Beyond the One-Step Greedy Approach in Reinforcement Learning

Yonathan Efroni [1] **Gal Dalal** [1] **Bruno Scherrer** [2] **Shie Mannor** [1]

Abstract

The famous Policy Iteration algorithm alternates between policy improvement and policy evaluation. Implementations of this algorithm with several variants of the latter evaluation stage, e.g, n-step and trace-based returns, have been analyzed in previous works. However, the case of multiple-step lookahead policy improvement, despite the recent increase in empirical evidence of its strength, has to our knowledge not been carefully analyzed yet. In this work, we introduce the first such analysis. Namely, we formulate variants of multiple-step policy improvement, derive new algorithms using these definitions and prove their convergence. Moreover, we show that recent prominent Reinforcement Learning algorithms fit well into our unified framework. We thus shed light on their empirical success and give a recipe for deriving new algorithms for future study.

1. Introduction

Policy Iteration (PI) lies at the core of Reinforcement Learning (RL) and of many planning and on-line learning methods (Konda & Borkar, 1999; Kakade & Langford, 2002; Schulman et al., 2015a; Mnih et al., 2016; Silver et al., 2017b). The classic PI algorithm repeats consecutive stages of i) 1-step greedy policy improvement with respect to (w.r.t.) a value function estimate, and ii) evaluation of the value function w.r.t. the greedy policy. Although multiple variants of the evaluation task have been considered (Puterman & Shin, 1978; Bertsekas & Ioffe, 1996; Sutton & Barto, 1998), the usually considered policy update is the 1-step greedy improvement.

Conducting policy improvement using the common 1-step greedy approach is a specific choice, which is not necessarily the most appropriate one. Indeed, it was empirically recently suggested that greedy approaches w.r.t. multiple steps perform better than w.r.t. 1-step. Notable examples are Alpha-Go and Alpha-Go-Zero (Silver et al., 2016; 2017b;a). There, an approximate online version of multiple-step greedy improvement is implemented via Monte Carlo Tree Search (MCTS) (Browne et al., 2012). The celebrated MCTS algorithm, which instantiates several steps of lookahead improvement, encompasses additional historical impressive accomplishments dating back to the past century and previous decade (Tesauro & Galperin, 1997; Sheppard, 2002; Bouzy & Helmstetter, 2004; Veness et al., 2009). To the best of our knowledge, and despite such empirical successes, the use of a multiple-step greedy policy improvement has never been rigorously studied before. The motivation of this work is to fill this gap, and suggest new possible algorithms in this spirit.

The paper is organized as follows. We start by defining the h-greedy policy, a policy that is greedy w.r.t. a horizon of h steps. Using this definition, we introduce the h-PI algorithm, a class of PI algorithms with multiple-step greedy policy improvement and a guarenteed convergence to the optimal policy. We stress that the term "n-step return" often refered to in the literature, is used in the context of policy evaluation (Sutton & Barto, 1998; Seijen & Sutton, 2014). Therefore, to avoid confusion, we choose to denote the multiple-step greedy policy by the letter h.

We then introduce a novel class of optimal Bellman operators, which is controlled by a continuous parameter $\kappa \in [0, 1]$. This operator is used to define a new greedy policy, the κ-greedy policy, leading to a new PI algorithm which we name κ-PI. This is analogous to the famous TD(λ) algorithm (Sutton, 1988) for the improvement stage. In the TD(λ) algorithm, the λ parameter interpolates between the single-step evaluation update for $\lambda = 0$ and the infinite-horizon (Monte Carlo) evaluation update for $\lambda = 1$. Similarly, for $\kappa = 0$, we recover the traditional 1-step greedy policy for the improvement update and for $\kappa = 1$ we get the infinite-horizon greedy policy, i.e. the optimal policy. Roughly speaking, the κ-greedy policy can be viewed as allowing to make an 'interpolation' over all h-greedy policies. Similarly to the previous paragraph, we use the letter κ to avoid confusion with the parameter λ of TD(λ). Remarkably, we show that computing the κ-greedy policy is equivalent to solving the optimal policy of a surrogate $\kappa\gamma$-

[1]Technion, Israel Institute of Technology [2]INRIA, Villers-lès-Nancy, F-54600, France. Correspondence to: Yonathan Efroni <jonathan.efroni@gmail.com>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

discounted and stationary MDP.

As an additional generalization, we introduce the $\kappa\lambda$-PI. This algorithm has a similar improvement step as the κ-PI, but its policy evaluation stage is 'relaxed', similarly to λ-PI (Bertsekas & Ioffe, 1996). The $\kappa\lambda$-PI further illustrates the difference between the λ and κ parameters. While the former controls the depth of the evaluation task in a way similar to previous works (Sutton & Barto, 1998; Seijen & Sutton, 2014), the latter controls the depth of the improvement step.

Next, we discuss the relation of this work to existing literature, and argue that it offers a generalized view for several recent impressive empirical advancements in RL, which are seemingly unrelated (Schulman et al., 2015b; Silver et al., 2017b). We thus show relevance of our proposed mathematical framework to current state-of-the-art algorithms. We conclude with an empirical display of the influence of these new parameters κ and h on a basic planning task. We empirically demonstrate that the best performance is obtained with non-trivial choices of them. This motivates future study of new RL algorithms which can be derived from the introduced framework in this work.

2. Preliminaries

Our framework is the infinite-horizon discounted Markov Decision Process (MDP). An MDP is defined as the 5-tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$ (Puterman, 1994), where $\mathcal{S}$ is a finite state space, $\mathcal{A}$ is a finite action space, $P \equiv P(s'|s,a)$ is a transition kernel, $R \equiv r(s,a)$ is a reward function, and $\gamma \in (0,1)$ is a discount factor. Let $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ be a stationary policy, where $\mathcal{P}(\mathcal{A})$ is a probability distribution on $\mathcal{A}$. Let $v^\pi \in \mathbb{R}^{|\mathcal{S}|}$ be the value of a policy π, defined in state s as $v^\pi(s) \equiv \mathbb{E}^\pi_{|s}[\sum_{t=0}^\infty \gamma^t r(s_t, \pi(s_t))]$, where $\mathbb{E}^\pi_{|s}$ denotes expectation w.r.t. the distribution induced by π and conditioned on the event $\{s_0 = s\}$. For brevity, we respectively denote the reward and value at time t by $r_t \equiv r(s_t, \pi_t(s_t))$ and $v_t \equiv v(s_t)$. It is known that

$$v^\pi = \sum_{t=0}^{\infty} \gamma^t (P^\pi)^t r^\pi = (I - \gamma P^\pi)^{-1} r^\pi,$$

with the component-wise values $[P^\pi]_{s,s'} \triangleq P(s' \mid s, \pi(s))$ and $[r^\pi]_s \triangleq r(s, \pi(s))$. Our goal is to find a policy π^* yielding the optimal value v^* such that

$$v^* = \max_\pi (I - \gamma P^\pi)^{-1} r^\pi = (I - \gamma P^{\pi^*})^{-1} r^{\pi^*}. \quad (1)$$

