A COMPARISON OF LEARNING SPEED AND ABILITY TO COPE WITHOUT EXPLORATION BETWEEN DHP AND TD(0)

Michael Fairbank, Eduardo Alonso
Department of Computing, School of Informatics, City University London, London, UK
We asked the question, “In Adaptive Dynamic Programming and Reinforcement Learning, what are the motivations to use Dual Heuristic Programming (DHP) instead of HDP/TD(0)?”

HDP=“Heuristic Dynamic Programming”. HDP is equivalent to Temporal Difference algorithm TD(0).

Notation: States $x_t$, actions, $a_t$. State transition function: $x_{t+1} = f(x_t, a_t)$ Policy: $a_t = \pi(x_t)$ Rewards given: $r_t = r(x_t, a_t)$

Value function: $V(x)$ = expectation total discounted reward for a trajectory starting from $x$.

Advantages of DHP over HDP (TD(0)):

1) Improved Speed (1700 times faster in our simple experiment)
2) Automatic local exploration

“Exploration versus exploitation” becomes “exploration and exploitation” when following a greedy policy, locally at least.

When DHP makes progress in learning the value gradient along a single trajectory, the trajectory will start to automatically bend itself towards a locally optimal shape, when used with a greedy policy. When TD(0) learns just the values along a single trajectory, the trajectory will not necessarily bend itself towards a locally optimal shape. Ask for a demo!

The blue line shows a trajectory fully trained by HDP/TD(0). The Bellman residual error totalled along this trajectory is zero, while under a greedy policy. The green curve is the theoretical optimal trajectory. Hence zeroing the HDP/TD(0) error along a trajectory does not ensure the trajectory is optimal (because Bellman’s optimality principle needs satisfying over the whole of state space to apply).

The blue line shows a trajectory fully trained by DHP. The DHP-residual error totalled along this trajectory is zero, while under a greedy policy. The green curve is the theoretical optimal trajectory. Here zeroing the DHP does make the trajectory optimal.
TD(0) cannot cope in deterministic environments without explicit local exploration. DHP can cope well.

A simple experiment to learn a single action $\alpha$ which maximises $R = -\alpha^2$

**What is the difference between DHP and HDP (TD(0))?**

Whereas HDP/TD(0) learns a critic by learning values $\tilde{V}$, DHP learns value gradients $\frac{\partial \tilde{V}}{\partial x} \equiv \tilde{G}$.

DHP Weight Update: $\Delta w_t = \alpha \left( \frac{\partial \tilde{G}}{\partial w} \right)_t \left( \frac{\partial r}{\partial w} \right)_t + \gamma \left( \frac{\partial \tilde{V}}{\partial x} \right)_t \tilde{G}_{t+1} - \tilde{G}_t$ where $\tilde{G}$ is the vector output of a neural network, which represents $\frac{\partial \tilde{V}}{\partial x}$.

HDP Weight Update: $\Delta w_t = \alpha \left( \frac{\partial \tilde{V}}{\partial w} \right)_t (r_t + \gamma \tilde{V}_{t+1} - \tilde{V}_t)$, where $\tilde{V}$ is the scalar output of a neural network.

Both are on-line algorithms (and can be off-line!). DHP is model based due to the $\left( \frac{\partial f}{\partial x} \right)$ term. The D/Dx notation means full derivative using the chain rule.
Optimality conditions:

According to Pontryagin’s Maximum principle, zeroing the DHP error all along a trajectory, under a greedy policy, will usually ensure the trajectory is locally optimal. And according to Bellman’s Optimality Principle, zeroing the DHP error all over state space will ensure global optimality.

Zeroing the HDP error along a trajectory does not assure anything. The HDP error needs zeroing all over state space for Bellman’s condition to apply.

Greedy Policy Issues

1. The greedy policy must be satisfied if Bellman’s Optimality Condition is to apply
2. The greedy policy does not care about the absolute value of the relative successor states. Only the relative values are important – and in continuous spaces, those are given by the value-gradient, not the values.

Hence, value-gradients need learning, whether by HDP or by DHP. HDP gets there eventually, after it has learned the values over the whole of state space through exploration. DHP does it directly, for whatever single trajectory it is considering at any moment.

What are the advantages of HDP (TD(0)) over DHP?

HDP is be model free: works even when you don’t know \( f(x_t, a_t) \) and \( r(x_t, a_t) \), when an actor-critic architecture is used;

whereas DHP requires knowledge of the model functions \( f(x_t, a_t) \) and \( r(x_t, a_t) \), and these functions, and the policy, must be differentiable.

DHP assumes the cost of model-function learning is low compared to the cost of value-function learning.

Quote from other DHP researchers: “We mention that some view this model dependence to be an unnecessary ‘expense’. The position of the authors, however, is that the expense is in many contexts more than compensated for by the additional information available to the learning/optimization process.” (G. G. Lendaris and J. C. Neidhoefer, pp. 97–124, Handbook of learning and ADP, 2004).
**Pontryagin’s Maximum Principle (PMP): a visual guide.**

**A Value Function Surface and an Optimal Trajectory:**
Can we just learn the portion of the Value Function surface that is directly under the trajectory?

Instead of a whole surface this would leave us a rollercoaster shaped object.

- This is not quite enough.
- We need *shear* adding to the rollercoaster track. That then gives us the least possible portion of the value function surface in order to decide whether our trajectory is optimal or not.
- The trajectory must simultaneously be found by a greedy policy, whilst the shear is correct. This is PMP!
- DHP error=0 plus greedy policy satisfied all along a trajectory $\Rightarrow$ PMP is satisfied $\Rightarrow$ necessary condition for local optimality.