July 10, 1988
(revised May 27, 1989)

LIDS-P-1793

Solving H-Horizon, Stationary Markov Decision Problems in Time Proportional to log(H)*

by

Paul Tseng†

Abstract

We consider the H-horizon, stationary Markov decision problem. For the discounted case, we give an ϵ-approximation algorithm whose time is proportional to $\log(1/\epsilon)$, $\log(H)$ and $1/(1-\alpha)$, where α is the discount factor. Under an additional stability assumption, we give an exact algorithm whose time is proportional to $\log(H)$ and $1/(1-\alpha)$. For problems where α is bounded away from 1, we obtain, respectively, a fully polynomial approximation scheme and a polynomial-time algorithm. We derive analogous results for the undiscounted case under the assumption that all stationary policies are proper.

KEY WORDS: computational complexity, dynamic programming, Markov decision process.

---

* This research is partially supported by the U.S. Army Research Office, contract DAAL03-86-K-0171 (Center for Intelligent Control Systems), and by the National Science Foundation under grant NSF-ECS-8519058.

† The author is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139.

Acknowledgement: We gratefully acknowledge the many helpful comments made by an anonymous referee, particularly for suggesting a usable version of the bound t^*, and by Professor J.N. Tsitsiklis.

1. Introduction

Complexity analysis [5], [13] has been widely applied in the areas of theoretical computer science and combinatorial/integer optimization to measure the inherent difficulty of problems. In dynamic programming, such analysis has been less common [11], [12], [14]. In this article, we make some progress towards filling this gap. In particular, we consider the H-horizon, stationary Markov decision problem [1], [4], [7], which is not known to be polynomial-time solvable, and show that an ε-optimal solution is computable in time that is proportional to $\log(1/\varepsilon)$ and $\log(H)$. Under an additional stability assumption, we show that an exact solution is computable in time that is proportional to $\log(H)$. For the special case of discounted problems where the discount factor is bounded away from 1, we obtain, respectively, a fully polynomial approximation scheme and a polynomial-time algorithm. Our result in a sense brings us closer to a complete complexity theory for Markov decision problems, for which it is known that the infinite horizon, stationary case is P-complete, and the finite horizon, nonstationary case is in NC [14] (complexity for the infinite horizon, nonstationary case is undefined). [See Appendix A for a brief explanation of the complexity terms used throughout this article.]

We describe the stationary Markov decision problem below. We are given a time horizon $H > 0$ (possibly $H = +\infty$), a finite set of states $S = \{1,\ldots,n\}$ and, for each state i, a finite set $D_i = \{1,\ldots,m_i\}$ of controls. At each time t ($t = 0, 1,\ldots, H-1$), we are in exactly one of the n states (the state at time 0 is given) and, if we are in state i, we choose a control from D_i. If we choose control $k \in D_i$, we incur a cost g_i^k and, with probability p_{ij}^k, we arrive in state j at time t+1. If we terminate in state i at time H, we incur a cost c_i. Let $u_i(t)$ denote the control chosen when we are in state i at time t and let $\mu(t) = (u_1(t),\ldots,u_n(t))$. Then the Markov decision problem is to choose a <u>policy</u> $(\mu(0),\mu(1),\ldots,\mu(H-1))$ to minimize the expected total cost

$$\sum_{t=0}^{H-1} \alpha^t \sum_{i \in S} g_i^{u_i(t)}\, p(i,t;\mu(0),\ldots,\mu(H-1)) \;+\; \alpha^H \sum_{i \in S} c_i\, p(i,H;\mu(0),\ldots,\mu(H-1)),$$

where $\alpha \in (0,1]$ is the <u>discount factor</u> and $p(i,t;\mu(0),\ldots,\mu(H-1))$ denotes the probability of being in state i at time t under the policy $(\mu(0),\mu(1),\ldots,\mu(H-1))$. Such a policy will be called an <u>optimal policy</u>. Also a policy $(\mu(0),\mu(1),\ldots,\mu(H-1))$ satisfying $\mu(0) = \mu(1) = \ldots = \mu(H-1)$ will be called <u>stationary</u> and will be written as $\mu(0)$. The Markov decision problem is <u>discounted</u> (<u>undiscounted</u>) if $\alpha < 1$ ($\alpha = 1$) and has finite (infinite) horizon if $H < +\infty$ ($H = +\infty$). In what follows, $\|\cdot\|$ will denote the L_∞-norm and $\log(\cdot)$ will denote the logarithm in base 2. For any $\mu = (u_1,\ldots,u_n) \in D_1 \times \ldots \times D_n$, we will denote $P^\mu = [p_{ij}^{u_i}]$ and $g^\mu = (g_1^{u_1},\ldots,g_n^{u_n})$. We will also denote $c = (c_1,\ldots,c_n)$.

We make the following standing assumption:

Assumption A: α and the p_{ij}^k's are rational numbers. The g_i^k's and c_i's are integers.

