A Tighter Analysis of Randomised Policy Iteration

Meet Taraviya and **Shivaram Kalyanakrishnan**
Department of Computer Science and Engineering
Indian Institute of Technology Bombay
{mtaraviya, shivaram}@cse.iitb.ac.in

Abstract

Policy Iteration (PI) is a popular family of algorithms to compute an optimal policy for a given Markov Decision Problem (MDP). Starting from an arbitrary initial policy, PI repeatedly performs locally-improving switches until an optimal policy is found. The exact form of the switching rule gives rise to different variants of PI. Two decades ago, Mansour and Singh [1999] provided the first non-trivial "strong" upper bound on the number of iterations taken by "Howard's PI" (HPI), a widely-used variant of PI (strong bounds depend only on the number of states and actions in the MDP). They also proposed a randomised variant (RPI) and showed an even tighter strong upper bound. Their bounds for HPI and RPI have not been improved subsequently.

We revisit the algorithms and analysis of Mansour and Singh [1999]. We prove a novel result on the structure of the policy space for k-action MDPs, $k \geq 2$, which generalises a result known for $k = 2$. Also proposing a new counting argument, we obtain a strong bound of $(O(\sqrt{k \log k}))^n$ iterations for an algorithm akin to RPI, improving significantly upon Mansour and Singh's original bound of roughly $O((k/2)^n)$. Similar analysis of a randomised variant of HPI also yields a strong upper bound of $(O(\sqrt{k \log k}))^n$ iterations, registering the first exponential improvement for HPI over the trivial bound of k^n. Our other contributions include a lower bound of $\Omega(n)$ iterations for RPI and an upper bound of 1.6001^n iterations for a randomised variant of "Batch-Switching PI" [Kalyanakrishnan *et al.*, 2016a] on 2-action MDPs—the tightest strong upper bound shown yet for the PI family.

1 INTRODUCTION

Markov Decision Problems (MDPs) [Bellman, 1957; Puterman, 1994] are an abstraction of sequential decision making, providing a formal basis for problems such as automated planning and reinforcement learning. Applications of MDPs span a variety of domains [White, 1985; White, 1988; Feinberg and Schwartz, 2002].

An MDP describes an agent's *environment*. For each state $s \in S$ in which the agent can be, and for each action $a \in A$ that it can take, the MDP specifies for all $s' \in S$ the probability that taking a from s will reach s'. Denote this probability$P(s, a, s')$. The transition from s to s' by taking action a also yields a numeric reward $R(s, a, s')$.

The agent's behaviour is encoded as a *policy* $\pi : S \to A$, which specifies the action $\pi(s)$ that the agent must take from each state $s \in S$. If indeed the agent starts from some state $s^0 \in S$ and follows π, then it encounters a random "state-reward" sequence $s^0, r^0, s^1, r^1, \ldots$ over time, wherein for $t \geq 0$, s^{t+1} is drawn according to $P(s^t, \pi(s^t), \cdot)$, and $r^t = R(s^t, \pi(s^t), s^{t+1})$. The *value* of s^0 is commonly defined to be $\mathbb{E}[r^0 + \gamma r^1 + \gamma^2 r^2 + \ldots]$, where $\gamma \in [0, 1)$ is a discount factor. An MDP is fully specified by S, A, P, R, and γ. In this paper, we shall assume that S and A are both finite.

Our definition above interprets value as "infinite discounted reward". The analysis provided in our paper can also be applied if alternative definitions such as "average reward" [Mahadevan, 1996] and "total reward" [Fearnley, 2010] are used. Also, it suffices for our purposes to consider policies that do not vary over time, and which map states to actions (rather than map histories to distributions over actions). In other words, we consider policies that are stationary, deterministic, and Markovian.

Let (S, A, P, R, γ) be an arbitrary MDP, and Π be the set of all policies for this MDP. It is a well-known result that there is an *optimal policy* $\pi^\star \in \Pi$ such that for all $\pi \in \Pi$,

$s \in S$, $V^{\pi^\star}(s) \geq V^{\pi}(s)$. The problem we consider in this paper is precisely that of computing an optimal policy for a given MDP. We assume that P and R are provided as tables; there is no need for sampling (as is typical in reinforcement learning [Sutton and Barto, 1998]) or for any sort of generalisation (as is needed when S or A are very large [Bertsekas and Tsitsiklis, 1996]). Over the decades, many families of algorithms have been proposed to solve the problem we consider, which is denoted MDP planning. Approaches vary from formulations using Linear Programming to dynamic programming techniques such as Value Iteration, Policy Iteration, and combinations [Littman *et al.*, 1995].

In this paper, we consider the Policy Iteration (PI) family of algorithms [Howard, 1960]. PI is a conceptually-simple, iterative approach to search the space of policies. Although experimenters tend to find PI to work very well in practice [Littman *et al.*, 1995, see Section 4.2], there has only been limited progress towards formally quantifying its running time. We restrict our focus to *strong* running-time bounds—those that depend only on the number of states and actions in the given MDP. Strong bounds have long held appeal from a theoretical standpoint [Megiddo, 1982]. Suppose we assume that arithmetic, relational and logical operations can all be performed exactly, in constant time, regardless of the operand size, the question is how many operations are needed to compute an optimal policy. The number of operations performed by PI is known to be polynomial in the number of states and actions if associated parameters such as the discount rate γ and the number of bits B needed to represent the MDP are treated as constants [Ye, 2011; Scherrer, 2013]. Strong bounds, on the other hand, have no dependence on parameters such as γ and B.

The first non-trivial strong bounds for PI were provided by Mansour and Singh [1999], who showed an improved upper bound on the running time of Howard's PI (a commonly-used variant), and proposed a randomised variant of PI with a tighter upper bound. More recently, even tighter strong upper bounds have been shown for newer variants of PI—both deterministic [Gupta and Kalyanakrishnan, 2017] and randomised [Kalyanakrishnan *et al.*, 2016b].

Our main contributions are based on the algorithms and analysis of Mansour and Singh [1999]. Applying some fresh ideas in conjunction with their proof structure, we show significantly tighter upper bounds for related randomised algorithms, including a variant of Howard's PI, on k-action MDPs, $k \geq 3$. We provide a *lower* bound for the randomised variant of Mansour and Singh [1999], matching a bound shown previously for Howard's PI. We also investigate a randomised "batch-switching" variant of PI [Kalyanakrishnan *et al.*, 2016a]. While the (strong) upper bound we furnish for this variant is the tightest yet for the PI family, experiments suggest that the randomised algorithm of Mansour and Singh [1999] might itself be more efficient.

We present our technical contributions in sections 4 and 5, after first describing PI in Section 2 and surveying previous analyses in Section 3. In Section 6 we share experimental findings to accompany our theoretical results. We conclude with a discussion in Section 7.