This goal can be achieved using the three classical operators (with equalities holding component-wise):

$$\forall v, \pi,\ T^\pi v = r^\pi + \gamma P^\pi v, \quad (2)$$

$$\forall v,\ Tv = \max_\pi T^\pi v, \quad (3)$$

$$\forall v,\ \mathcal{G}(v) = \{\pi : T^\pi v = Tv\}, \quad (4)$$

where T^π is a linear operator, T is the optimal Bellman operator and both T^π and T are γ-contraction mappings w.r.t. the max norm. It is known that the unique fixed points of T^π and T are v^π and v^*, respectively. The set $\mathcal{G}(v)$ is the standard set of 1-step greedy policies w.r.t. v. Furthermore, given v^*, the set $\mathcal{G}(v^*)$ coincides with that of stationary optimal policies. In other words, every policy that is 1-step greedy w.r.t. v^* is optimal and vice versa.

3. The h-Greedy Policy and h-PI

In this section we introduce the h-greedy policy, a generalization of the 1-step greedy policy. This leads us to formulate a new PI algorithm which we name "h-PI". The h-PI is derived by replacing the improvement stage of the PI, i.e, the 1-step greedy policy, with the h-greedy policy. We further prove its convergence and show it inherits most properties of PI.

Let $h \in \mathbb{N}\backslash\{0\}$. A h-greedy policy w.r.t. a value function v belongs to the following set of policies,

$$\arg\max_{\pi_0} \max_{\pi_1,..,\pi_{h-1}} \mathbb{E}^{\pi_0 \cdots \pi_{h-1}}_{|\bullet} \left[\sum_{t=0}^{h-1} \gamma^t r(s_t, \pi_t(s_t)) + \gamma^h v(s_h) \right]$$
$$= \arg\max_{\pi_0} \mathbb{E}^{\pi_0}_{|\bullet} \left[r(s_0, \pi_0(s_0)) + \gamma \left(T^{h-1} v\right)(s_1) \right] \quad (5)$$

where the notation $\mathbb{E}^{\pi_0 \cdots \pi_{h-1}}_{|\bullet}$ means that we condition on the trajectory induced by the choice of actions $(\pi_0(s_0), \pi_1(s_1), \ldots, \pi_{h-1}(s_{h-1}))$ and the starting state $s_0 = \bullet$. Since the argument in (5) is a vector, the maximization is component-wise, i.e., we wish to find the choice of actions that will jointly maximize the entries of the vector. Thus, the h-greedy policy chooses the first optimal action of a non-stationary, optimal control problem with horizon h. As the equality in (5) suggests, this policy can be equivalently interpreted as the 1-step greedy policy w.r.t. $T^{h-1}v$. Although the former view is more natural, it is, in fact, the latter on which this section's proofs are based. Thus, the set of h-greedy policies w.r.t. v, $\mathcal{G}_h(v)$, can be expressed as follows:

$$\forall v, \pi,\ T^\pi_h v \stackrel{\text{def}}{=} T^\pi T^{h-1} v, \quad (6)$$

$$\forall v,\ \mathcal{G}_h(v) = \{\pi : T^\pi_h v = T^h v\}. \quad (7)$$

Remark 1. *This is a generalization of the standard 1-step greedy operation, which one recovers by taking $h = 1$. Each call to the greedy operator $\mathcal{G}_h$ amounts to identifying for all states, the best first action of an h-horizon optimal control problem. More interestingly, one may compute these first optimal actions with specifically designed procedures such as A^*-like/optimistic tree exploration algorithm (Hren & Munos, 2008; Busoniu & Munos, 2012; Munos, 2014; Szörényi et al., 2014; Grill et al., 2016).*

With this set of operators, we consider Algorithm 1, the h-PI algorithm, where the assignments hold point-wise. This algorithm alternates between i) identifying the h-greedy policy, i.e, solving the h-horizon optimal control problem, and ii) estimating the value of this policy.

Algorithm 1 h-PI

Initialize: $h \in \mathbb{N}\backslash\{0\},\ v \in \mathbb{R}^{|\mathcal{S}|}$
while the value v changes **do**
  $\pi \leftarrow \arg\max_{\pi_0} \max_{\pi_1,..,\pi_{h-1}} \mathbb{E}_{|\bullet}^{\pi_0 \cdots \pi_{h-1}} \left[\sum_{t=0}^{h-1} \gamma^t r_t + \gamma^h v_h\right]$
  $v \leftarrow \mathbb{E}_{|\bullet}^{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$
end while
Return π, v

As we are about to see, this new algorithm inherits most properties of standard PI. We start by showing a monotonicity property for the h-greedy operator.

Lemma 1 (Policy improvement of the h-greedy policy)**.** *Let $\pi' \in \mathcal{G}_h(v^\pi)$. Then $v^{\pi'} \geq v^\pi$ component-wise, with equality holding if and only if π is an optimal policy.*

Proof. First observe that

$$v^\pi = T^\pi v^\pi \leq T v^\pi. \tag{8}$$

Then, sequentially using (8), (7) and (6), we have

$$v^\pi = (T^\pi)^h v^\pi \leq T^h v^\pi = T_h^{\pi'} v^\pi = T^{\pi'}(T^{h-1} v^\pi). \tag{9}$$

This leads to the following inequalities:

$$\begin{aligned} v^\pi &\leq T^{\pi'}(T^{h-1} v^\pi) && (10)\\ &\leq T^{\pi'}(T^{h-1} T v^\pi) = T^{\pi'}(T^h v^\pi) && (11)\\ &= T^{\pi'}\left(T^{\pi'} T^{h-1} v^\pi\right) = \left(T^{\pi'}\right)^2 (T^{h-1} v^\pi) && (12)\\ &\leq \cdots \leq \lim_{n\to\infty} \left(T^{\pi'}\right)^n (T^{h-1} v^\pi) = v^{\pi'}. && (13) \end{aligned}$$

In the above derivation, (10) is due to (9), (11) is due to (8) and the monotonicity of $T^{\pi'}$ and T (and thus of their composition), (12) is due to (9), and (13) is due to the fixed point property of $T^{\pi'}$. Lastly, notice that $v^\pi = v^{\pi'}$ if and only if (cf. (8)) $T v^\pi = v^\pi$, which holds if and only if π is the optimal policy. □

Thus, the improvement property of the 1-step greedy policy also holds for the h-greedy policy. As a consequence, the h-PI algorithm produces a sequence of policies with component-wise increasing values, which directly implies convergence (since the sequence is bounded). We can be more precise about the convergence speed by generalizing several known results on PI to h-PI. Let us begin by the following lemma, which is essentially a consequence of the fact that T^h is a γ^h-contraction (see Appendix C in the supplementary material for a proof).