Let δ be the smallest positive integer for which $\delta\alpha$ and the δp_{ij}^k's are all integers and $|g_i^k| \le \delta$, $|c_i| \le \delta$ for all i and k. [δ represents the accuracy in the problem data.] We will denote

$$L = n\log(\delta)(\sum_i m_i + 1),$$

which represents the input size for the ∞-horizon problem. [Note that log(H) is part of the input only if $H < +\infty$.] For notational simplicity, we will also denote $L' = L + n\log(n)(\sum_i m_i)$.

To motivate our results, consider the case $H < +\infty$. This problem is known to be P-hard (i.e. as hard as any problem that is solvable in polynomial time) [14], but it is not even known to be in NP (although it can be seen to be in the larger class PSPACE). If we use dynamic programming to solve this problem, the complexity is $O(n(\sum_i m_i)H)$ arithmetic operations. If we use linear programming, then since the the linear program formulation can be seen to contain $H\sum_i m_i$ constraints with an input size of HL, the (theoretically) fastest linear programming algorithm [8], [10] would take $O(H^4(\sum_i m_i)^3L)$ arithmetic operations! Hence, as a first step towards polynomial-time complexity, we would at least like to find algorithms whose time is a polynomial in log(H). We propose a number of such algorithms, both exact and inexact. These algorithms can be viewed as truncated dynamic programming methods whereby truncation occurs the moment that an optimal stationary policy for the ∞-horizon problem is identified. For the discounted case, we obtain an ε-approximation algorithm that has a complexity of $O((n\sum_i m_i \log(1/\varepsilon) + nL')/(1-\alpha) + n^3\log H)$ arithmetic operations and, under an additional stability assumption, an exact algorithm that has a complexity of $O(nL'/(1-\alpha) + n^3\log H)$ arithmetic operations. Analogous algorithms are obtained for the undiscounted case under the assumption that all stationary policies are proper.

This article proceeds as follows: in §2 we show that, for the ∞-horizon, discounted problem, an optimal stationary policy can be identified by the successive approximation method in time that is a polynomial in L and $(1-\alpha)^{-1}$; in §3 we use the preceding fact to derive exact and approximation algorithms for the finite horizon, discounted problem; in §4 and §5 we perform an analogous analysis for the undiscounted problem; in §6 we present our conclusion and discuss extensions.

2. Infinite Horizon, Discounted Case

In this case $H = +\infty$ and $\alpha \in (0,1)$. Let $T:\Re^n \to \Re^n$ be the function whose i-th component is given by

$$T_i(x) = \min_{k \in D_i} T_i^k(x), \qquad \forall\, x \in \Re^n, \tag{1}$$

where, for each $k \in D_i$, we let

$$T_i^k(x) = \alpha \sum_j p_{ij}^k x_j + g_i^k. \tag{2}$$

Also, for each $\mu = (u_1,\ldots,u_n) \in D_1 \times \ldots \times D_n$, let $T^\mu : \Re^n \to \Re^n$ denote the function whose i-th component is $T_i^{u_i}$. It is easily shown using (1)-(2) that T is a contraction mapping of modulus α with respect to the L_∞-norm. Hence T has a unique fixed point, which we denote by $x^* = (x_1^*, \ldots, x_n^*)$ (i.e. $x^* = T(x^*)$). Furthermore, there exists at least one optimal policy that is stationary and each stationary policy μ is optimal if and only if

$$x^* = T^\mu(x^*)$$

(see [1, §5.3]).

Consider the Jacobi successive approximation iterations [1, §5.2] for solving this discounted problem:

$$x(t+1) = T(x(t)), \qquad t = 0, 1, \ldots, \tag{3a}$$

$$x(0) = c. \tag{3b}$$

Since T is a contraction mapping of modulus α with respect to the L_∞-norm, we have from (3a) that

$$\|x(t+1) - x^*\| \le \alpha \|x(t) - x^*\|, \quad t = 0, 1, \ldots; \tag{4}$$

hence the iterates x(t) converge to x^* at a geometric rate. Furthermore, it is known that an optimal stationary policy is identified after a finite number of iterations [1, pp. 236], [4]. Below we refine this result by giving an explicit bound on the number of iterations:

Lemma 1 Let t^* be the smallest positive integer such that, for all $t \ge t^*$, $x(t+1) = T^\mu(x(t))$ implies $x^* = T^\mu(x^*)$. Then $t^* \le \hat{t}$, where

$$\hat{t} = \lceil \log(2\delta^{2n+2} n^n (\|c\| + \max_{i,k} |g_i^k|/(1-\alpha)))/\log(1/\alpha) \rceil.$$