2 POLICY ITERATION

In this section, we formalise Policy Iteration (PI), borrowing notation and definitions from Mansour and Singh [1999]. We assume that the set of states S and the set of actions A are finite, with $|S| = n \geq 1$ and $|A| = k \geq 2$. Specifically, we take $A = \{0, 1, \ldots, k-1\}$.

Policy evaluation is a basic step that is used in PI and many other approaches for MDP planning. Given a policy $\pi \in \Pi$, observe that state values (together the *value function* V^{π}) satisfy a recursive relation: for $s \in S$,

$$V^{\pi}(s) = \sum_{s' \in S} P(s, \pi(s), s')(R(s, \pi(s), s') + \gamma V^{\pi}(s')).$$

Hence, a given policy π can be *evaluated* (that is, V^{π} computed) by solving a system of linear equations.

For policies $\pi, \pi' \in \Pi$, we write $\pi \succeq \pi'$ if for all $s \in S$, $V^{\pi}(s) \geq V^{\pi'}(s)$. If $\pi \succeq \pi'$, and in addition, for some $s \in S$, $V^{\pi}(s) > V^{\pi'}(s)$, then we write $\pi \succ \pi'$. Observe that if $\pi \succeq \pi'$ and $\pi' \succeq \pi$, then V^{π} and $V^{\pi'}$ must be equal, in which case we write $\pi \approx \pi'$. We find it convenient to distinguish between such "equally-good" policies by using an arbitrary total order L on Π. Since policies can be represented as n-length k-ary strings from $\{0, 1, \ldots, k-1\}^n$, we take that for $\pi, \pi' \in \Pi$, $\pi L \pi'$ if and only if π lexicographically precedes or equals π'. We define $\pi \gtrsim \pi'$ if (1) $\pi \succ \pi'$ or (2) $\pi \approx \pi'$ and $\pi L \pi'$. By this definition, observe that there is a *unique* optimal policy $\pi^\star$ such that for all $\pi \in \Pi$, $\pi^\star \gtrsim \pi$. Our algorithms will be designed to find this policy.

A naïve way to find $\pi^\star$ would be to evaluate each of the k^n policies in Π and then compare them. As we see next, the PI family of algorithms exploits an interesting structure of the policy space to find $\pi^\star$ more efficiently.

The *action value function* Q^{π} for a policy $\pi \in \Pi$ provides for each $s \in S$, $a \in A$ the expected long-term reward obtained by taking a from s for a single time step, and thereafter acting according to π. It follows that

$$Q^{\pi}(s,a) = \sum_{s' \in S} P(s,a,s')(R(s,a,s') + \gamma V^{\pi}(s')).$$

Now, define T^{π} as follows:

$$\begin{aligned} T^{\pi} =& \{(s,a) | Q^{\pi}(s,a) > V^{\pi}(s)\} \cup \\ & \{(s,a) | Q^{\pi}(s,a) = V^{\pi}(s) \text{ and } a < \pi(s)\}. \end{aligned}$$

If $(s,a) \in T^{\pi}$, then s is termed an "improvable state" for π and a an "improving action" at s for π. For fixed π, let $\mathsf{states}(T^{\pi})$ denote the set of all improvable states $s \in S$, and $T^{\pi}(s)$ denote the set of all improving actions for fixed $s \in S$, with the convention that $T^{\pi}(s) = \emptyset$ if $s \notin \mathsf{states}(T^{\pi})$. Now, if $|T^{\pi}| > 0$, let $U \subseteq T^{\pi}$ be such that $|U| \geq 1$, and no two distinct elements (s,a) and $(s',a') \in U$ have $s = s'$. In other words, U collects a subset of improvable states—denoted $\mathsf{states}(U)$—and exactly one improving action for each such state. In general, there can be many choices of U that satisfy this property for T^{π}. For some fixed U, let $\mathsf{modify}(\pi, U)$ denote the policy π' such that for all $(s,a) \in U$, $\pi'(s) = a$ and for all $s \in S \setminus \mathsf{states}(U)$, $\pi'(s) = \pi(s)$. We collect all such policies π', each derived from a different choice of U, in a set $I(\pi)$: the set of (locally) "improving" policies of π. Observe that $|I(\pi)| = \prod_{s \in S}(|T^{\pi}(s)| + 1) - 1$, and so $I(\pi)$ is empty if and only if T^{π} is empty. The Policy Improvement Theorem, provided below, is a well-known result highlighting the relevance of $I(\pi)$. We omit the proof, which is provided by several other sources [Szepesvári, 2010; Bertsekas, 2012; Kalyanakrishnan *et al.*, 2016b].

Theorem 1. *For $\pi \in \Pi$: (1) if $T^{\pi} = \emptyset$, then for all $\pi' \in \Pi$, $\pi \succsim \pi'$; (2) else for all $\pi' \in I(\pi)$, $\pi' \succsim \pi$.*

The theorem allows us to test if a given policy π is optimal, and if it is not, to update to a dominating policy π'. The PI family of algorithms is based on repeatedly performing such updates until eventually, an optimal policy is reached. Given $\pi \in \Pi$, observe that V^{π} and Q^{π}, and therefore T^{π}, can be computed efficiently—using poly(n,k) arithmetic and comparison operations. The complexity of PI is therefore determined primarily by the number of iterations performed to reach $\pi^{\star}$. Algorithms in the PI family are set apart by their "switching rules": how they pick $U \subseteq T^{\pi}$ for setting $\pi' = \mathsf{modify}(\pi, U)$. In general, these algorithms can be randomised. We discuss several PI variants in the next section, but before proceeding, present two related ideas.

First, it is convenient for our analysis to also consider the "opposite" of policy-improvement: only switching to actions that are *not* improving. The following corollary is easily proven in the same manner as Theorem 1.

Corollary 2. *For $\pi, \pi' \in \Pi$, suppose for all $s \in S$: $(\pi'(s) \neq \pi(s)) \implies (\pi'(s) \notin T^{\pi}(s))$. Then $\pi \succsim \pi'$.*

Second, it is worth noting that while our primary focus is the application of PI to MDPs, the upper bounds we show in Section 4, and indeed those originally given by Mansour and Singh [1999], also hold for solving a class of discrete objects called Acyclic Unique Sink Orientations (AUSOs) [Stickney and Watson, 1978; Szabó and Welzl, 2001]. Now, it follows from Theorem 1 and Corollary 2 that any two policies π and π' that differ in exactly one state must satisfy either $\pi \succsim \pi'$ or $\pi' \succsim \pi$. Thus, in particular, policies for 2-state MDPs can be arranged as vertices of an n-dimensional hypercube, with edges (1) connecting policies that differ in exactly one state, and (2) oriented according to $\succsim$. Each face of the hypercube is guaranteed to have a unique sink and no cycles, making the hypercube an AUSO [Gupta and Kalyanakrishnan, 2017]. Figures 1(a)–1(d) illustrate the relationship between 2-action MDPs and AUSOs with an example.