Lemma 2. *The sequence $\{\|v^* - v^{\pi_k}\|_\infty\}_{k\geq 0}$ is contracting with coefficient γ^h.*

Thus, the convergence rate is at most γ^h, which generalizes the known γ convergence rate of the original ($h = 1$) PI algorithm (Puterman, 1994). The next theorem describes a result with respect to the termination of the algorithm.

Theorem 3. *The h-PI algorithm converges in at most $|\mathcal{S}|(|\mathcal{A}| - 1)\left\lceil (h \log \frac{1}{\gamma})^{-1} \log\left(\frac{1}{1-\gamma}\right)\right\rceil$ iterations.*

Proof. The proof follows the steps of (Scherrer, 2016), Section 7, where instead of using the contraction coefficient of PI algorithm we use the contraction coefficient of h-PI, proved in Lemma 2. □

Remark 2. *Note that the fact that T^h is γ^h-contraction does not imply that all existing PI results extend to h-PI with γ^h replacing γ. For instance, the rightmost term in the logarithm of Theorem 3 is $\frac{1}{1-\gamma}$ and not $\frac{1}{1-\gamma^h}$.*

Remark 3. *Notice how the complexity term in Theorem 3 is a decreasing function of h. At the limit $h \to \infty$, running a single iteration of h-PI is sufficient for finding the optimal value-policy pair. However, although the total number of iterations is reduced with h increasing, each iteration is expected to be computationally more costly.*

4. The κ-Greedy Policy

In this section, we introduce an additional, novel generalization of the 1-step greedy policy: *the κ-greedy policy*. Similarly to the previous section, the newly introduced greedy policy leads to a new PI algorithm, which we shall name "κ-PI". This generalization will be based on the definition of a new *κ-optimal Bellman operator*. This operator will also naturally lead to a new Value Iteration (VI) algorithm, "κ-VI". In the later Section 6, we shall highlight the relation between the κ-greedy and the previously introduced h-greedy policy.

4.1. κ-Optimal Bellman Operator and the κ-Greedy Policy

In this section we will derive the κ-optimal Bellman operator and define its induced greedy policy, the κ-greedy policy.

Given a parameter $\kappa \in [0, 1]$, consider the following operator:

$$\forall v, \pi,\ T_\kappa^\pi v \stackrel{\text{def}}{=} (1-\kappa) \sum_{j=0}^{\infty} \kappa^j (T^\pi)^{j+1} v. \tag{14}$$

This linear operator is identical to the one of the λ-PI algorithm (Bertsekas & Ioffe, 1996). By simple linear algebra arguments (see e.g. (Scherrer, 2013a)), one can see that

$$\forall v, \pi,\ T_\kappa^\pi v = (I - \kappa\gamma P^\pi)^{-1}(r^\pi + (1-\kappa)\gamma P^\pi v) \quad (15)$$
$$= v + (I - \kappa\gamma P^\pi)^{-1}(T^\pi v - v). \quad (16)$$

Comparing (1) and (15), we can interpret (15), given a fixed v, as the value of policy π in a *surrogate* stationary MDP. This MDP has the same dynamics as the MDP we wish to solve, a $\kappa\gamma \leq \gamma$ discount factor and a reward function $\hat{r}^{\pi,v}$ given by

$$\hat{r}^{\pi,v} \stackrel{\text{def}}{=} r^\pi + (1-\kappa)\gamma P^\pi v. \quad (17)$$

Thus, this surrogate stationary MDP depends on both v and κ. According to basic MDP theory, the surrogate MDP has an optimal value. We shall denote its optimal value by $T_\kappa v$ and refer to T_κ as the "κ-optimal Bellman operator". Note that, from (1) and (15), we have

$$T_\kappa v = \max_\pi (I - \kappa\gamma P^\pi)^{-1}\hat{r}^{\pi,v} = \max_\pi T_\kappa^\pi v. \quad (18)$$

The κ-optimal Bellman operator naturally induces a new set of greedy policies, the set of κ-greedy policies w.r.t. v, which we shall denote by $\mathcal{G}_\kappa(v)$, and define as follows:

$$\forall v,\ \mathcal{G}_\kappa(v) = \{\pi : T_\kappa^\pi v = T_\kappa v\}. \quad (19)$$

Remark 4. *The κ-optimal Bellman operator is a generalization of the optimal Bellman operator, which one recovers by taking $\kappa = 0$; i.e., $T_{\kappa=0} = T$. Additionally, applying once $T_{\kappa=1}$ is equivalent to solving the original γ-discounted MDP; i.e, for any v, $v^* = T_{\kappa=1}v$. For all other values $0 < \kappa < 1$, applying T_κ amounts to solving a stationary MDP with reduced discount factor, the solution of which can be obtained using any generic planning, model-free or model-based RL algorithm. As was previously analyzed in (Petrik & Scherrer, 2009; Strehl et al., 2009; Jiang et al., 2015) and reported (François-Lavet et al., 2015), solving a MDP with a smaller discount factor is in general easier.*

In the next lemma we prove that both T_κ^π and T_κ are contractions with respective fixed points v^π and v^*.

Lemma 4. *For any π, T_κ^π and T_κ are ξ-contraction mappings w.r.t. the max norm, where $\xi = \frac{(1-\kappa)\gamma}{1-\gamma\kappa} \in [0, \gamma]$, and have unique fixed points v^π and v^*, respectively. Moreover, $\mathcal{G}_\kappa(v^*) = \mathcal{G}(v^*)$.*

Proof. From (15), we see that for all v and w,

$$T_\kappa^\pi v - T_\kappa^\pi w = (1-\kappa)(I - \kappa\gamma P^\pi)^{-1}\gamma P^\pi (v - w), \quad (20)$$

which implies, by taking the max norm, that T_κ^π is a ξ-contraction mapping.

From (18), we see that for all v and w,

$$T_\kappa^{\pi_w^*} v - T_\kappa^{\pi_w^*} w \leq T_\kappa v - T_\kappa w \leq T_\kappa^{\pi_v^*} v - T_\kappa^{\pi_v^*} w, \quad (21)$$

and this in turn implies, by again taking the max norm, that T_κ is also a ξ-contraction mapping.