Proof: By (3b) and (4), after $t = \lceil \log(\varepsilon/\|c - x^*\|)/\log(\alpha) \rceil$ iterations, the error $\|x(t) - x^*\|$ is less than ε for any $\varepsilon > 0$. We show below that, for $\varepsilon \le 1/(2\,\delta^{2n+2}\, n^n)$, the corresponding policy is optimal. This would then imply that an optimal stationary policy can be identified after $\lceil \log(2\delta^{2n+2} n^n \|c - x^*\|) / \log(1/\alpha) \rceil$ iterations. To obtain a usable bound, notice (cf. (1)-(2)) that x^* satisfies

$$(I - \alpha P^\mu) x^* = g^\mu, \tag{5}$$

for some $\mu \in D_1 \times \ldots \times D_n$. Hence

$$\begin{aligned} \|x^*\| &= \|(I + (\alpha P^\mu) + (\alpha P^\mu)^2 + \ldots)\, g^\mu\| \\ &\le \|g^\mu\| + \|(\alpha P^\mu) g^\mu\| + \|(\alpha P^\mu)^2 g^\mu\| + \ldots \\ &\le \|g^\mu\|/(1-\alpha) \\ &\le \max_{i,k} |g_i^k|/(1-\alpha), \end{aligned}$$

so that $\|c-x^*\|$ is upper bounded by the computable quantity $\|c\| + \max_{i,k} |g_i^k|/(1-\alpha)$. By plugging the latter quantity into the above bound, we obtain the desired $\hat{t}$.

It only remains to show that if $\|x(t)-x^*\| < 1/(2\,\delta^{2n+2}\,n^n)$, then the corresponding policy is optimal. Since $\delta^2(I-\alpha P^\mu)$ and $\delta^2 g^\mu$ are both integers (and the entries of $\delta^2(I-\alpha P^\mu)$ do not exceed δ^2), it follows from (5), Cramer's rule, and the Hadamard determinant inequality [6] that $x^* = w/(\delta^{2n}\,n^n)$, for some integer vector $w = (w_1, \ldots, w_n)$. Consider any $i \in S$ and any $k \in D_i$ such that $x_i^* \neq T_i^k(x^*)$. Since (cf. (2))

$$\begin{aligned} T_i^k(x^*) &= \alpha \textstyle\sum_j p_{ij}^k\, x_j^* + g_i^k \\ &= (\delta^2\, \alpha \textstyle\sum_j p_{ij}^k\, w_j + (\delta^{2n+2}\, n^n)\, g_i^k)/(\delta^{2n+2}\, n^n) \end{aligned}$$

and the numerator is an integer, it must be that x_i^* and $T_i^k(x^*)$ differ by at least $1/(\delta^{2n+2}\,n^n)$. Hence if $\|x(t)-x^*\| < 1/(2\,\delta^{2n+2}\,n^n)$, then

$$\begin{aligned} |T_i^k(x(t))-x_i^*| &= |\alpha \textstyle\sum_j p_{ij}^k\, (x_j(t)-x_j^*) + T_i^k(x^*)-x_i^*| \\ &\geq |T_i^k(x^*)-x_i^*| - |\alpha \textstyle\sum_j p_{ij}^k\, (x_j(t)-x_j^*)| \\ &\geq 1/(\delta^{2n+2}\, n^n) - \alpha\|x(t)-x^*\| \\ &> 1/(2\delta^{2n+2}\, n^n) \\ &> \|x(t)-x^*\|. \end{aligned}$$

Since (cf. (3a), (4)) $\|T(x(t))-x^*\| \leq \|x(t)-x^*\|$, this implies that $T_i^k(x(t)) \neq T_i(x(t))$. Q.E.D.

[A slightly different value for $\hat{t}$ is obtained if we use the alternative bound $\|c-T(c)\|/(1-\alpha)$ on $\|c-x^*\|$.] Since $\log(\cdot)$ is a concave function and its slope at 1 is 1, we have

$$\begin{aligned} \log(\alpha) &= \log(1-(1-\alpha)) \\ &\leq -(1-\alpha). \end{aligned}$$

This, together with the facts (cf. Assumption A) $\|c\| \leq \delta$, $\max_{i,k} |g_i^k|/(1-\alpha) \leq \delta^2$, implies that

$$\hat{t} = O(n\log(n\delta)/(1-\alpha)), \tag{6}$$

which is a polynomial in L and $1/(1-\alpha)$. This apriori estimate of t^*, although, as we shall see, is sufficiently small for deriving our complexity results, is nonetheless too large to be practical. A more accurate estimate of t^* is obtained by using a more accurate upper bound on $\|x^*-x(t)\|$. [Generation of such a bound, although not useful for complexity analysis, is discussed in [1, pp. 190].] Then we can estimate t^* by t whenever this bound is less than $1/(2\,\delta^{2n+2}\,n^n)$. Alternatively, we can estimate t^* by using bounds on x^* to eliminate inactive controls. This approach is based on the following lemma:

Lemma 2 Fix any positive integer $\bar{t}$. Let Δ be any scalar satisfying $\|x^*-x(\bar{t})\| \leq \Delta$. Then, for any $i \in S$ and $k \in D_i$, if

$$T_i^k(x(\bar{t})) - x_i(\bar{t}+1) > 2\Delta(2\alpha + g_i^k + g_i^{\bar{k}}),$$

where $\bar{k}$ is any element of D_i satisfying $T_i^{\bar{k}}(x(\bar{t})) = x_i(\bar{t}+1)$, then $T_i(x(t)) \neq T_i^k(x(t))$ for all $t \geq \bar{t}$.