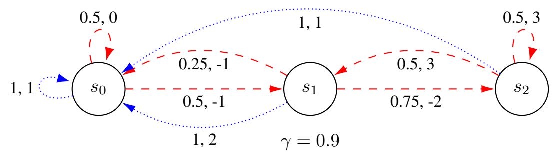
Solving a given AUSO amounts to identifying its sink. For so doing, an algorithm may *evaluate* vertices one at a time, thereby discovering their outgoing edges. In this context, PI would begin with an arbitrary vertex u, and repeatedly update to some vertex v in the subface formed by the outgoing edges of u (thus v "locally improves" upon u). If u has no outgoing edges, it must be the sink.

Interestingly, it can be shown that AUSOs resulting from MDPs must additionally satisfy the *Holt-Klee conditions* [Holt and Klee, 1999], which are: for an n-dimensional AUSO, every d-dimensional face, $1 \leq d \leq n$, should have at least d vertex-disconnected paths from source to sink. The Holt-Klee conditions hold for all AUSOs induced by Linear Programs, and can be shown for MDPs by considering their Linear programming formulation [Post and Ye, 2013]. In the example from Figure 1(d), notice that from the source 001, there are paths through 000 and 100; 011 and 010; and also 101 and 111 to the sink 110 . All 2- and 1-dimensional AUSOs necessarily satisfy the Holt-Klee conditions.

3 RELATED WORK

In this section, we survey results on the complexity of different variants of PI. We only consider strong bounds—which solely depend on the number of states n and the number of actions k in the MDP. Note that polynomial upper bounds in n and k exist for variants of PI if dependence on additional parameters such as the discount factor γ and the representation size B are permitted [Ye, 2011; Scherrer, 2013]. Also note that the LP route for MDP planning can yield a strong upper bound of poly$(n,k)\cdot\exp(O(\sqrt{n\log n}))$ [Matoušek *et al.*, 1996] operations. Strong upper bounds known yet for PI are all exponential in n. Since policy evaluation is polynomial in n and k, the bounds we discuss below are on the number of policy evaluations performed by PI.

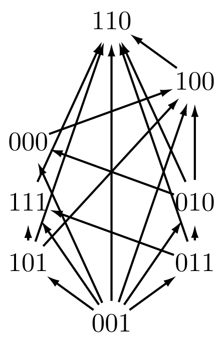
We claim $u(n,k)$ as an upper bound for some variant if for *every* n-state, k-action MDP, from *every* starting policy, the (expected) number of policy evaluations performed is at most $u(n,k)$ (expectation needed only for randomised variants). On the other hand, $l(n,k)$ is a lower bound for some algorithm if there exist an n-state, k-action MDP and a starting policy from which the (expected) number of policy evaluations performed is at least $l(n,k)$. Table 1 summarises existing upper bounds side-by-side with our improvements, which we proceed to present. Assume $\pi\in\Pi$ is the policy being considered for improvement and U the subset of T^π to be used for switching—to obtain $\pi' = \mathsf{modify}(\pi, U)$.



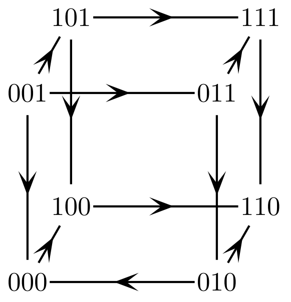
(a)

π	$V^\pi(s_0)$	$V^\pi(s_1)$	$V^\pi(s_2)$	T^π
000	4.45	6.55	10.82	$\{(s_0,1)\}$
001	-5.61	-5.74	-4.05	$\{(s_0,1),(s_1,1),(s_2,0)\}$
010	2.76	4.48	9.12	$\{(s_0,1),(s_1,0)\}$
011	2.76	4.48	3.48	$\{(s_0,1),(s_2,0)\}$
100	10.00	9.34	13.10	$\{(s_1,1)\}$
101	10.00	7.25	10.00	$\{(s_1,1),(s_2,0)\}$
110	10.00	11.00	14.45	$\{\}$
111	10.00	11.00	10.00	$\{(s_2,0)\}$

(b)



(c)



(d)

Figure 1: (a) Example of a 3-state 2-action MDP. Red (dashed) and blue (dotted) edges correspond to actions 0 and 1, respectively. Each arrow shows the corresponding transition probability and reward. (b) V^π and T^π for all $\pi\in\Pi$. (c) Graph with each policy π as a vertex, with edges leading to all policies $\pi'\in I(\pi)$. (d) Induced AUSO (details in text).

Table 1: Upper bounds for PI variants for general k. References and descriptions of the algorithms are given in the text. Except the entry marked "D", all the results from this paper correspond to randomised variants, of which some are obtained by specialising or slightly altering the original variant.

Variant	**Previous**	**This paper**
HPI	$O(\frac{k^n}{n})$	$(O(k\log k))^{n/2}$
RPI	$O(((1+\frac{2}{\log_2 k})\frac{k}{2})^n)$	$(O(k\log k))^{n/2}$
BSPI	$k^{0.7207n}$	$k^{0.7019n}$ [D] $k^{0.6782n}$
RSPI	$(O(\log k))^n$	–

Howard's PI. The earliest variant of PI, introduced by Howard [1960], is also the most commonly used. Howard's PI (HPI) dictates that *every* improvable state be switched; in other words, *States*(U) = *States*(T^π). For 2-action MDPs, this description fixes the switching rule, since $|T^\pi(s)|\leq 1$ for every state s. If $k\geq 3$, there might be multiple improving actions for some state—in which case we may pick an *arbitrary* one for switching. The tightest upper bound on the number of iterations taken by HPI is $O(k^n/n)$ [Mansour and Singh, 1999]; the multiplicative factor has subsequently been improved [Hollanders *et al.*, 2014]. Mansour and Singh's analysis partitions Π into a set of "large-improvement" policies and a set of "small-improvement" policies. For large-improvement policies, defined as those for which $|\mathsf{states}(T^\pi)|$ exceeds a threshold m, a structural argument shows that π' will dominate or be incomparable to at least m policies that themselves dominate π. Since each large-improvement policy therefore eliminates m policies, and since the number of small-improvement policies can be upper-bounded in terms of m, tuning m yields an overall bound of $O(k^n/n)$ iterations.

Randomised PI. Mansour and Singh [1999] also propose a randomised variant of PI (RPI), in which $\mathsf{states}(U)$ is picked uniformly at random from the non-empty subsets of $\mathsf{states}(T^\pi)$. Again, if $k\geq 3$, improving actions can be picked arbitrarily. In this case, it can be shown that visits to large-improvement policies eliminate $\Theta(2^m)$ policies in expectation, leading to an overall bound of $O(((1+2/\log_2 k)(k/2))^n)$ iterations. For the special case of $k=2$, Mansour and Singh [1999] provide a a tighter bound of $O(1.7172^n)$ iterations.

