Improved successive approximation methods for discounted Markov decision processes

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics

STATISTICS AND OPERATION RESEARCH GROUP

Memorandum COSOR 74-06

Improved successive approximation methods for discounted Markov decision processes

by

J.A.E.E. van Nunen

Eindhoven, April 1974

Abstract

Successive Approximation (S.A.) methods, for solving discounted Markov decision problems, have been developed to avoid the extensive computations that are connected with linear programming and policy iteration techniques for solving large scaled problems. Several authors give such an S.A. algorithm.
In this paper we introduce some new algorithms while furthermore it will be shown how the several S.A. algorithms may be combined. For each algorithm converging sequences of upper and lower bounds for the optimal value will be given.

§ 1. Introduction.

We consider a finite state, discrete time Markov system that is controlled by a decisionmaker (see for example [4]). After each transition $n = 0,1,2,\ldots$ the system may be identified as being in one of N possible states. Let $S := \{1,2,\ldots,N\}$ represent the set of states. After observing state $i \in S$ the decisionmaker selects an action k from a nonempty finite set K(i). Now p_{ij}^k is the probability of a transition to state $j \in S$, if the system is actually in state $i \in S$, and action $k \in K(i)$ has been selected. An (expected) reward $q^k(i)$ is then earned immediately, while future income is discounted by a constant factor $0 \leq \alpha < 1$.
We suppose, which is permitted without loss of generality, that $q^k(i) \geq 0$ for all $i \in S$ and $k \in K(i)$.

The problem is to choose a policy which maximizes the total expected discounted return.

As known (e.g. [2], [10]), it is permitted to restrict the considerations to nonrandomized stationary strategies. A nonrandomized stationary strategy will be denoted by $f \in K := K(1) \times K(2) \times \ldots \times K(N)$. The coordinates $u_f(i)$ of the $N \times 1$ vector u_f give the total expected discounted return if the system's initial state is i and the stationary strategy $f \in K$ is used. The (stationary) strategy $f_* \in K$ is called optimal if $u_{f_*}(i) \geq u_f(i)$ for all $f \in K$ and for all $i \in S$.

Because S.A. algorithms are in some sense modifications of the standard dynamic programming method this method will be discussed first.

As in Blackwell [1] we define for each $f \in K$ the mapping $L_b(f)$ $(\mathbb{R}^N \to \mathbb{R}^N)$ which maps an $N \times 1$ column vector x into

$$L_b(f)x := q_f + \alpha P_f x ,$$

where q_f is the $N \times 1$ column vector having as its i-th component $q^{f(i)}(i)$, and P_f is the $N \times N$ Markov matrix with (i,j) element $p_{ij}^{f(i)}$.
$L_b(f)$ is monotone, i.e. if every coordinate of the $N \times 1$ vector x is at least as large as every coordinate of $y \in \mathbb{R}^N$ $(x \geq y)$, then:

$$L_b(f)x \geq L_b(f)y .$$

Furthermore, we define for some map $L_\beta(f)$:

$$L_\beta^0(f)x := x ;$$

$$L_\beta^n(f)x = L_\beta(f)(L_\beta^{n-1}(f)x) .$$

We define the mapping U_b: $\mathbb{R}^N \to \mathbb{R}^N$ by:

$$U_b x := \max_{f \in K} L_b(f)x .$$

It is easily seen that for every $x \in \mathbb{R}^N$ an $f \in K$ exists such that $L_b(f)x$ is maximal for each coordinate.
It may be proved that U_b is a monotone α-contraction mapping with fixed point u_*. For an optimal strategy f_* we have $u_* = u_{f_*}$, and a stationary strategy f which is optimal follows from $L_b(f)u_* := U_b u_*$ (see for instance [9]).
This property legitimates the standard dynamic programming algorithm that can be based on:

$$\text{I} \begin{cases} x_0^b := 0 \\ x_n^b := U_b\, x_{n-1}^b =: L_b(f_n^b)x_{n-1}^b . \end{cases}$$

It is possible to take x_n^b and f_n^b as estimates for u_{f_*} and f_* as follows from:

$$(1) \qquad x^b_{n-1} \le x^b_n \le u_{f^b_n} \le u_{f_*}$$

$$(2) \qquad \lim_{n\to\infty} x^b_n = \lim_{n\to\infty} U^n_b\, x^b_0 = u_{f_*},$$

see [5], [7], [9].

As starting vector we choose $x^b_0 = 0$. As appears from (1) and (2), this choice guarantees monotone convergence of x^b_n to u_{f_*}.

As known, the convergence, depending on α, may be relatively slow. MacQueen [5] constructed upper and more sophisticated lower bounds for $u_{f^b_n}$ and u_{f_*}.