Proof: Suppose that for some $t \geq \bar{t}$ we have $T_i(x(t)) = T_i^k(x(t))$. By (4) we have $\|x^*-x(t)\| \leq \Delta$ and hence $\|x(t)-x(\bar{t})\| \leq 2\Delta$. This, together with (1), the monotonicity of $T_i^{\bar{k}}$ and the choice of $\bar{k}$, implies

$$\begin{aligned} T_i^k(x(t)) &\leq T_i^{\bar{k}}(x(t)) \\ &\leq T_i^{\bar{k}}(x(\bar{t}) + 2\Delta e) \\ &= x_i(\bar{t}+1) + 2T_i^{\bar{k}}(e), \end{aligned}$$

where e denotes the n-vector all of whose components are 1's. Similarly, we have

$$\begin{aligned} T_i^k(x(t)) &\geq T_i^k(x(\bar{t}) - 2\Delta e) \\ &= T_i^k(x(\bar{t})) - 2\Delta T_i^k(e). \end{aligned}$$

Combine the above two inequalities and we obtain (cf. (2)) $T_i^k(x(\bar{t})) - x_i(\bar{t}+1) \leq 2\Delta(T_i^{\bar{k}}(e) + T_i^k(e)) = 2\Delta(2\alpha + g_i^k + g_i^{\bar{k}})$. Q.E.D.

Lemma 2 provides a test for eliminating controls that are inactive in all future iterations. [Similar tests for eliminating non-optimal controls are given in [1, pp. 198], [16].] When only those stationary policies that are optimal for the ∞-horizon problem (which can be determined apriori) are left, then the current iteration count is an estimate of t^*. We have emphasized the accurate estimation of t^* because, as we shall see in §3, t^* plays a key role in our solution of finite-horizon, discounted problems; the more accurately we estimate t^*, the better our solution times will be.

3. Finite Horizon, Discounted Case

In this case, $H < +\infty$ and $\alpha \in (0,1)$. Consider the following dynamic programming iterations:

$$x(t) = T(x(t+1)), \quad t = H-1, \ldots, 1, 0, \tag{7a}$$

$$x(H) = c, \tag{7b}$$

where T is given by (1)-(2) and x(t) denotes the cost-to-go vector at time t. A policy $(\mu(0),\mu(1), \ldots,\mu(H-1))$ can be seen to be optimal for the H-horizon discounted problem if and only if $x(t) = T^{\mu(t)}(x(t+1))$ for all $t = H-1, \ldots, 1, 0$. The problem is then to compute x(0), which is the optimal expected cost (and perhaps to determine the optimal policy as well).

Since the iteration (7a)-(7b) is identical to (3a)-(3b), except for the reversal in time, Lemma 1 motivates an algorithm for computing x(0) whereby T in the iteration (7a) is switched to T^{μ}, with μ being some optimal stationary policy for the ∞-horizon problem, the moment that such a policy is identified. We state this algorithm below:

Truncated DP (Dynamic Programming) Algorithm

Phase 0 Choose a positive integer $\tilde{t}$. Let $\tilde{x}(H) = c$.

Phase 1 Run the recursion $\tilde{x}(t) = T(\tilde{x}(t+1))$ until $t = H-\tilde{t}-1$. [If $H \le \tilde{t}$, then quit when t reaches 0.]

Phase 2 Let $\tilde{\mu}$ be any stationary policy satisfying $\tilde{x}(H-\tilde{t}-1) = T^{\tilde{\mu}}(\tilde{x}(H-\tilde{t}))$. Then compute

$$\tilde{x}(0) = (\alpha P^{\tilde{\mu}})^{H-\tilde{t}}\, \tilde{x}(H-\tilde{t}) + [I + (\alpha P^{\tilde{\mu}}) + \ldots + (\alpha P^{\tilde{\mu}})^{H-\tilde{t}-1}]\, g^{\tilde{\mu}}.$$

We have the following complexity and accuracy results:

Proposition 1 The following hold for the truncated DP algorithm:

(a) It has a complexity of $O(n(\sum_i m_i)\tilde{t} + n^3 \log(H-\tilde{t}))$ arithmatic operations.

(b) For $\tilde{t} = \hat{t}$, we have $\|\tilde{x}(0)-x(0)\| \le 4\alpha^H \delta^2$.

(c) If the ∞-horizon problem has a unique optimal stationary policy, then for $\tilde{t} = \hat{t}$, we have $\tilde{x}(0) = x(0)$.

Proof: We first prove part (a). It is easily seen that Phases 0 and 1 require $O(n(\sum_i m_i)\tilde{t})$ arithmetic operations. Since A^k and $I + A + \ldots + A^k$ can be computed using binary powering and factoring (see Appendix B) in $O(n^3 \log(k))$ arithmetic operations for any $n \times n$ matrix A and $k \ge 1$, we can perform Phase 2 in $O(n^3 \log(H-\tilde{t}))$ arithmatic operations.

