by this choice, if novel actions are selected whenever possible. If, on the other hand, a maximally exploratory behavior is enforced with the initialization $Q(s, a) \geq d \cdot \max R_{sa}$, where d is the depth of the tree, then $\max_a Q(s, a)$ may decrease over time and thus prioritized sweeping performs also updates that are inconsistent with Equation 4. In deterministic tMDPs this leads rapidly to a full exploration of the decision tree at the cost of low rewards in the initial phase of learning (Figure 1 dashed blue curves). Only bootstrapping methods, like Q-learning or prioritized sweeping, can be forced to aggressive exploration with large initial values. Episodic control is not of this type; the max operation in Equation 4 would just keep the Q -values constant at the large initialization value.

3.2. Prioritized sweeping with model reset outperforms episodic control in general environments

For optimal performance in stochastic (t)MDPs or in deterministic MDPs without tree structure, prioritized sweeping needs to store the transition table. This means, full prioritized sweeping requires more memory and computation than episodic control. However, an interesting competitor of episodic control in terms of memory and computation requirements is a learner that resets the model after each

episode but uses prioritized sweeping at the end of each episode for the backups. If the episodes are long enough, such that the learned model for each episode has not just tree structure, this learner may still have an advantage over episodic control.

In stochastic environments there needs to be some averaging somewhere, e.g. explicit averaging over episodes in Monte Carlo control, choosing a small learning rate in Q-learning or maintaining a maximum likelihood estimate of $T_{sas'}$ in prioritized sweeping. As already observed by Lengyel & Dayan (Lengyel & Dayan, 2008), algorithms that do not properly average, like episodic control or prioritized sweeping with model reset, may perform well at the initial phase of learning but they lead quickly to suboptimal performance (c.f. Figure 2B, red and orange curve). While I do not see a clear theoretical advantage of prioritized sweeping with model reset over episodic control in such environments, it nevertheless performs slightly better in stochastic tMDPs (see Figure 2B, orange above red curve), but overall clearly worse than normal prioritized sweeping.

In deterministic environments without tree structure, states can have multiple predecessor states. In this case, prioritized sweeping also performs backups along *virtually composed* episodes, i.e. episodes that were in their entirety not yet experienced, but whose possibility is known to the agent from experiencing episodes with shared states. This is similar to crossroad where the origin of some roads is known, or the example with the uncharted passage in the introduction. The deterministic maze environment in Figure 3A is of this type: with prioritized sweeping (blue curves) the agent learns rapidly to navigate from any starting position to the goal (red square), even if the model is reset after each episode (orange curves). Episodic control does not branch during the backups. While it performs well initially (red curves), it even gets quickly outperformed by the model-free Q-learner with exploration bonus (dashed green curves).

3.3. Simulation details

In all simulations the reward rate is measured as the mean reward obtained with an ϵ -greedy policy in adjacent windows of T time steps. The normalized reward rate is obtained by an affine transformation of the reward rate such that 0 corresponds to the value of the uniform random policy and 1 corresponds to the policy with the optimal Q-values (obtained by policy iteration) and the same ϵ -greedy action selection. In all figures, the thick curves are obtained by running M simulations with different random seeds on N samples of the MDP in question and averaging the results. The thin curves are 5 random samples of the $N \cdot M$ simulations. The small backups implementation of prioritized sweeping (Seijen & Sutton, 2013) was used with 3 backups per time step. In the variant of prioritized sweeping with model reset,