Batch-switching PI. In a relatively recent line of work, Kalyanakrishnan et al. [2016a] propose a scheme to translate upper bounds on the complexity of PI for small (constant-size) MDPs to ones for general MDPs. Arguing essentially based on the policy improvement theorem, they demonstrate that HPI can take at most 33 iterations on 7-state, 2-action MDPs. This bound comes from a relaxation called the "order regularity" problem [Gerencsér *et al.*, 2015]. Batch-switching PI (BSPI) is a variant of PI in which states only within a fixed-size batch of states are switched at each iteration. Assuming that batches are indexed, the batch with the highest index among those with improvable states is picked. For a batch size of 7, a recursive argument establishes that BSPI can take at most $33^{n/7} < 1.6479^n$ iterations on 2-action MDPs. The numerical computation of the bound for batch sizes 8 and higher has not been feasible, although evidence suggests that larger batch sizes might be even more economical. If such a trend is indeed true, then HPI—which can be construed as BSPI with a batch size of n—would itself enjoy an upper bound of 1.6479^n iterations.

Gupta and Kalyanakrishnan [2017] augment BSPI with a layer of recursion over *actions*, extending the bound of 1.6479^n iterations for 2-action MDPs to $k^{(\log_2 1.6479)n} < k^{0.7207n}$ iterations for k-action MDPs, $k \geq 2$. While their algorithm is deterministic, the currently-tightest upper bound for PI on k-action MDPs is for a randomised variant of "Simple PI" [Melekopoglou and Condon, 1994], which is the same as BSPI with a batch size of 1. Crucial to the analysis of this algorithm, RSPI [Kalyanakrishnan *et al.*, 2016b], is that an improving action be picked *uniformly at random* from those available for the chosen state. The resulting upper bound is $(2 + \ln(k-1))^n$.

Lower bounds. Interestingly, on 2-action MDPs, Simple PI can take as many as 2^n iterations [Melekopoglou and Condon, 1994]. However, only a lower bound of $\Omega(n)$ iterations has been shown on 2-action MDPs for HPI [Hansen and Zwick, 2010]. Exponential bounds have been shown for HPI on MDPs when the number of actions can depend linearly on the number of states [Fearnley, 2010; Hollanders *et al.*, 2012]. Schurr and Szabó [2005] also show an exponential lower bound for HPI on AUSOs. The AUSOs they construct do not satisfy the Holt-Klee conditions (and therefore do not originate from MDPs).

Our contributions. We make five contributions to the analysis of PI.

1. We show that a slight modification to the RPI algorithm of Mansour and Singh [1999] results in an upper bound $(O(\sqrt{k \log k}))^n$ iterations—significantly tighter than the authors had originally shown. Our proof rests on a key structural property of k-action MDPs and also a new counting argument.

2. We propose a randomised variant of HPI (switch *all* improvable states; select improving actions uniformly at random) that achieves an $(O(\sqrt{k \log k}))^n$ upper bound. This becomes the first exponential improvement for HPI over the trivial bound of k^n.

3. Using a search by computer program, we show that BSPI [Kalyanakrishnan *et al.*, 2016a], if implemented with RPI in place of HPI within each batch, achieves an upper bound of 1.6001^n iterations for 2-action MDPs. This bound is the tightest yet shown for the PI family on 2-action MDPs. Our search also uncovers a bound of 1.6266^n iterations for HPI-based BSPI on 2-action MDPs. This bound translates to $k^{0.7019n}$ for k-action MDPs, and is the tightest bound yet for a deterministic PI variant.

4. Using the MDP construction of the Melekopoglou and Condon [1994], we show an $\Omega(n)$ lower bound for RPI on 2-action MDPs, matching the one shown by Hansen and Zwick [2010] for HPI.

5. We present an experimental comparison of the different PI variants analysed in this paper. Our results show many interesting trends that are not explained by the current theory, and motivate further analysis.

We proceed to our contributions.

4 UPPER BOUNDS

In this section, we present new upper bounds on three variants of PI. We begin by noting that for analysing RPI, Mansour and Singh [1999] deal separately with the cases of $k = 2$ and $k \geq 3$. The main reason for this bifurcation is their use of a specific structural property for 2-action MDPs. We begin by generalising this property for all $k \geq 2$. Consequently our analysis does not need separate cases, and also yields tighter bounds for $k \geq 3$.

Lemma 3. *For policies* $\pi_1, \pi_2 \in \Pi$, *if* $|T^{\pi_1}(s)| = |T^{\pi_2}(s)|$ *for all states* $s \in S$, *then* $\pi_1 = \pi_2$.

Proof. Assume that $|T^{\pi_1}(s)| = |T^{\pi_2}(s)|$ for some policies π_1 and π_2, for all states s. Now, for each state s, note that $A \setminus T^{\pi_1}(s)$ and $T^{\pi_2}(s) \cup \{\pi_2(s)\}$ cannot be disjoint, since that would imply $(T^{\pi_2}(s) \cup \{\pi_2(s)\}) \subseteq T^{\pi_1}(s)$, which cannot be true since $|T^{\pi_2}(s) \cup \{\pi_2(s)\}| = 1 + |T^{\pi_1}(s)|$. Hence, we may construct a policy π_3 such that for each state s, $\pi_3(s) \in (A \setminus T^{\pi_1}(s)) \cap (T^{\pi_2}(s) \cup \{\pi_2(s)\})$. By Theorem 1, it follows that $\pi_3 \gtrsim \pi_2$. Similarly, by Corollary 2, it must be that $\pi_1 \gtrsim \pi_3$. Hence,

$\pi_1 \gtrsim \pi_3 \gtrsim \pi_2$. Now, if we construct a policy π_4 such that for each state s, $\pi_4(s) \in (A \setminus T^{\pi_2}(s)) \cap (T^{\pi_1}(s) \cup \{\pi_1(s)\})$, a similar argument yields $\pi_2 \gtrsim \pi_4 \gtrsim \pi_1$. Since $\pi_2 \gtrsim \pi_1$ and $\pi_1 \gtrsim \pi_2$, we get $\pi_1 = \pi_2$. □

The lemma establishes that for a given policy $\pi \in \Pi$, the sequence $(|T^{\pi}(s)|)_{s \in S}$ is unique. Since $0 \leq |T^{\pi}(s)| \leq k-1$, the number of possible sequences is k^n. As the number of policies is also k^n, we have a bijection between Π and this set of "improvement sequences". This connection was already known for $k = 2$ [Mansour and Singh, 1999; Szabó and Welzl, 2001], which is simpler to analyse because $|T^{\pi}(s)| \in \{0, 1\}$ becomes an indicator for whether s is improvable. Our generalisation for all $k \geq 2$ is novel. Our use of $\gtrsim$, which break ties, gives Lemma 3 the convenient form of a bijection. Our analysis can also be undertaken using only $\succ$ and $\succeq$, albeit with extra cases to account for ties.