We next prove part (b). From (7a)-(7b) and the fact that T is an L_∞-norm contraction mapping of modulus α we have that

$$\|x(0)-x^*\| \le \alpha^H \|c-x^*\|, \qquad \|\tilde{x}(H-\tilde{t})-x^*\| \le \alpha^{\tilde{t}} \|c-x^*\|. \tag{8}$$

By Lemma 1, $\tilde{\mu}$ is an optimal stationary policy for the ∞-horizon problem, so that $x^* = T^{\tilde{\mu}}(x^*)$. This, together with the observation that $\tilde{x}(0)$ equals $H-\tilde{t}$ successive applications of $T^{\tilde{\mu}}$ to $\tilde{x}(H-\tilde{t})$, implies $\|\tilde{x}(0)-x^*\| \le \alpha^{H-\tilde{t}} \|\tilde{x}(H-\tilde{t})-x^*\|$. By combining this with (8), we obtain

$$\|\tilde{x}(0)-x(0)\| \le \|\tilde{x}(0)-x^*\| + \|x^*-x(0)\| \le 2\alpha^H \|c-x^*\|.$$

Since $\|c-x^*\| \le \|c\| + \|x^*\| \le 2\delta^2$, this proves (b).

To prove part (c), note that, by Lemma 1, every stationary policy μ satisfying $x(H-\tilde{t}-1) = T^{\mu}(x(H-\tilde{t}))$ is optimal for the ∞-horizon problem. Since the ∞-horizon problem, by assumption, has a unique optimal stationary policy, it follows that $x(t) = T^{\tilde{\mu}}(x(t+1))$ for all $t = H-\tilde{t}-1,\ldots, 1, 0$. This, together with the observation that $\tilde{x}(H-\tilde{t}) = x(H-\tilde{t})$ and that $\tilde{x}(0)$ equals $H-\tilde{t}$ successive applications of $T^{\tilde{\mu}}$ to $\tilde{x}(H-\tilde{t})$, implies that $\tilde{x}(0) = x(0)$. Q.E.D.

Parts (a) and (c) of Proposition 1 says that, if the ∞-horizon problem has a unique optimal stationary policy, then $x(0)$ (and the corresponding optimal policy) can be computed exactly in (cf. (6))

$$O(n(\textstyle\sum_i m_i)\hat{t} + n^3\log(H-\hat{t}))$$
$$\leq O(nL'/(1-\alpha) + n^3\log(H)) \qquad (9)$$

arithmetic operations. If α is bounded away from zero, then this time is a polynomial in the input size. To verify (in polynomial time) the uniqueness assumption, we can first compute x^*, the fixed point of T. [This can be done either by using recursion (7a)-(7b) to identify an optimal stationary policy μ in time $\hat{t}$ (cf. Lemma 1) and then solving (5), or by solving the linear programming formulation of the ∞-horizon problem [1, pp. 206].] Then we check to see if, for some $i \in S$, there exist two distinct k and k′ in D_i satisfying $x_i^* = T_i^k(x^*) = T_i^{k'}(x^*)$. This requires an additional $O(n^2 \sum_i m_i)$ arithmetic operations.

If the ∞-horizon problem does not have a unique optimal stationary policy, then the optimal policy may oscillate with time (see Appendix C for an example; also see [4, pp. 30]). In this case, it is not even known if the optimal policy has a polynomial-sized description. Nonetheless, we have the following ε-approximation algorithm for solving this problem:

ε-Approximation Algorithm ($\varepsilon > 0$)
If $H \leq (\log(1/\varepsilon) + 2\log\delta + 2)/\log(1/\alpha)$, then run the truncated DP algorithm with $\tilde{t} = H$;
otherwise run the truncated DP algorithm with $\tilde{t} = \hat{t}$.

The complexity of this ε-approximation algorithm is given below:

Proposition 2 For any $\varepsilon > 0$, the ε-approximation algorithm computes an $\tilde{x}(0)$ satisfying $\|\tilde{x}(0) - x(0)\| \leq \varepsilon$ in $O((n\sum_i m_i \log(1/\varepsilon) + nL')/(1-\alpha) + n^3\log(H))$ arithmatic operations.

Proof: If $H \leq (\log(1/\varepsilon) + 2\log\delta + 2)/\log(1/\alpha)$, then clearly the algorithm computes $x(0)$ and the time complexity is $O(n(\sum_i m_i)H) \leq O(n(\sum_i m_i)(\log(1/\varepsilon) + \log\delta)/\log(1/\alpha))$. Otherwise we have from part (b) of Proposition 1 that the algorithm computes an $\tilde{x}(0)$ satisfying

$$\|\tilde{x}(0)-x(0)\| \leq 4\alpha^H\delta^2 \leq \varepsilon,$$

where the second inequality follows from the hypothesis on H. The time complexity follows from part (a) of Proposition 1 and Eq. (9). Q.E.D.