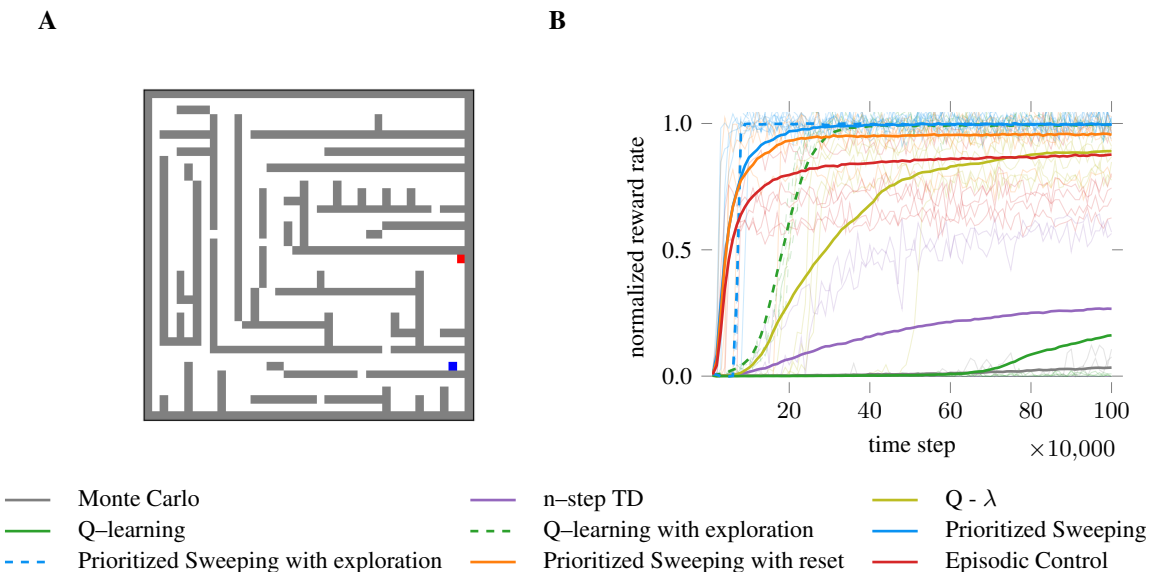


Figure 3. Deterministic maze. **A** Example of a deterministic maze with random initial conditions (blue square) and one goal state (red square). The gray lines indicate walls. The agent moves around by choosing the actions up, down, left and right. If an action would move the agent into a wall, it stays at the current position. There are usually multiple paths that lead from start to goal. The reward is 1 for reaching the goal, 0 otherwise. **B** Prioritized sweeping with model reset (orange curves) performs nearly optimally, while episodic control is rapidly outperformed even by Q-learning with exploration bias. See [subsection 3.3](#) for details.

no backups were performed until the end of the episode, at which point at most d backups are performed, where d is the length of the episode. This is the same number of backups as in episodic control (c.f. [Equation 4](#)). After the end of each episode all transition counts and observed rewards are reset to the initial value in prioritized sweeping with reset. The parameters of the other algorithms were chosen after a short search in the parameter space and are probably close to but not exactly optimal for Q-learning in stochastic tMDPs, n-step TD and Q- λ in all environments. See [Table 1](#) for the parameter choices. The code is available at <http://github.com/jbrea/episodiccontrol>.

4. Discussion

Prioritized sweeping allows to learn efficiently from a small number of experiences. In deterministic environments the demands on memory and computation are low and they increase gracefully with increasing stochasticity. Episodic control is equivalent to prioritized sweeping in the case of deterministic tree Markov Decision Processes, because in this case it is not necessary to maintain a model beyond one episode. The observation that episodic control performs surprisingly well in the initial phase of some Atari games ([Blundell et al., 2016](#)) may be a hint that these games are close to deterministic tree Markov Decision Processes.

I suggested model reset after the end of each episode. While this is a rather ruthless form of forgetting, the simulation

results show that it is still possible to learn fairly well in the maze task. Future work will be needed to explore more elaborate forgetting heuristics that allow to keep the memory load low while preserving high performance.

The current formulation of prioritized sweeping requires a tabular environment, i.e. a representation of states by integer numbers, and its superiority to episodic control in deterministic environments becomes only evident in a Markovian setting where states have multiple predecessors. However, tabular environments contrast with the natural continuity of space and sensor values encountered by biological agents and robots. Additionally, the sensory input may contain details that are irrelevant to the task at hand, e.g. the pedestrians I encounter every time I walk by the same place in my home town, which are irrelevant to my task of walking from A to B. Also, in many tasks the state is not fully observable and state aliasing may occur, i.e. different states have the same observation and can only be distinguished by taking the history of the agent into account.

This raises the question: how can we find functions that map the sensory input to a representation useful for prioritized sweeping? Unfortunately, prioritized sweeping does not immediately lend itself to function approximation, in contrast to model-free learning methods like episodic control, Q-learning or policy gradient learning ([Sutton & Barto, 2018](#)). But an interesting approach could be to learn a function that maps the high dimensional sensory input to a discrete representation, either using unsupervised learning, e.g. a

Is prioritized sweeping the better episodic control?

environment	shared parameters					n-step TD		Q - λ		Q - learning	with expl.	
	γ	ϵ	T	N	M	α	n	α	λ	α	Q_0^Q	$Q_0^{P.S}$
det tMDP Figure 1	1.0	0.1	200	100	8	0.08	5	1.0	0.2	1.0	5.0	5.0
stoch tMDP Figure 2	1.0	0.1	100	100	50	0.05	5	0.1	0.2	0.1	5.0	5.0
maze Figure 3	0.99	0.1	10^4	50	8	0.01	25	0.005	0.2	1.0	5.0	0.005

Table 1. Simulation parameters. Discount factor γ , probability to choose non-optimal action ϵ , duration of window to estimate reward rate T , number of sampled MDPs N , number of trials per MDP M , learning rates α , initial Q-values for Q-learning and prioritized sweeping Q_0^Q and $Q_0^{P.S}$, respectively.

variational autoencoder (Maddison et al., 2016), or using also reward information to shape the map in a similar spirit as the neural episodic control model (Pritzel et al., 2017).

How can we generalize prioritized sweeping to partially observable environments? There does not seem to be an obvious answer to this question, but memories of sequences that lead to reward together with a mechanism to attach converging sequences could allow to go beyond updating single sequences as in episodic control but rather make use of branching during backups as in prioritized sweeping. It is an open question whether this can be made to perform better than the average over smartly selected episodes proposed by Gershman and Daw (Gershman & Daw, 2017).