Since both operators are contraction mappings, they have one and only one fixed point. To identify them, it is thus sufficient to show that the foreseen fixed points are indeed fixed points. By (16), since $v^\pi = T^\pi v^\pi$, it is clear that $v^\pi = T_\kappa^\pi v^\pi$. Now, from (18) and (16),

$$T_\kappa v^* = \max_\pi T_\kappa^\pi v^* = v^* + z, \quad (22)$$
$$\text{with } z = \max_\pi (I - \kappa\gamma P^\pi)^{-1}(T^\pi v^* - v^*).$$

By the optimality of v^*, we know that for any π, $T^\pi v^* - v^* \leq 0$; since the matrix $(I - \kappa\gamma P^\pi)^{-1} = \sum_{i=0}^\infty (\kappa\gamma P^\pi)^i$ is only made of non-negative coefficients, it follows that $z \leq 0$. Then, since for $\pi = \pi^*$, $T^\pi v^* = v^*$, we deduce that $z = 0$ and, as a consequence, $T_\kappa v^* = v^*$.

Lastly, we show that $\mathcal{G}_\kappa(v^*) = \mathcal{G}(v^*)$. Let $\pi \in \mathcal{G}_\kappa(v^*)$. Thus, $T_\kappa^\pi v^* = T_\kappa v^* = v^*$, where the second equality holds since v^* is the fixed point of T_κ. From (22) we deduce that $0 = z = T^\pi v^* - v^*$. Thus, $T^\pi v^* = v^* = Tv^*$ and π is in $\mathcal{G}(v^*)$. To show the opposite direction, we assume that π is in $\mathcal{G}(v^*)$. Thus, it holds that $z = T^\pi v^* - v^* = 0$; hence, $T_\kappa^\pi v^* = v^*$, and indeed π is in $\mathcal{G}_\kappa(v^*)$. □

4.2. Two New Algorithms: κ-PI and κ-VI

In the previous subsection, we derived the κ-optimal Bellman operator, and defined its induced κ-greedy policy. These operators lead us to consider Algorithm 2, the κ-PI algorithm, where the assignments hold component-wise.

Algorithm 2 κ-PI

Initialize: $\kappa \in [0,1],\ v \in \mathbb{R}^{|\mathcal{S}|}$
while the value v changes **do**
  $\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{|\cdot}^{\pi'}[\sum_{t=0}^\infty (\kappa\gamma)^t (r_t + \gamma(1-\kappa)v_{t+1})]$
  $v \leftarrow \mathbb{E}_{|\cdot}^{\pi}[\sum_{t=0}^\infty \gamma^t r_t]$
end while
Return π, v

This algorithm repeats consecutive steps of i) identifying the κ-greedy policy, i.e, solving the optimal policy of a surrogate, stationary MDP, with a reduced $\gamma\kappa$ discount factor, and ii) estimating the value of this policy. As we shall see, the iterative process is guaranteed to converge to the optimal policy-value pair of the MDP we wish to solve.

Similarly to Section 3, we shall now prove that the κ-PI algorithm inherits many properties from PI. We start by showing a monotonicity property for the κ-greedy operator.

Lemma 5 (Policy improvement of the κ-greedy policy)**.** *Let $\pi' \in \mathcal{G}_\kappa(v^\pi)$. Then $v^{\pi'} \geq v^\pi$ component-wise, with equality if and only if π is an optimal policy.*

Proof. Since v^π is also the fixed point of T_κ^π, by Lemma 4, and using the definition of T_κ (see 18),

$$v^\pi = T_\kappa^\pi v^\pi \leq T_\kappa v^\pi = T_\kappa^{\pi'} v^\pi \tag{23}$$

by choosing $\pi' \in \mathcal{G}_\kappa(v^\pi)$. Using the monotonicity of the operator $T_\kappa^{\pi'}$ and repeating (23) we get,

$$v^\pi = T_\kappa^\pi v^\pi \leq T_\kappa^{\pi'} v^\pi \leq \cdots \leq \lim_{n\to\infty} \left(T_\kappa^{\pi'}\right)^n v^\pi = v^{\pi'}.$$

The final equality holds since $T_\kappa^{\pi'}$ is a contraction mapping, and $v^{\pi'}$ is its fixed point, due to Lemma 4. According to the same lemma, equality holds if and only if $v^\pi = v^*$. □

Similarly to Lemma 2, let us state a result that is a direct consequence of the fact that T_κ, that induces the κ-greedy policy, is a ξ contraction (see Appendix D in the supplementary material for a proof).

Lemma 6. *The sequence $(\|v^* - v^{\pi_k}\|_\infty)_{k\geq 0}$ is contracting with coefficient ξ.*

In the following theorem to the lemma we upper bound the maximal number of iteration it takes the κ-PI algorithm to terminate (the proof is the same as that for Theorem 3).

Theorem 7. *The κ-PI algorithm converges in at most $|\mathcal{S}|(|\mathcal{A}|-1)\left\lceil (\log \frac{1-\kappa\gamma}{(1-\kappa)\gamma})^{-1} \log\left(\frac{1}{1-\gamma}\right)\right\rceil$ iterations.*

Remark 5. *As in the case of h-PI, the complexity term in Theorem 7 is a decreasing function of κ. This complements the fact that κ-PI converges in one iteration if $\kappa = 1$. Again, as κ increases, each iteration is expected to be computationally more demanding since the surrogate MDP is less discounted.*

Lastly, using the κ-optimal Bellman operators, T_κ, we derive a non-trivial generalization of the VI algorithm which we name "κ-VI". The κ-VI algorithm repeatedly applies T_κ until convergence. The convergence proof of κ-VI and its rate of convergence, ξ, thus follows easily as a corollary of Lemma 4. In the next section, we shall group both κ-PI κ-VI into a single, larger, class of algorithms that contains them both.

5. $\kappa\lambda$-PI

Even though κ-PI and κ-VI are distinct algorithms, in this section we unite them under the single "$\kappa\lambda$-PI" class of algorithms (Algorithm 3). This generalization is similar to the λ-PI which interpolates between standard PI and VI (Bertsekas & Ioffe, 1996; Bertsekas & Tsitsiklis, 1995; Scherrer, 2013b; Bertsekas, 2015). We shall describe how to estimate the value of the κ-greedy policy on a surrogate MDP with a smaller horizon and shaped reward, and show that this still yields convergence to the optimal policy and value. Thus, in $\kappa\lambda$-PI, we ease the policy evaluation phase of κ-PI via solving a simpler task.

In the λ-PI, the improvement stage is the common 1-step greedy policy, and the evaluation stage is relaxed by applying the T_λ^π operator (14) instead of fully estimating the value, where $\lambda \in [0,1]$. We start by formulating the appropriate generalization of the T_λ^π operator to κ-PI. Let $\bar{\lambda} \in [0,1]$. The analogous T_λ^π to our framework is the following linear operator:

$$\forall v, \pi,\ T_{\bar{\lambda},\kappa}^\pi v = (1-\bar{\lambda}) \sum_{j=0}^{\infty} \bar{\lambda}^j T_\kappa^\pi v.$$

Interestingly, due to the fact this operator is affine, the following lemma shows that $T_{\bar{\lambda},\kappa}^\pi$ is equivalent to yet another T_λ^π operator (14) where λ is a function of $\bar{\lambda}$ and κ (see Appendix E in the supplementary material for a proof).