Notice that the ε-approximation algorithm is not suited for problems where n is large since it must compute powers of $P^{\tilde{\mu}}$, which are typically not sparse. Also notice that if α is bounded away from zero, then the ε-approximation algorithm is, in the terminology of [5], a fully polynomial approximation scheme. In this special case, the H-horizon, discounted Markov decision problem remains P-hard [14], but is not known to be in NP.

4. Infinite Horizon, Undiscounted Case

In this case $H = +\infty$ and $\alpha = 1$. This problem is of interest primarily when there is a cost-free state, say state 1, which is absorbing. The objective then is to reach this state at minimum expected cost (see [2, §4.3.2]). More precisely, we say that a stationary policy μ is proper if every entry in the first column of $(P^\mu)^t \to 1$ as $t \to +\infty$. We make the following assumption in addition to Assumption A:

Assumption B: $p_{11}{}^k = 1$ and $g_1{}^k = 0$ for all $k \in D_1$. Furthermore, all stationary policies are proper.

Assumption B essentially requires that all states other than state 1 be transient and that state 1 incurs zero cost. Under Assumption B, it can be shown (see Proposition 3.3 in [2, §4.3.2]) that an optimal stationary policy for this problem exists. Moreover, a stationary policy μ is optimal if and only if

$$x^* = T^\mu(x^*).$$

where T is given by (1)-(2) with $\alpha = 1$ and x^* is the unique fixed point of T restricted to the subspace $X = \{ x \in \Re^n \mid x_1 = 0 \}$.

An important fact is that T restricted to X is a contraction with respect to some weighted L_∞-norm (see Ex. 3.3 in [2, §4.3]), which then allows us to apply an argument similar to that used in §2. We give a short proof of this fact below. Under Assumption B, {2,...,n} can be partitioned into nonempty subsets $S_1, \ldots, S_r$ such that for any $s \in \{1,\ldots,r\}$, $i \in S_s$, and $k \in D_i$, there exists some $j' \in \{1\} \cup S_1 \cup \ldots \cup S_{s-1}$ such that $p_{ij'}{}^k > 0$ (j' depends on both i and k). Define weights $\omega_2, \ldots, \omega_n$ as follows

$$\omega_i = 1 - \eta^{2s}, \qquad \forall\, i \in S_s,\ \forall\, s = 1,\ldots,r, \tag{10}$$

where $\eta = \min\{ p_{ij}^k \mid i, j, k \text{ such that } p_{ij}^k > 0 \}$. We have the following lemma:

Lemma 3 $\sum_{j\neq 1} p_{ij}^k \omega_j/\omega_i \leq \gamma$, for all $k\in D_i$ and all $i\neq 1$, where
$\gamma = (1-\eta^{2r-1})/(1-\eta^{2r})$. (11)

Proof: Since $\eta\in(0,1)$, we have from (10) that
$0 < \omega_i < 1, \quad \forall\, i\in S.$ (12)
Fix any $s \geq 1$, $i\in S_s$, and $k\in D_i$. Let j' be an element of $\{1\}\cup S_1\cup\ldots\cup S_{s-1}$ such that $p_{ij'}^k > 0$. We have from (10)-(12) that

$$\begin{aligned}(\textstyle\sum_{j\in S} p_{ij}^k \omega_j)/\omega_i &\leq (\textstyle\sum_{j\in S\setminus\{j'\}} p_{ij}^k + p_{ij'}^k \omega_{j'})/\omega_i \\ &= (1 + p_{ij'}^k(\omega_{j'}-1))/\omega_i \\ &\leq (1+\eta(\omega_{j'}-1))/\omega_i \\ &\leq (1-\eta^{2s-1})/\omega_i \\ &= (1-\eta^{2s-1})/(1-\eta^{2s}) \\ &\leq \gamma,\end{aligned}$$

where the second inequality follows from the fact $p_{ij'}^k \geq \eta$ and the third inequality follows from the fact (cf. (10)) $\omega_{j'} \leq 1-\eta^{2s-2}$. Q.E.D.

[Lemma 3 is a refinement of Ex. 3.3 in [2, §4.3] in that it gives an explicit expression for ω_i and γ.] Lemma 3 implies that (cf. (1)-(2)), for any $i \neq 1$, any $x = (x_1,\ldots,x_n)\in X$ and $y = (y_1,\ldots,y_n)\in X$,

$$\begin{aligned}T_i(x)-T_i(y) &\leq \textstyle\sum_j p_{ij}^k(x_j-y_j) \\ &= \textstyle\sum_{j\neq 1} (p_{ij}^k \omega_j)(x_j-y_j)/\omega_j \\ &\leq \gamma\,\omega_i \max_j \{(x_j-y_j)/\omega_j\},\end{aligned}$$

where k is an element of D_i for which $T_i(y) = T_i^k(y)$. Similarly we have

$$T_i(y)-T_i(x) \leq \gamma\,\omega_i \max_j \{(y_j-x_j)/\omega_j\}.$$

It then follows that

$$\|T(x)-T(y)\|^\omega \leq \gamma\,\|x-y\|^\omega, \quad \forall\, x\in X, \forall\, y\in X, \tag{13}$$

where $\|\cdot\|^\omega$ denotes the L_∞-norm scaled by $(1,\omega_2,\ldots,\omega_n)$, i.e. $\|x\|^\omega = \|(x_1,x_2/\omega_2,\ldots,x_n/\omega_n)\|$.