4.1 Improving over RPI's Upper Bound

In order to effectively use Lemma 3, we propose a slight modification to RPI. In the new variant, denoted RPI-UIP, π' is picked uniformly at random from $I(\pi)$, the set of improving policies. Even if $I(\pi)$ itself is exponentially large, note that it "factors" into improvements at each state, and so only a polynomial-time operation is needed to pick π' uniformly at random from $I(\pi)$.

RPI-UIP is identical to RPI on 2-action MDPs, but since RPI picks uniformly at random among the improvable *states* (and picks improving actions arbitrarily), the methods do not coincide for $k \geq 3$. Lemma 3 facilitates a tighter bound for RPI-UIP when used in conjunction with the analysis structure of Mansour and Singh [1999].

Definition 4. *A policy π is called a small-improvement policy if $|I(\pi)| \leq \alpha$ (we shall fix the parameter $\alpha > 0$ subsequently). A policy that is not a small-improvement policy is called a large-improvement policy.*

We present a novel bound on the number of small-improvement policies. This bound is slightly looser than the specialised one derived by Mansour and Singh [1999] for $k = 2$, but is tighter than theirs for $k \geq 3$.

Lemma 5. *For all $\alpha > 0$, there are at most $(\alpha+1)H_k^{n-1}$ small-improvement policies, where $H_k = \sum_{i=1}^{k} \frac{1}{i}$. Note that $H_k = \Theta(\log k)$.*

Proof. For convenience we take $S = \{1, 2, \ldots, n\}$. The bijection in Lemma 3 allows us to associate each policy π with a unique n-length, k-ary string x^{π} of the form $x_1^{\pi} x_2^{\pi} \ldots x_n^{\pi}$, wherein for s $\in S$, $x_s^{\pi} = |T^{\pi}(s)| + 1$. It is immediate that $|I(\pi)| = \prod_{i=1}^{n} x_i^{\pi} - 1$. Thus, π is a small-improvement policy if and only if $\prod_{i=1}^{n} x_i^{\pi} \leq \alpha + 1$. For $\beta > 0$, let $N(n, \beta)$ denote the number of n-length strings over {1, 2, …, k}—of the form $x = x_1 x_2 \ldots x_n$—for which $\prod_{i=1}^{n} x_i \leq \beta$. To prove the lemma, we induct on n to show that $N(n, \beta) \leq \beta H_k^{n-1}$.

Clearly for all $\beta > 0$, $N(1, \beta) = \min(\lfloor \beta \rfloor, k) \leq \beta$. Now assume that for all $\beta > 0$ and some $n \geq 1$, $N(n, \beta) \leq \beta H_k^{n-1}$. For $l \in \{1, 2, \ldots, k\}$, consider $N(n+1, \beta, x_1 = l)$, the number of $(n+1)$-length strings in which the first element is l and $\prod_{i=1}^{n+1} x_i \leq \beta$. We have: (1) $N(n+1, \beta) = \sum_{l=1}^{k} N(n+1, \beta, x_1 = l)$, and (2) $N(n+1, \beta, x_1 = l) = N(n, \frac{\beta}{l})$. Applying the induction hypothesis, we get $N(n+1, \beta) \leq \sum_{l=1}^{k} \frac{\beta}{l} H_k^{n-1} = \beta H_k^{n}$, which completes the proof. □

Next we lower-bound the progress made by each step of RPI-UIP. By constructing a total order, we obtain a simpler proof of the following lemma than the one given by Mansour and Singh [1999, see Lemma 9].

Lemma 6. *Let π' be the policy obtained by performing policy improvement to a policy π using RPI-UIP. Then, with probability at least $\frac{1}{2}$, there exist $\lfloor \frac{|I(\pi)|}{2} \rfloor$ policies $\pi'' \neq \pi$ such that $\pi'' \gtrsim \pi$ and $\neg(\pi'' \gtrsim \pi')$.*

Proof. We define a relation R between policies: $\pi_1 R \pi_2 \iff (\pi_1 \gtrsim \pi_2) \vee (\neg(\pi_1 \gtrsim \pi_2) \wedge (\pi_1 L \pi_2))$. Observe that R induces a total order on Π, and therefore on $I(\pi)$. Since π' is picked uniformly at random from $I(\pi)$, with probability at least $1/2$, we will have $\pi' R \pi''$—which implies $\neg(\pi'' \gtrsim \pi')$—for some $\lfloor |I(\pi)|/2 \rfloor$ policies π''. Since $\pi'' \in I(\pi)$, we have $\pi'' \neq \pi$, and by Theorem 1, $\pi'' \gtrsim \pi$. □

We are ready to upper-bound the complexity of RPI-UIP.

Theorem 7. *The expected number of policies evaluated by RPI-UIP is at most $O(k^{n/2} H_k^{(n-1)/2})$.*

Proof. Define $L^{\star} = k^n / \lfloor \alpha/2 \rfloor$. Since each large-improvement policy eliminates at least $\lfloor \alpha/2 \rfloor$ policies with probability at least $1/2$, and in total there are k^n policies, the expected number of large-improvement policies visited is at most $2L^{\star}$. By Lemma 5, the total number of small-improvement policies is at most $(\alpha + 1)H_k^{n-1}$. Taking $\alpha = \sqrt{k^n / H_k^{n-1}}$, we obtain a bound of $O(k^{n/2} H_k^{(n-1)/2})$ on the expected iterations of RPI-UIP. Note that the number of iterations decays exponentially: the probability that $6L^{\star}$ large-improvement policies are visited is upper-bounded by $\binom{6L^{\star}}{5L^{\star}} \left(\frac{1}{2}\right)^{5L^{\star}} = \binom{6L^{\star}}{L^{\star}} \left(\frac{1}{2}\right)^{5L^{\star}} \leq \left(\frac{6e}{32}\right)^{L^{\star}} < 0.51^{L^{\star}}$. □

Whereas RPI is shown to eliminate $(\Theta(1))^n$ policies in large-improvement steps [Mansour and Singh, 1999], we have shown that RPI-UIP eliminates $(\text{poly}(k))^n$ policies in such steps—which leads to the tighter upper bound.

4.2 A new upper bound for HPI

In HPI, every improvable state must necessarily be switched. Yet, for $k \geq 3$, there could sometimes be two or more improving actions for a particular state—so there is still a choice to resolve. We propose HPI-R, a variant of HPI, in which an improving action is picked uniformly at random from those available for every improvable state. Equivalently, let us say that a policy π' *strictly improves* upon a policy π if $\forall s \in S : (\pi'(s) \in (T^\pi(s) \cup \{\pi(s)\})) \wedge (\pi'(s) = \pi(s) \Rightarrow |T^\pi(s)| = 0)$. Let $I_{\text{strict}}(\pi)$ be the set of all policies that strictly improve upon π. HPI-R picks an improving policy π' uniformly at random from $I_{\text{strict}}(\pi)$. The crux of our analysis is that $I_{\text{strict}}(\pi)$ is still sufficiently large if $I(\pi)$ is large. If π is optimal, $I_{\text{strict}}(\pi) = I(\pi) = \emptyset$, else

$$|I_{\text{strict}}(\pi)| = \prod_{s \in S} \max(|T^\pi(s)|, 1) \geq \prod_{s \in S} \left(\frac{|T^\pi(s)| + 1}{2} \right)$$
$$\geq \frac{\prod_{s \in S}(|T^\pi(s)| + 1) - 1}{2^n} = \frac{|I(\pi)|}{2^n}.$$

With this connection established, we can use the same analysis structure as before. With probability at least $1/2$, HPI-R eliminates $\lfloor I(\pi)/2^{n+1} \rfloor$ policies after visiting π. Taking $\alpha = \sqrt{2^n k^n / H_k^{n-1}}$, we obtain the following upper bound.