Lemma 8. *For every $\bar{\lambda}, \kappa \in [0,1]$, $T_{\bar{\lambda},\kappa}^\pi = T_\lambda^\pi$, where $\lambda = \kappa + \bar{\lambda}(1-\kappa)$, i.e, $\lambda \in [\kappa, 1]$.*

Algorithm 3 $\kappa\lambda$-PI

Initialize: $\kappa \in [0,1],\ \lambda \in [\kappa, 1],\ v \in \mathbb{R}^{|\mathcal{S}|}$
while some stopping condition is not satisfied **do**
  $\pi \leftarrow \arg\max_{\pi'} \mathbb{E}_{|\cdot}^{\pi'}[\sum_{t=0}^{\infty}(\kappa\gamma)^t(r_t + \gamma(1-\kappa)v_{t+1})]$
  $v \leftarrow \mathbb{E}_{|\cdot}^{\pi}[\sum_{t=0}^{\infty}(\lambda\gamma)^t(r_t + \gamma(1-\lambda)v_{t+1})]$
end while
Return π, v

Consider $\kappa\lambda$-PI (Algorithm 3, in which again the assignments hold component-wise). Its improvement stage is similar to the κ-PI algorithm; however, in the evaluation step, we apply $T_{\bar{\lambda},\kappa}^\pi$, or, equivalently, T_λ^π. Indeed, by setting $\lambda = \kappa$ we recover κ-VI (see Appendix F) and by setting $\lambda = 1$ we recover κ-PI. Moreover, by setting $\kappa = 0$, we obtain the class of λ-PI algorithms. Notice that $\lambda \in [\kappa, 1]$, whereas for λ-PI, $\lambda \in [0,1]$. We leave for future work the question whether the $\kappa\lambda$-PI makes sense for $\lambda \in [0, \kappa)$.

At this point of the paper, we have reached a very general algorithmic formulation. We shall not only prove convergence for the $\kappa\lambda$-PI, but also provide a sensitivity analysis that shows how errors may propagate along the steps. This may indeed be of interest if we use approximations when computing a κ-greedy policy or updating the value function. Also, doing so, we generalize similar results on λ-PI (Scherrer, 2013a). In the following the subscript notation, k, refers to the kth iteration of the algorithm (the proof is deferred to the end of the paper for clarity, in Appendix A).

Theorem 9. *Let $\kappa \in [0,1]$ and $\lambda \in [\kappa, 1]$. Assume that in Algorithm 3 we employ noisy versions of the two steps at the kth iteration*

$$\begin{aligned} \pi_{k+1} &\leftarrow \hat{\mathcal{G}}_{\kappa}^{\delta_{k+1}}(v_k) \\ v_{k+1} &\leftarrow T_{\lambda}^{\pi_{k+1}} v_k + \epsilon_{k+1}, \end{aligned} \tag{24}$$

where the noisy improvement of (24) means:

$$T_{\kappa}^{\pi_{k+1}} v_k \geq T_{\kappa} v_k - \delta_{k+1}.$$

Assume that for all k, $\|\epsilon_k\|_\infty \leq \epsilon$ and $\|\delta_k\|_\infty \leq \delta$. Then,

$$\begin{aligned} \limsup_{k\to\infty} \|v^* - v^{\pi_k}\| &\leq \frac{2\xi\epsilon + \delta}{(1-\xi)^2} \\ &= \frac{2\gamma(1-\kappa)(1-\kappa\gamma)\epsilon + (1-\kappa\gamma)^2\delta}{(1-\gamma)^2}. \end{aligned}$$

Once again, we can measure how increasing κ allows improving these asymptotic bounds. Furthermore, observe that the bounds do not depend on λ.

By setting $\epsilon = \delta = 0$ we get the following corollary.

Corollary 10. *The $\kappa\lambda$-PI algorithms converges to the optimal value-policy pair.*

6. Relation to Existing Works

In this section we compare the κ-greedy policy (Section 4) to the h-greedy policy (Section 3). Furthermore, we connect previous works of the literature to the framework developed in this paper.

Let $v : \mathcal{S} \to \mathbb{R}$ be a value function, $h \in \mathbb{N}\backslash\{0\}$ and define the following random variable, the future h-step return:

$$r_v^{(h)} \stackrel{\text{def}}{=} \sum_{t=0}^{h-1} \gamma^t r_t + \gamma^h v(s_h).$$

Consider the h-greedy policy w.r.t. to v, defined in (5). We have for all $\pi \in \mathcal{G}_h(v)$ and $s \in \mathcal{S}$,

$$\pi(s) = \arg\max_{\pi_0} \max_{\pi_1,..,\pi_{h-1}} \mathbb{E}_{|s}^{\pi_0 \cdots \pi_{h-1}} \left[r_v^{(h)} \right].$$

Alternatively, consider a different generalization of the greedy step in the form of a greedy policy w.r.t. a *κ-weighted* average of the future rewards $\{r_v^{(h)}\}_{h\geq 1}$:

$$\mathcal{G}_{\kappa\text{-weighted}}(v) \stackrel{\text{def}}{=} \arg\max_{\pi'} \mathbb{E}_{|\bullet}^{\pi'} [(1-\kappa) \sum_{h=0}^{\infty} \kappa^h r_v^{(h+1)}].$$

We can highlight the following strong relation between these two greedy sets (see Appendix G in the supplementary material for a proof).

Proposition 11. *Let $\delta_v(s_t) = r(s_t, \pi(s_t)) + \gamma v(s_{t+1}) - v(s_t)$. We have*

$$\mathcal{G}_{\kappa\text{-weighted}}(v) = \mathcal{G}_{\kappa}(v) = \arg\max_{\pi} \mathbb{E}_{|\bullet}^{\pi} \left[\sum_{t=0}^{\infty} (\kappa\gamma)^t \delta_v(s_t) \right].$$

The second equality in Proposition 11 reveals a connection to two existing works: planning with shorter horizons (Ng, 2003) and generalized advantage estimation (Schulman et al., 2015b). Using the terminology of our work, the approach of (Ng, 2003) is equivalent to performing a single κ-greedy improvement step. The theory we developped in this paper suggests that there is no reason to stop after only one step. In (Schulman et al., 2015b), the implemented algorithm is equivalent to an on-line Policy Gradient variant of the κ-PI algorithm. As Proposition 11 in our work states, the objective function considered in (Schulman et al., 2015b) describes the very same surrogate MDP that is being solved in the improvement step of κ-PI. Moreover, in that work, an evaluation algorithm estimates the value of the current policy similarly to the evaluation stage of κ-PI. We can interpret the empirically demonstrated trade-off in λ of (Schulman et al., 2015b) as a trade-off in κ. Finally, the policy update phase in the MCTS approach in RL, and in Alpha-Go (Silver et al., 2016; 2017b;a) as an instance of it, is conceptually similar to the policy update in an asynchronous online version of h-PI.