Since $\eta = z/\delta$ for some integer z and $r \leq n-1$, it can be seen from (10)-(11) that each $\log(\omega_i)$ and $\log(\gamma)$ is a polynomial in L and that $1-\gamma \geq \eta^{2r}$. Also, since $g_1^k = 0$ for all $k\in D_1$, $T(X) \subseteq X$. Then by a contraction argument analogous to that used in the proof of Lemma 1, we obtain that an optimal stationary policy can be identified after a number of successive approximation iterations (i.e. Eq. (3a)) that is bounded by a polynomial in η^{-2r} and L. [Alternatively, it can be seen that T restricted to X is an r-stage contraction with respect to the ordinary L_∞-norm, and that the modulus of contraction is

estimated by $1 - \min_{i\neq 1,\mu_2,\ldots,\mu_r} [P^{\mu_2}\cdots P^{\mu_r}]_{i1}$. This estimate is difficult to compute in general, but itself can be seen to be upper bounded by $1-\eta^r$.] Notice that we do not need to know the S_s's in order to compute this bound; it suffices to know η and an upper bound on r. On the other hand, if a tight upper bound on r is not available, then we can compute the S_s's by using, say, a labeling algorithm similar to Dijkstra's shortest path algorithm, whereby at the s-th iteration all $i\in S_s$ are labeled. The complexity of this algorithm can be shown to be $O(n^2\sum_i m_i)$. We can also minimize the function $\max_{i\neq 1,k\in D_i}\{\sum_{j\neq 1} p_{ij}^k\, \omega_j/\omega_i\}$ over all $(\omega_2,\ldots,\omega_n)\in (0,+\infty)^{n-1}$ to obtain the sharpest estimate of the modulus of contraction; but this minimization seems to be difficult in general.

5. Finite Horizon, Undiscounted Case

In this case $H < +\infty$ and $\alpha = 1$. By making (in addition to Assumption A) the Assumption B and combining the arguments in §4 with arguments analogous to those made in §2 and §3, we can find an ε-approximate solution in time that is a polynomial in $\log(1/\varepsilon)$, η^{-2r}, L and log(H), where η and r are defined as in Lemma 3. If the ∞-horizon problem has a unique optimal stationary policy, then we can find an exact solution in time that is a polynomial in η^{-2r}, L and log(H). These times are unfortunately very slow even for moderately large values of r and $1/\eta$.

6. Conclusion and Extensions

In this article we have shown that an ε-approximate solution of the H-horizon Markov decision problem with $H < +\infty$ is computable in time proportional to $\log(1/\varepsilon)$ and log(H) and, under an additional stability assumption, an exact solution is computable in time proportional to log(H). For the discounted case where the discount factor is bounded away from 1, we obtain, respectively, a fully polynomial approximation scheme and a polynomial-time algorithm. However, in view of the stability assumptions needed to obtain an exact solution and the absence of negative results, we are still far from a complete complexity theory for this problem. If the stability assumptions are removed, the example in Appendix C shows that we must consider policies that have a certain periodic property. Optimal policies having such a periodic property remain poorly understood.

References

[1] Bertsekas, D. P., *Dynamic Programming: Deterministic and Stochastic Models*, Prentice-Hall, Englewood Cliffs, NJ (1987).

[2] Bertsekas, D. P. and Tsitsiklis, J. N., *Parallel and Distributed Computation: Numerical Methods*, Prentice-Hall, Englewood Cliffs, NJ (1989).

[3] Cook, S. A., "Towards a complexity theory of synchronous parallel computation," *Enseign. Math. 2*, 27 (1981), 99-124.

[4] Federgruen, A. and Schweitzer, P. J., "Discounted and undiscounted value-iteration in Markov decision problems: a survey," in (Puterman, M. L. ed.) *Dynamic Programming and Its Applications*, Academic Press, New York, NY (1978), 23-52.

[5] Garey, M. R. and Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, CA (1979).

[6] Householder, A. S., *The Theory of Matrices in Numerical Analysis*, Dover Publications, New York, NY (1964).

[7] Howard, R. A., *Dynamic Programming and Markov Processes*, MIT Press, Cambridge, MA (1960).

[8] Karmarkar, N., "A new polynomial-time algorithm for linear programming," *Combinatorica*, 4 (1984), 373-395.

[9] Lewis, H. R. and Papadimitriou, C. H., *Elements of the Theory of Computation*, Prentice-Hall, Englewood Cliffs, NJ (1981).

[10] Megiddo, N. (ed.), *Progress in Mathematical Programming: Interior-Point and Related Methods*, Springer-Verlag, New York, NY (1989).

[11] Orlin, J., "The complexity of dynamic languages and dynamic optimization problems," *Proc. 13th STOC* (1981), 218-227.