Theorem 8. *The expected number of policies evaluated by HPI-R is at most $O(2^{n/2} k^{n/2} H_k^{(n-1)/2})$.*

For $k \geq 5$, this bound marks an exponential improvement over the previous bound of $O(k^n/n)$ for HPI [Mansour and Singh, 1999].

4.3 Tighter bounds for BSPI

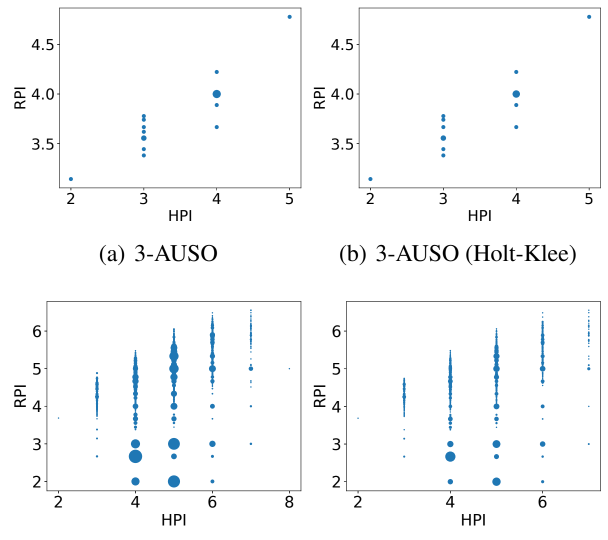
The tightest strong bound yet for deterministic PI variants is achieved by BSPI [Kalyanakrishnan *et al.*, 2016a]. Assume S is partitioned arbitrarily into $\lceil n/b \rceil$ b-sized batches (with one batch possibly smaller). If the batches are indexed, then at each iteration, BSPI selects the highest-indexed batch $B \subseteq S$ that has improvable states, and switches *all* the improvable states in B. This operation can be viewed as performing HPI within B. The analysis of BSPI rests on a recursive argument that if $\tau(b)$ is an upper bound on the iterations taken by HPI on a b-state MDP, then BSPI with a batch size of b on an n-state MDP will take at most $\tau(b)^{\lceil n/b \rceil}$ iterations. While the original analysis was for 2-action MDPs [Kalyanakrishnan *et al.*, 2016a], a carefully-designed recursion over actions is shown to yield an upper bound of $k^{\lceil n/b \rceil \log_2 \tau(b)}$ iterations for k-action MDPs [Gupta and Kalyanakrishnan, 2017]. Using the order regularity relaxation to bound $\tau(b)$, a computer search shows $\tau(7) \leq 33$ [Gerencsér *et al.*, 2015].

We obtain tighter bounds for BSPI by enumerating all possible AUSOs of dimension up to 4 (the number of AUSOs is doubly-exponential in the dimension, and currently infeasible to enumerate for dimension 5). We directly calculate the number of evaluations performed by HPI starting at each possible vertex on each AUSO; this number is guaranteed not to exceed the order regularity bound. Interestingly, we can also test which AUSOs satisfy the Holt-Klee conditions, possibly further tightening the upper bound applicable to MDPs.

Ignoring isomorphisms, there are 18 AUSOs in dimension 3 (or "3-AUSOs"), of which 16 satisfy the Holt-Klee conditions.[1] There are instances of both types of AUSOs on which HPI can perform 5 evaluations, matching the bound from the order regularity problem. A more interesting fact emerges from the set of 4-AUSOs. There are 12640 4-AUSOs, of which 6113 satisfy the Holt-Klee conditions. For the latter set, HPI never needs more than 7 evaluations. Hence, we obtain $7^{n/4} < 1.6266^n$ iterations as a bound for BSPI with $b = 4$ on 2-action MDPs, which improves upon the existing bound of $33^{n/7} < 1.6479^n$. The corresponding improvement for k-action MDPs is a bound of $k^{0.7019}$ iterations in place of $k^{0.7207n}$ [Gupta and Kalyanakrishnan, 2017]. HPI never performs more than 8 evaluations on 4-AUSOs; indeed there is only one 4-AUSO on which it performs 8 evaluations. This AUSO, shown in Appendix **??** (see supplementary material), does not satisfy the Holt-Klee conditions.

While the improved upper bounds above become the tightest strong worst-case bounds known for the PI family (since HPI is deterministic), our enumerated list of AUSOs also facilitates the analysis of a randomised variant of BSPI in which RPI is used within each batch. We denote this variant BSPI-R. We are not aware of relaxations such as the order regularity problem for analysing BSPI-R; our direct enumeration of AUSOs provides the first non-trivial bounds. We find that the expected number of iterations RPI takes on 3-AUSOs is at most 4.7778 and on 4-AUSOs is at most 6.5544 (in both cases the Holt-Klee conditions do not lead to tighter bounds). Although these bounds—maximised over all AUSOs in the corresponding family—are smaller than those for HPI, there do exist AUSOs on which HPI

[1]Of the 19 oriented 3-cubes listed by Stickney and Watson [1978, see Figure 3], the 19th contains a cycle.



(a) 3-AUSO (b) 3-AUSO (Holt-Klee)

(c) 4-AUSO (d) 4-AUSO (Holt-Klee)

Figure 2: Number of iterations taken by HPI and RPI on instances of different families of AUSOs. The area of each circle is proportional to the number of AUSOs whose HPI- and RPI-iterations are the centre's x and y coordinates, respectively.

dominates RPI. The distributions of the number of iterations taken by RPI and HPI on 3- and 4-AUSOs are shown in Figure 2. Reusing the recursion shown by Kalyanakrishnan *et al.* [2016a], we obtain $6.5544^{n/4} < 1.6001^n$ as a bound on the number of iterations taken by BSPI-R on 2-action MDPs. This strong upper bound is the tightest shown to date for the PI family for $k = 2$. For $k \geq 3$, the tightest such bounds are $(O(\log k))^n$ [Gupta and Kalyanakrishnan, 2017].

5 LOWER BOUND

Melekopoglou and Condon [1994] derived the first exponential lower bound for PI by analysing runs of Simple PI on a family of 2-action MDPs, parameterised by the number of states n. We use the same family of MDPs to show a linear lower bound on the expected number of policies evaluated by RPI [Mansour and Singh, 1999].