7. Experimental Results

In this section we empirically test the h- and κ-PI algorithms on a toy grid-world problem. As mentioned before, in h-PI, performing the greedy step amounts to solving a h-horizon optimal control problem (Remark 1), and in κ-PI, it amounts to solving a $\gamma\kappa$-discounted stationary MDP (Remark 4). In both cases, conducting these operations in practice can be done with either a generic planning, model-free or model-based algorithm. Here, we implement the h- and κ-greedy step via the VI algorithm. In the former case, we simply do h steps, while in the latter case, we stop VI when the value change in max norm is less than $\epsilon = 10^{-5}$ (other values did produce the same qualitative behavior). With this choice, note that $\kappa = 1$ is equivalent to solving the problem with VI.

We conduct our simulations on a simple $N \times N$ deterministic grid-world problem with $\gamma = 0.97$. The actions set is {'up','down','right','left','stay'}. In each experiment, we randomly chose a single state and placed a reward $r_g = 1$. In all other states the reward was drawn uniformly from $[-0.1r_g, 0.1r_g]$. In the considered problem there is no terminal state. Also, the entries of the initial value function are drawn from $\mathcal{N}(0, r_g^2)$. We ran h- and κ-PI and counted the *total* number of calls to the simulator. Each such "call" takes a state-action pair (s, a) as input, and returns the current reward and next (deterministic) state.

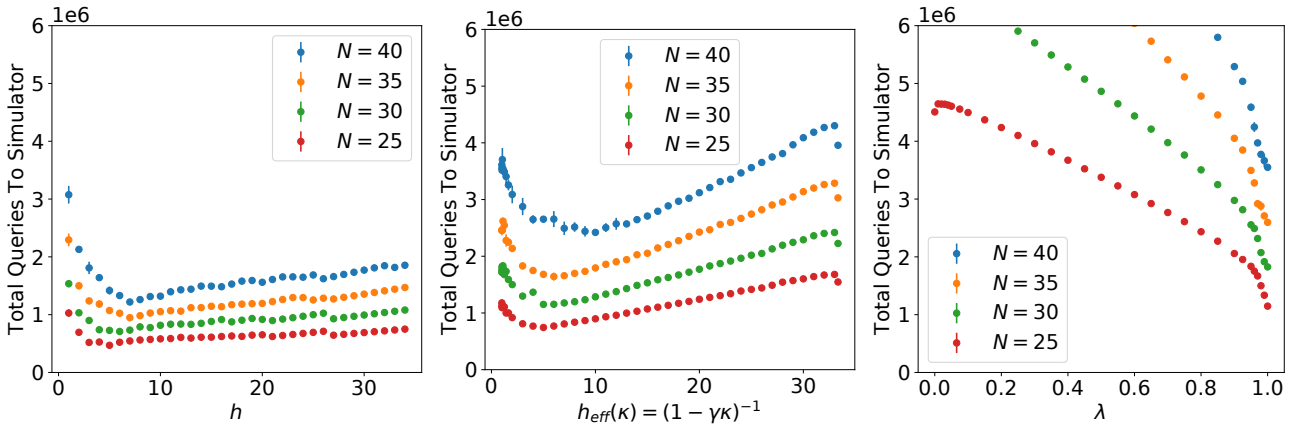

Figure 1: Empirical performance of h-PI, κ-PI and λ-PI for different grid sizes, N (see Section 7). The shown results are the average of 5 experiments. The standard deviation is shown as errorbars. In all plots, the y-axis is the total queries to simulator until convergence, which we chose as a performance criterion. **(Left)** Performance of h-PI as a function of h. **(Center)** Performance of κ-PI as a function of $h_{\text{eff}}(\kappa) = (1 - \gamma\kappa)^{-1}$, the 'effective' planning horizon. The κ values that resulted in the lowest number of simulator queries are $\kappa_{\text{opt}} = 0.82, 0.82, 0.88, 0.92$ for $N = 25, 30, 35, 40$, respectively. **(Right)** Performance of λ-PI as a function of λ. This corresponds to $\kappa\lambda$-PI with $\kappa = 0$. The three plots demonstrate that the algorithms introduced in this work, h-PI and κ-PI, can outperform λ-PI in terms of best empirical performance.

Figure 1 shows the average number of calls to the simulator as a function of h or κ. For each value, experiments were conducted 5 times. The figure depicts that optimal computational efficiency is obtained for some value of the parameters that is not trivial ($h \notin \{1, \infty\}$ and $\kappa \notin \{0, 1\}$). Empirically, these 'optimal' parameter values slowly grow with the grid dimension N. For a comparison, we measured the empirical performance of λ-PI (notice that $\lambda = 1$, which is PI, corresponds to κ-PI with $\kappa = 0$). Our simulations show that the performance of λ-PI is inferior to that of our new algorithms.

8. Discussion and Future Work

In this work, we introduced and formulated two possible approaches that generalize the traditional 1-step greedy operator. Borrowing from the general principle behind Dynamic Programming, we proposed a new family of techniques for solving a single complex problem via iteratively solving smaller sub-problems. We showed that the discussed approaches are coherent with previous lines of work, by generalizing existing analyses on the 1-step greedy policy to the h- and κ-greedy policies. In particular, we derived new algorithms and showed their convergence. By introducing and analyzing the $\kappa\lambda$-PI, we demonstrated that the κ-PI can be used with a 'relaxed' value estimation and that the effects of noise are controlled. Last but not least, by making connections with some recent empirical works (Schulman et al., 2015b; Silver et al., 2017b), our work sheds some light on the reasons of their impressive success. We conducted simulations on a toy example and have shown how a generic RL algorithm (VI) can be used for the greedy step of the κ- and h-PI frameworks. We empirically demonstrated that such a new algorithm leads to a performance improvement. By using other techniques for solving the greedy step (see in particular Remarks 1 and 4), many new algorithms may be built.

It may be interesting to consider online versions (i.e. stochastic approximations) of the algorithmic schemes we have introduced here, which would probably require to consider at least two time scales (one for the inner surrogate problems, one for the main loop). On the theoretical side, our first attempts in this direction suggest that previous approaches for proving online convergence of PI (Konda & Borkar, 1999; Kakade & Langford, 2002) may not be so straightforwardly generalized. We are currently working on this extension.