[12] Papadimitriou, C. H., "Games against nature," *J. Comput. Syst. Sci.*, 31 (1985), 288-301.

[13] Papadimitriou, C. H. and Steiglitz, K., *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, NJ (1982).

[14] Papadimitriou, C. H. and Tsitsiklis, J. N., "The complexity of Markov decision processes," *Math. Oper. Res.*, 12 (1987), 441-450.

[15] Parberry, I., *Parallel Complexity Theory*, Pitman, London (1987).

[16] White, D. J., "Elimination of nonoptimal actions in Markov decision processes," in (Puterman, M. L. ed.) *Dynamic Programming and Its Applications*, Academic Press, New York, NY (1978), 131-160.

Appendix A

We briefly explain below the complexity terms PSPACE, NP, P, P-hard, P-complete and NC. [See [14] for a more detailed explanation. Also see [3], [5], [9], [13], [15] for comprehensive discussions.]

PSPACE is the class of problems that can be solved using polynomial space.

NP is the class of problems that can be solved nondeterministically in polynomial time (e.g. independent set, Hamilton circuit problem, integer programming).

P is the class of problems that can be solved in polynomial time (e.g. linear program).

A problem is P-hard if any problem in P is reducible to it using logarithmic space.

A problem is P-complete if it is both P-hard and in P.

NC is the class of problems that can be solved in parallel using a polynomial number of processors in time that is a polynomial in the logarithm of the input size.

The following hierarchy (in order of increasing difficulty) for the above problem classes are known to hold: NC $\subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE and P-complete $\subseteq$ P. Notice that if any P-hard problem is shown to be in NC, then P = NC.

Appendix B

Let A be any n×n matrix. We show below that, for any integer $k \geq 1$, we can compute A^k and $I + A + \ldots + A^k$ in $O(\log(k))$ matrix multiplications. First suppose that k is a power of 2. Then, by using the recursive equations

$$A^k = (A^{k/2})(A^{k/2}),$$

$$I + A + \ldots + A^k = (I + A + \ldots + A^{k/2}) + A^{k/2}(I + A + \ldots + A^{k/2}),$$

we see that if $A^{k/2}$ and $I + A + \ldots + A^{k/2}$ are computable in $3\log(k/2)$ matrix multiplications, then A^k and $I + A + \ldots + A^k$ are computable in $3\log(k/2) + 3 = 3\log(k)$ matrix multiplications. Hence, by induction, we can compute A^d and $I + A + \ldots + A^d$ for <u>all</u> $d = 2^0, 2^1, 2^2, \ldots, k$ in $3\log(k)$ matrix multiplications. Now suppose that k is not a power of 2. Let us first compute and store the matrices

$$A^d \quad \text{and} \quad I + A + \ldots + A^{d-1} + A^d,$$

for all $d = 2^0, 2^1, 2^2, \ldots, 2^h$, where $h = \lfloor \log(k) \rfloor$. [This takes 3h matrix multiplications as we argued above.] We claim that, given the above matrices, A^i and $I + A + \ldots + A^i$ are computable in $3\lceil \log(i) \rceil$ matrix multiplications for any positive integer $i \leq k$. This claim clearly holds for $i = 1$. Suppose that it holds for all i up to (but not including) some $r \in \{2,3,\ldots,k\}$. Then by first computing A^{r-d} and $I + A + \ldots + A^{r-d}$, where d is the largest power of 2 less than r, and then using the identities

$$A^r = (A^d)(A^{r-d}),$$

$$I + A + \ldots + A^r = (I + A + \ldots + A^d) + A^d(I + A + \ldots + A^{r-d}),$$

we can compute A^r and $I + A + \ldots + A^r$ in $3\lceil \log(r-d) \rceil + 3$ matrix multiplications. Since $r-d \leq d$, this bound is less than $3\log(d) + 3 = 3\log(2d)$. Since d is the largest power of 2 less than r, we have $d < r \leq 2d$ so that $\lceil \log(r) \rceil = \log(2d)$. This then completes the induction.

Appendix C

Consider the following H-horizon, discounted Markov decision problem:

$$\alpha = .5, \quad g^1 = (0, 0), \quad g^2 = (0, 0), \quad H < +\infty,$$

$$P^1 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad P^2 = \begin{bmatrix} .5 & .5 \\ .5 & .5 \end{bmatrix}.$$

Then

$$T(x) = .5 \begin{bmatrix} \min\{x_2, .5(x_1+x_2)\} \\ \min\{x_1, .5(x_1+x_2)\} \end{bmatrix},$$

and $T_1(x) < T_2(x)$ if $x_1 > x_2$ while $T_1(x) > T_2(x)$ if $x_1 < x_2$. Therefore if $x_1(H) \neq x_2(H)$, then the optimal control at time t would alternate between (1,2) and (2,1), depending on whether t is odd or even. If $x_1(H) = x_2(H)$, then any sequence of controls is optimal. Note that, for this example, any of the four stationary policies is optimal for the ∞-horizon version of the problem.