Prioritized sweeping relies on learning the model of the environment, i.e. the reward table and the transition table. This may look like far from anything we think of episodic memory. But in regions of the transition table with few branching states, but many chains of (s, a, s') -triplets that are experienced only once, backups along such chains look like recalling an episode in reversed order. With increasing experience, it may become impossible to identify all the episodes that contributed to the transition table. This may lead to a gradual shift from episodic memory to semantic memory.

Is prioritized sweeping occurring in brains? The reverse replay of navigation episodes observed in the hippocampus of rats (Foster & Wilson, 2006) can be seen as a signature of both episodic control and prioritized sweeping. But the observation of "never-experienced novel-path sequences" in hippocampal sharp wave ripples of rats (Gupta et al., 2010) is a feature of prioritized sweeping. If a form of prioritized sweeping is implemented in some part of the brain of some species it would be interesting to learn what prioritization heuristics is used.

On a sufficiently abstract level, where irrelevant details of the low-level sensory representations disappear, many tasks solved by humans seem to be tabular and fairly deterministic and it is in this setting where prioritized sweeping excels. This lead me to the question, if prioritized sweeping is the better episodic control. If the defining characteristics of episodic control is sample efficient learning with moderate memory and computation footprint, I would argue, the

answer is yes.

References

Blundell, C., Uria, B., Pritzel, A., Li, Y., Ruderman, A., Leibo, J. Z., Rae, J., Wierstra, D., and Hassabis, D. Model-Free Episodic Control. *ArXiv e-prints*, June 2016.

Clayton, Nicola S., Salwiczek, Lucie H., and Dickinson, Anthony. Episodic memory. *Current Biology*, 17(6): R189R191, Mar 2007. ISSN 0960-9822. doi: 10.1016/j.cub.2007.01.011. URL <http://dx.doi.org/10.1016/j.cub.2007.01.011>.

Foster, David J. and Wilson, Matthew A. Reverse replay of behavioural sequences in hippocampal place cells during the awake state. *Nature*, 440(7084):680–683, Feb 2006. ISSN 1476-4679. doi: 10.1038/nature04587. URL <http://dx.doi.org/10.1038/nature04587>.

Gershman, Samuel J. and Daw, Nathaniel D. Reinforcement learning and episodic memory in humans and animals: An integrative framework. *Annual Review of Psychology*, 68(1):101–128, Jan 2017. ISSN 1545-2085. doi: 10.1146/annurev-psych-122414-033625. URL <http://dx.doi.org/10.1146/annurev-psych-122414-033625>.

Gupta, Anoopum S., van der Meer, Matthijs A.A., Touretzky, David S., and Redish, A. David. Hippocampal replay is not a simple function of experience. *Neuron*, 65(5): 695705, Mar 2010. ISSN 0896-6273. doi: 10.1016/j.neuron.2010.01.034. URL <http://dx.doi.org/10.1016/j.neuron.2010.01.034>.

Lengyel, Máté and Dayan, Peter. Hippocampal contributions to control: The third way. In Platt, J. C., Koller, D., Singer, Y., and Roweis, S. T. (eds.), *Advances in Neural Information Processing Systems 20*, pp. 889–896. Curran Associates, Inc., 2008. URL <http://papers.nips.cc/paper/3311-hippocampal-contributions-to-control-the-third-way.pdf>.

Maddison, C. J., Mnih, A., and Whye Teh, Y. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. *ArXiv e-prints*, November 2016.

Moore, Andrew W. and Atkeson, Christopher G. Prioritized sweeping: Reinforcement learning with less data and less time. *Machine Learning*, 13(1):103130, Oct 1993. ISSN 1573-0565. doi: 10.1007/bf00993104. URL <http://dx.doi.org/10.1007/BF00993104>.

Peng, Jing and Williams, R. J. Efficient learning and planning within the dyna framework. *Adaptive Behavior*, 1(4):437454, Mar 1993. ISSN 1059-7123. doi: 10.1177/105971239300100403. URL <http://dx.doi.org/10.1177/105971239300100403>.

Pritzel, A., Uria, B., Srinivasan, S., Puigdomènech, A., Vinyals, O., Hassabis, D., Wierstra, D., and Blundell, C. Neural Episodic Control. *ArXiv e-prints*, March 2017.

Seijen, Harm Van and Sutton, Rich. Planning by prioritized sweeping with small backups. In Dasgupta, Sanjoy and McAllester, David (eds.), *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 361–369, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL <http://proceedings.mlr.press/v28/vanseijen13.html>.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. *ArXiv e-prints*, December 2017.

Sutton, Richard S. Dyna, an integrated architecture for learning, planning, and reacting. *ACM SIGART Bulletin*, 2(4):160163, Jul 1991. ISSN 0163-5719. doi: 10.1145/122344.122377. URL <http://dx.doi.org/10.1145/122344.122377>.

Sutton, Richard S. and Barto, Andrew G. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, (in progress) second edition, 2018. URL <http://incompleteideas.net/book/the-book-2nd.html>.