The MDP $M_n = (S, A, P, R, \gamma)$, for $n \geq 2$, is shown in Figure 3 and explained below. In M_n, there are 3 types of states in the set $S = \{1, 2, \ldots, n\} \cup \{0', 1', \ldots, n'\} \cup \{\tilde{0}, \tilde{1}\}$. The action set is $A = \{0, 1\}$. States labelled i have deterministic transitions: depending on the action, the next state is either $i - 1$ or i' for $i \geq 2$. From states labelled i', regardless of the action taken, the next state is picked uniformly at random from $(i-1)'$ and $(i-2)$ for $i \geq 3$. States $\tilde{0}$ and $\tilde{1}$ are sinks; the agent stays in them forever once they are reached. There is a reward of -1 on entering $\tilde{1}$ from $0'$ or $1'$. All other rewards are zero. The transitions that make up the corner cases are shown in Figure 3. As in the original construction [Melekopoglou and Condon, 1994], we use $\gamma = 1$. However, it can be verified that the switches made by PI—which depend solely on the Q function—remain the same for $\gamma < 1$.

Strictly speaking, M_n has $2n + 3$ states and hence 2^{2n+3} policies. However, there are only n states where the action non-trivially affects the outcome of the next transition. The remaining states are "dummy". Formally, for a policy π and state $s \in \{0', 1', \ldots, n'\} \cup \{\tilde{0}, \tilde{1}\}$ such that $\pi(s) = 0$, we get $Q^\pi(s, 0) = Q^\pi(s, 1) = V^\pi(s)$, and so $T^\pi(s) = \emptyset$. Hence s will not be switched. We adopt the convention that we start PI at a policy π^0 such that all "dummy" states s have $\pi^0(s) = 0$. The action at these states will remain 0 in every policy thereafter visited by PI. In our analysis, we only deal with states in $\{1, 2, \ldots, n\}$, and we treat policies as mappings from $\{1, 2, \ldots, n\}$ to $\{0, 1\}$. Consequently we may represent a policy π as a bit-string $w_1 w_2 \ldots w_n \in \{0, 1\}^n$ where $\pi(s) = w_s$. In this notation, it can be seen that $\pi^0 = 0^n$ and $\pi^\star = 10^{n-1}$. We show the following lower bound.

Theorem 9. *Starting from $\pi^0 = 0^n$, the expected number of policies RPI evaluates on M_n before terminating is at least $\frac{n+1}{2}$.*

The proof is based on structural properties of M_n identified by Melekopoglou and Condon [1994]. We provide the proof in Appendix B in the supplementary material.

6 EXPERIMENTS

Interestingly, although the paper by Mansour and Singh [1999] is nearly twenty years old, we are not aware of an experimental comparison between RPI and HPI published in the literature. We present an experimental comparison of these methods and the variants discussed in this paper, summarising our results in Figure 4.

Unlike our theoretical bounds, which are *maximised* over MDPs, each graph plots an *average* over 500 randomly generated MDPs. In the same manner as Kalyanakrishnan *et al.* [2016a], we generate an n-state, k-action MDP by uniformly sampling $n/5$ possible successors for each of the nk state-action pairs. For each (s, a, s') triple thus obtained, the reward $R(s, a, s')$ is drawn from a standard normal distribution; $P(s, a, s')$ is drawn uniformly from $[0, 1]$ and thereafter normalised. The remaining rewards and probabilities are set to 0. We take $n = 60$ and $\gamma = 0.99$. The starting policy for PI on each MDP is picked uniformly at random.

Figure 4(a) compares variants of HPI and RPI as k is varied. Our first observation is that running HPI with

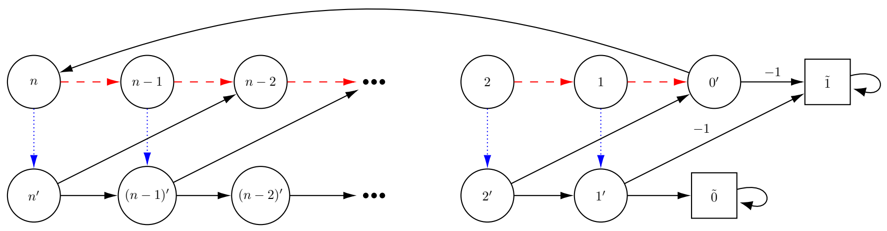


Figure 3: Family of MDPs used to prove the lower bound on RPI. Red (dashed) and blue (dotted) edges from states $1, 2, \ldots, n$ correspond to actions 0 and 1, respectively. Black (solid) edges from all others states are equiprobable under both actions.

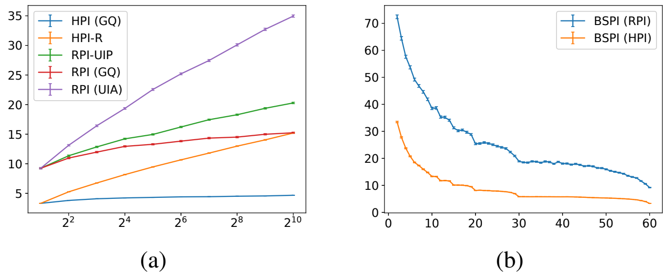


Figure 4: Expected iterations (y axis) against number of actions k in (a) and against batch size b in (b). Each point is an average from 500 independent runs; error bars show one standard error. PI variants HPI (GQ) and RPI (UIA) are described in Section 6.

greedy action selection based on the Q-function (GQ) is by far the most efficient variant. HPI-R comes second for small values of k. Among RPI variants, too, GQ switching performs the best. Note that GQ-switching is only applicable to MDPs, whereas the other variants apply more generally to AUSOs. RPI-UIP consistently takes fewer iterations than than RPI (UIA), a variant of RPI [Mansour and Singh, 1999] in which improving actions are picked uniformly at random from chosen improvable states, which are themselves first selected uniformly at random.

Figure 4(b) assesses the effect of using RPI within the batches of BSPI, comparing it with the canonical approach of using HPI [Kalyanakrishnan *et al.*, 2016a]. This experiment uses $k = 2$. For every fixed batch size b, we find that the HPI-based variant performs better in aggregate. Both variants show the same trend: a fairly consistent drop in the number of iterations as b is increased. If it can be proven that a larger batch size implies a tighter upper bound for RPI-based BSPI, it would follow that our bound of 1.6001^n, obtained by analysing 4-AUSOs, also applies to RPI itself (as it is equivalent to BSPI with a batch size of n). Such a result would improve the current bound of $O(1.7172^n)$ iterations for RPI substantially.