A potential practical extension of this work would be to consider a state-dependent κ value, that is a function $\kappa : \mathcal{S} \to [0, 1]$. Though the details of such a generalization need to be written carefully, we believe that the machinery we developped here will still hold. We expect the convergence to be assured with rates that would intricately depend on this κ function. Using such an approach, the algorithm designer could put more 'prior knowledge' into the learning phase. In general though, understanding better when the choice of a κ function would be good or not (and even understanding how to choose the κ or h parameters of the algorithms described here) is intriguing and deserves future investigation.

A. Proof of Theorem 9

We might follow the steps in (Scherrer & Thiery, 2010; Scherrer et al., 2015). But we give here an alternative and shorter proof, which is new even in the specific context of $\kappa = 0$.

Using (15), for any π, we have

$$\begin{aligned} T_\kappa^\pi v_1 - T_\kappa^\pi v_2 &= \gamma(1-\kappa)(I-\kappa\gamma P^\pi)^{-1}(v_1 - v_2) \\ &= \xi D_\kappa^\pi P^\pi (v_1 - v_2) \stackrel{\text{def}}{=} \xi \tilde{P}^\pi (v_1 - v_2), \end{aligned}$$

where we defined $D_\kappa^\pi = (1-\kappa\gamma)(I-\kappa\gamma P^\pi)^{-1}$, a stochastic matrix, and $\tilde{P}_\kappa^\pi = D_\kappa^\pi P^\pi$, also a stochastic matrix.

Define the following alternative error $\epsilon'_k = \epsilon_k - C_k e$, where $C_k = \frac{\max \delta_{k+1} + \max \epsilon_k - \xi \min \epsilon_k}{1-\xi}$ and e is the constant vector made of ones. Since for all α, $T_\kappa^\pi(v + \alpha e) = T_\kappa^\pi v + \xi \alpha e$ and $\hat{\mathcal{G}}_\kappa^\delta(v) = \hat{\mathcal{G}}_\kappa^\delta(v + \alpha e)$, the sequence of policies that can be generated by the original algorithm, with error ϵ_k, is the same as that that would be generated by the algorithm with erros ϵ'_k. From now on, let us consider the latter.

With a similar invariance argument, let us assume, without loss of generality, that $v_0 - T_\kappa^{\pi_1} v_0 \le 0$. Then, one can see that for all $k \ge 1$,

$$\begin{aligned} b_k &\stackrel{\text{def}}{=} v_k - T_\kappa^{\pi_{k+1}} v_k \\ &\le v_k - T_\kappa^{\pi_k} v_k + \delta_{k+1} \\ &= (v_k - \epsilon'_k) - T_\kappa^{\pi_k}(v_k - \epsilon'_k) + (1-\xi)\epsilon'_k + \delta_{k+1} \\ &\le T_\lambda^{\pi_k} v_{k-1} - T_\kappa^{\pi_k} T_\lambda^{\pi_k} v_{k-1} \\ &\quad + (\max \delta_{k+1} + \max \epsilon'_k - \xi \min \epsilon'_k) e \\ &= T_\lambda^{\pi_k} v_{k-1} - T_\lambda^{\pi_k} T_\kappa^{\pi_k} v_{k-1} \\ &\quad + \underbrace{(\max \delta_{k+1} + \max \epsilon_k - C_k - \xi(\min \epsilon_k - C_k))}_{0} e \\ &= \xi_\lambda \tilde{P}_\lambda^{\pi_k}(v_{k-1} - T_\kappa^{\pi_k} v_{k-1}) = \xi_\lambda \tilde{P}_\lambda^{\pi_k} b_{k-1}. \end{aligned}$$

In the fifth relation we used Proposition 13 (see Appendix B) that shows $T_\kappa T_\lambda = T_\lambda T_\kappa$, and defined $\xi_\lambda \stackrel{\text{def}}{=} \gamma \frac{1-\lambda}{1-\lambda\gamma}$. Thus, since $\tilde{P}_\lambda^{\pi_k}$ has only non-negative elements, by induction, we have $b_k \le 0$ for all k.

Then, since $(1-\bar{\lambda})\sum_{j=0}^\infty \bar{\lambda}^j = 1$, one can see that

$$\begin{aligned} d_{k+1} &\stackrel{\text{def}}{=} v_* - (v_{k+1} - \epsilon'_{k+1}) \\ &= v^* - (1-\bar{\lambda}) \sum_{j=0}^\infty \bar{\lambda}^j (T_\kappa^{\pi_{k+1}})^{j+1} v_k \\ &= (1-\bar{\lambda}) \sum_{j=0}^\infty \bar{\lambda}^j (v^* - (T_\kappa^{\pi_{k+1}})^{j+1} v_k). \end{aligned}$$

Each term in the sum satisfies:

$$\begin{aligned} & v^* - (T_\kappa^{\pi_{k+1}})^{j+1} v_k \\ &= T_\kappa^{\pi^*} v^* - T_\kappa^{\pi^*} v_k + T_\kappa^{\pi^*} v_k \\ &\quad - T_\kappa^{\pi_{k+1}} v_k + \sum_{i=1}^j (T_\kappa^{\pi_{k+1}})^i v_k - (T_\kappa^{\pi_{k+1}})^{i+1} v_k \\ &\le \xi \tilde{P}_\kappa^{\pi^*}(v^* - v_k) + \delta_{k+1} + \sum_{i=1}^j (\xi \tilde{P}_\kappa^{\pi_{k+1}})^i b_k \\ &\le \xi \tilde{P}_\kappa^{\pi^*}(v^* - v_k) + \delta_{k+1}, \end{aligned}$$

and hence we deduce that

$$\begin{aligned} d_{k+1} &\le \xi \tilde{P}_\kappa^{\pi^*}(v^* - v_k) + \delta_{k+1} \\ &= \xi \tilde{P}_\kappa^{\pi^*} d_k - \xi \tilde{P}_\kappa^{\pi^*} \epsilon'_k + \delta_{k+1} \end{aligned}$$

and thus

$$\begin{aligned} & \max d_{k+1} \\ &\le \xi \max d_k + \max \delta_{k+1} - \xi \min \epsilon'_k \\ &= \xi \max d_k + \max \delta_{k+1} \\ &\quad - \xi \left(\min \epsilon_k - \frac{\max \delta_{k+1} + \max \epsilon_k - \xi \min \epsilon_k}{1-\xi} \right) \\ &= \xi \max d_k + \frac{\max \delta_{k+1} + \xi(\max \epsilon_k - \min \epsilon_k)}{1-\xi}, \end{aligned}$$

which eventually leads to

$$\lim_{k\to\infty} d_k \le \frac{\xi(\max \epsilon_k - \min \epsilon_k) + \max \delta_{k+1}}{(1-\xi)^2}.$$