7 DISCUSSION

Our experimental results are perfectly consistent with our theoretical upper bounds: for 60-state MDPs, the upper bounds far exceed the ranges plotted in Figure 4. Yet, curiously, many *trends* seen in practice—even if only on one family of randomly-generated MDPs—are quite opposite to the *trends* among the theoretical bounds. By way of RPI-UIP, we have shown the tightest upper bounds yet for the PI family, which improve upon those for RPI and HPI. Yet, HPI seems to work much better in practice. We may attribute this disparity both to the looseness of the upper bounds, and to our choice of test MDPs. The lower bound we show for RPI matches the tightest for HPI on 2-action MDPs—but both bounds are only linear. It remains to be seen if there are MDPs on which RPI and HPI have substantially higher complexity.

The analysis provided in this paper does not exploit any properties specific to MDPs, but applies more generally to AUSOs. It would be interesting to analyse RPI specifically on MDPs. For example, it is known that the Simplex method runs in strongly polynomial time on the Linear Program arising from *deterministic* MDPs [Post and Ye, 2013]. To the best of our knowledge, HPI and RPI have not been shown to enjoy the same guarantee.

ACKNOWLEDGEMENTS

SK was partially supported by grants from SERB (ECR/2017/002479) and Ubisoft India.

References

[Bellman, 1957] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1^{st} edition, 1957.

[Bertsekas and Tsitsiklis, 1996] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[Bertsekas, 2012] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volume 2. Athena Scientific, 4^{th} edition, 2012.

[Fearnley, 2010] John Fearnley. Exponential lower bounds for policy iteration. In *Proceedings of the Thirty-seventh International Colloquium on Automata, Languages and Programming (ICALP 2010)*, pages 551–562. Springer, 2010.

[Feinberg and Schwartz, 2002] Eugene A. Feinberg and Adam Schwartz, editors. *Handbook of Markov Decision Processes: Methods and Applications*. Springer, 2002.

[Gerencsér *et al.*, 2015] Balázs Gerencsér, Romain Hollanders, Jean-Charles Delvenne, and Raphaël M. Jungers. A complexity analysis of policy iteration through combinatorial matrices arising from unique sink orientations, 2015. URL: http://arxiv.org/pdf/1407.4293v2.pdf.

[Gupta and Kalyanakrishnan, 2017] Anchit Gupta and Shivaram Kalyanakrishnan. Improved strong worst-case upper bounds for mdp planning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 1788–1794, 2017.

[Hansen and Zwick, 2010] Thomas Dueholm Hansen and Uri Zwick. Lower bounds for Howard's algorithm for finding minimum mean-cost cycles. In *Proceedings of the Twenty-second International Symposium on Algorithms and Computation (ISAAC 2011)*, pages 425–426. Springer, 2010.

[Hollanders *et al.*, 2012] Romain Hollanders, Balázs Gerencsér, and Jean-Charles Delvenne. The complexity of policy iteration is exponential for discounted Markov decision processes. In *Proceedings of the Fifty-first IEEE Conference on Decision and Control (CDC 2012)*, pages 5997–6002. IEEE, 2012.

[Hollanders *et al.*, 2014] Romain Hollanders, Balázs Gerencsér, Jean-Charles Delvenne, and Raphaël M. Jungers. Improved bound on the worst case complexity of policy iteration, 2014. URL: http://arxiv.org/pdf/1410.7583v1.pdf.

[Holt and Klee, 1999] Fred Holt and Victor Klee. A proof of the strict monotone 4-step conjecture. *Contemporary Mathematics*, 223:201–216, 1999.

[Howard, 1960] Ronald A. Howard. *Dynamic programming and Markov processes*. MIT Press, 1960.

[Kalyanakrishnan *et al.*, 2016a] Shivaram Kalyanakrishnan, Utkarsh Mall, and Ritish Goyal. Batch-switching policy iteration. In *Proceedings of the Twenty-fifth International Joint Conference on Artificial Intelligence (IJCAI 2016)*, pages 3147–3153. AAAI Press, 2016.

[Kalyanakrishnan *et al.*, 2016b] Shivaram Kalyanakrishnan, Neeldhara Misra, and Aditya Gopalan. Randomised procedures for initialising and switching actions in policy iteration. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI 2016)*, pages 3145–3151. AAAI Press, 2016.

[Littman *et al.*, 1995] Michael L. Littman, Thomas L. Dean, and Leslie Pack Kaelbling. On the complexity of solving Markov decision problems. In *Proceedings of the Eleventh Annual Conference on Uncertainty in Artificial Intelligence (UAI 1995)*, pages 394–402. Morgan Kaufmann, 1995.

[Mahadevan, 1996] Sridhar Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning*, 22(1–3):159–195, 1996.

[Mansour and Singh, 1999] Yishay Mansour and Satinder Singh. On the complexity of policy iteration. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI 1999)*, pages 401–408. Morgan Kaufmann, 1999.

[Matoušek *et al.*, 1996] Jiří Matoušek, Micha Sharir, and Emo Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4/5):498–516, 1996.

[Megiddo, 1982] Nimrod Megiddo. Is binary encoding appropriate for the problem-language relationship? *Theoretical Computer Science*, 19:337–341, 1982.

[Melekopoglou and Condon, 1994] Mary Melekopoglou and Anne Condon. On the complexity of the policy improvement algorithm for Markov decision processes. *INFORMS Journal on Computing*, 6(2):188–192, 1994.

[Post and Ye, 2013] Ian Post and Yinyu Ye. The simplex method is strongly polynomial for deterministic Markov decision processes. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2013)*, pages 1465–1473. Morgan Kaufmann, 2013.

[Puterman, 1994] Martin L. Puterman. *Markov Decision Processes*. Wiley, 1994.

[Scherrer, 2013] Bruno Scherrer. Improved and generalized upper bounds on the complexity of policy iteration. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, pages 386–394. Curran Associates, 2013.

[Schurr and Szabó, 2005] Ingo Schurr and Tibor Szabó. Jumping doesn't help in abstract cubes. In Michael Jünger and Volker Kaibel, editors, *Integer Programming and Combinatorial Optimization*, pages 225–235, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[Stickney and Watson, 1978] Alan Stickney and Layne Watson. Digraph models of Bard-type algorithms for the linear complementarity problem. *Mathematics of Operations Research*, 3(4):322–333, 1978.

[Sutton and Barto, 1998] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[Szabó and Welzl, 2001] Tibor Szabó and Emo Welzl. Unique sink orientations of cubes. In *Proceedings of the Forty-second Annual Symposium on Foundations of Computer Science (FOCS 2001)*, pages 547–555. IEEE, 2001.

[Szepesvári, 2010] Csaba Szepesvári. *Algorithms for Reinforcement Learning*. Morgan & Claypool, 2010.

[White, 1985] Douglas J. White. Real applications of Markov Decision Processes. *Interfaces*, 15(6):73–83, 1985.

[White, 1988] Douglas J. White. Further real applications of Markov Decision Processes. *Interfaces*, 18(5):55–61, 1988.

[Ye, 2011] Yinyu Ye. The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate. *Mathematics of Operations Research*, 36(4):593–603, 2011.