To conclude, we finally observe that

$$\begin{aligned} s_k &\stackrel{\text{def}}{=} v_{k+1} - \epsilon_{k+1} - v_{\pi^{k+1}} \\ &= (1-\bar{\lambda}) \sum_{j=0}^\infty \bar{\lambda}^j (T_\kappa^{\pi_{k+1}})^{j+1} v_k - v^{\pi_{k+1}} \\ &= (1-\bar{\lambda}) \sum_{j=0}^\infty \bar{\lambda}^j ((T_\kappa^{\pi_{k+1}})^{j+1} v_k - v^{\pi_{k+1}}), \end{aligned}$$

and each term of the sum satisfies:

$$\begin{aligned} & (T_\kappa^{\pi_{k+1}})^{j+1} v_k - v^{\pi_{k+1}} \\ &= \sum_{i \ge j+1} (T_\kappa^{\pi_{k+1}})^i v_k - (T_\kappa^{\pi_{k+1}})^{i+1} v_k \\ &= \sum_{i \ge j+1} (\xi \tilde{P}_\kappa^{\pi_{k+1}})^i b_k \le 0. \end{aligned}$$

In other words, we deduce that s_k is non-positive. The result follows from the fact that $v_* - v_{\pi_k} = d_k + s_k$ and that $\max \epsilon_k - \min \epsilon_k \le 2\|\epsilon_k\|$.

Acknowledgments

We thank Timothy Mann and Guy Tennenholtz for helpful discussions and the reviewers for their comments. We also thank Liyu Chen for pointing to an incomplete proof which is now fixed. This work was partially funded by the Israel Science Foundation under contract 1380/16 and by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement 306638 (SUPREL)

References

Bertsekas, D. P. Lambda-policy iteration: A review and a new implementation. *arXiv preprint arXiv:1507.01029*, 2015.

Bertsekas, D. P. and Ioffe, S. Temporal differences-based policy iteration and applications in neuro-dynamic programming. 1996.

Bertsekas, D. P. and Tsitsiklis, J. N. Neuro-dynamic programming: an overview. In *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, volume 1, pp. 560–564. IEEE, 1995.

Bouzy, B. and Helmstetter, B. Monte-carlo go developments. In *Advances in computer games*, pp. 159–174. Springer, 2004.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

Busoniu, L. and Munos, R. Optimistic Planning for Markov Decision Processes. In *15th International Conference on Artificial Intelligence and Statistics, AISTATS-12*, volume 22 of *Journal of Machine Learning Research: Workshop and Conference Proceedings*, pp. 182–189, La Palma, Canary Islands, Spain, April 2012. URL https://hal.archives-ouvertes.fr/hal-00756736.

François-Lavet, V., Fonteneau, R., and Ernst, D. How to discount deep reinforcement learning: Towards new dynamic strategies. *arXiv preprint arXiv:1512.02011*, 2015.

Grill, J.-B., Valko, M., and Munos, R. Blazing the trails before beating the path: Sample-efficient Monte-Carlo planning. In *Neural Information Processing Systems*, Barcelona, Spain, December 2016. URL https://hal.inria.fr/hal-01389107.

Hren, J.-F. and Munos, R. Optimistic planning of deterministic systems. In *European Workshop on Reinforcement Learning*, pp. 151–164, France, 2008. URL https://hal.archives-ouvertes.fr/hal-00830182.

Jiang, N., Kulesza, A., Singh, S., and Lewis, R. The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pp. 1181–1189. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

Kakade, S. and Langford, J. Approximately Optimal Approximate Reinforcement Learning. In *International Conference on Machine Learning*, pp. 267–274, 2002.

Konda, V. R. and Borkar, V. S. Actor-critic–type learning algorithms for markov decision processes. *SIAM Journal on control and Optimization*, 38(1):94–123, 1999.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pp. 1928–1937, 2016.

Munos, R. From Bandits to Monte-Carlo Tree Search: The Optimistic Principle Applied to Optimization and Planning. Technical report, 2014. URL https://hal.archives-ouvertes.fr/hal-00747575. 130 pages.

Ng, A. Y. Shaping and policy search in reinforcement learning. pp. 48–52, 2003.

Petrik, M. and Scherrer, B. Biasing approximate dynamic programming with a lower discount factor. In *Advances in neural information processing systems*, pp. 1265–1272, 2009.

Puterman, M. L. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 1994.

Puterman, M. L. and Shin, M. C. Modified policy iteration algorithms for discounted markov decision problems. *Management Science*, 24(11):1127–1137, 1978.

Scherrer, B. Performance Bounds for Lambda Policy Iteration and Application to the Game of Tetris. *Journal of Machine Learning Research*, 14:1175–1221, January 2013a. URL https://hal.inria.fr/hal-00759102.

Scherrer, B. Performance bounds for λ policy iteration and application to the game of tetris. *Journal of Machine Learning Research*, 14(Apr):1181–1227, 2013b.

Scherrer, B. Improved and Generalized Upper Bounds on the Complexity of Policy Iteration. *Mathematics of Operations Research*, February 2016. doi: 10.1287/

moor.2015.0753. URL https://hal.inria.fr/hal-00829532. Markov decision processes ; Dynamic Programming ; Analysis of Algorithms.

Scherrer, B. and Thiery, C. Performance bound for Approximate Optimistic Policy Iteration. Technical report, 2010. URL https://hal.inria.fr/inria-00480952.

Scherrer, B., Ghavamzadeh, M., Gabillon, V., Lesner, B., and Geist, M. Approximate Modified Policy Iteration and its Application to the Game of Tetris. *Journal of Machine Learning Research*, 16:1629–1676, 2015. URL https://hal.inria.fr/hal-01091341. A paraître.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015a.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

Seijen, H. and Sutton, R. True online td (lambda). In *International Conference on Machine Learning*, pp. 692–700, 2014.

Sheppard, B. World-championship-caliber scrabble. *Artificial Intelligence*, 134(1-2):241–275, 2002.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017a.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017b.

Strehl, A. L., Li, L., and Littman, M. L. Reinforcement learning in finite mdps: Pac analysis. *Journal of Machine Learning Research*, 10(Nov):2413–2444, 2009.

Sutton, R. S. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Szörényi, B., Kedenburg, G., and Munos, R. Optimistic planning in Markov decision processes using a generative model. In *Advances in Neural Information Processing Systems 27*, Montréal, Canada, December 2014. URL https://hal.inria.fr/hal-01079366.

Tesauro, G. and Galperin, G. R. On-line policy improvement using monte-carlo search. In *Advances in Neural Information Processing Systems*, pp. 1068–1074, 1997.

Veness, J., Silver, D., Blair, A., and Uther, W. Bootstrapping from game tree search. In *Advances in neural information processing systems*, pp. 1937–1945, 2009.