The S.A. methods discussed in the following sections are based on contraction mappings (see Denardo [2]).

In section 2, S.A. methods based on mappings U_0, U_a, U_s) of the same type as U_b will be given; i.e. also these mappings are monotone contraction mappings with fixed point u_{f_*}. Furthermore, combinations of these mappings which lead to mappings (U_{hs}, U_{h0}) with the same property will be discussed. In section 3 extension of the above algorithms will be given, while in section 4 upper and lower bounds for several methods are discussed. This enables us to incorporate a test for the suboptimality of actions (see also [6]). Finally (section 5) some examples are given to illustrate several methods.

§ 2. "Improved" successive approximation methods.

2.1. Hastings [3] introduced the following (Gauss-Seidel) idea to modify the policy improvement procedure in Howard's policy iteration algorithm. Let u_f for a given strategy $f \in K$ be computed. Determine a better strategy $g \in K$ with components $g(i)$ as follows:

$$g(1) \text{ follows from } v_g(1) := \max_{k\in K(1)} \{q^k(1) + \alpha \sum_{j=1}^{N} p^k_{ij}\, u_f(j)\}$$

$$=: \{q^{g(1)}(1) + \alpha \sum_{j=1}^{N} p^{g(1)}_{ij}\, u_f(j)\},$$

$g(i)$ follows from $v_g(i) := \max_{k \in K(i)} \{q^k(i) + \alpha \sum_{j<i} p_{ij}^k v_g(j) + \alpha \sum_{j \geq i} p_{ij}^k u_f(j)\}$

$$=: \{q^{g(i)}(i) + \alpha \sum_{j<i} p_{ij}^{g(i)} v_g(j) + \alpha \sum_{j \geq i} p_{ij}^{g(i)} u_f(j)\} .$$

This idea can also be used in an S.A. algorithm.

Let $x \in \mathbb{R}^N$. Define $L_h(f)$ by:

$$L_h(f)x(i) := q^{f(i)}(i) + \alpha \sum_{j<i} p_{ij}^{f(i)} (L_h(f)x(j)) + \alpha \sum_{j \geq i} p_{ij}^{f(i)} x(j) , \quad i \in S .$$

Define the mapping U_h by:

$$U_h x := \max_f L_h(f)x .$$

It is easily verified that $L_h(f)$ and U_h are monotone α-contractions with fixed point u_f and u_{f_*}, respectively, so an S.A. algorithm might be based on

$$\text{II} \begin{cases} x_0^h := 0 \\ x_n^h := U_h x_{n-1}^h =: L_h(f_n^h) x_{n-1}^h . \end{cases}$$

As in standard dynamic programming, the sequence $\{x_n^h\}$ will have the following properties:

$$x_{n-1}^h \leq x_n^h \leq u_{f_n^h} \leq u_{f_*} , \tag{3}$$

$$\lim_{n \to \infty} x_n^h := u_{f_*} . \tag{4}$$

Furthermore, a comparison with the x_n^b of the dynamic programming algorithm yields inductively

$$x_n^h \geq x_n^b . \tag{5}$$

2.2. Also "overrelaxation" (see [8], [9]) may be used in successive approximation algorithms. Where the overrelaxation factor appears, for instance, if we try to find better estimates for u_f by computing for certain paths the exact contribution to the total expected discounted reward.
Let $f \in K$ be given, then

$$u_f = L_f u_f = q_f + \alpha P_f u_f \ .$$

Another expression for $u_f(i)$ may be found by computing the contribution to the expected reward until the time the system leaves i explicitly ($f(i) =: k$):

$$\begin{aligned} (6) \qquad u_f(i) &= q^k(i) + \alpha p_{ii}^k \, q^k(i) + (\alpha p_{ii}^k)^2 \, q^k(i) + \ldots \\ &\quad + \alpha \sum_{j \neq i} p_{ij}^k \, u_f(j) \\ &\quad + \alpha^2 p_{ii}^k \sum_{j \neq i} p_{ij}^k \, u_f(j) \\ &\quad \vdots \\ &= \frac{1}{1 - \alpha p_{ii}^k} \, q^k(i) + \frac{\alpha}{1 - \alpha p_{ii}^k} \sum_{j \neq i} p_{ij}^k \, u_f(j) \ . \end{aligned}$$

Let $\omega_i^k := \dfrac{1}{1 - \alpha p_{ii}^k}$, then with $k = f(i)$, (6) can be given as

$$(7) \qquad u_f(i) = \omega_i^k \, q^k(i) + \alpha \omega_i^k \sum_{j \neq i} p_{ij}^k \, u_f(j) \ .$$

(7) can also be deduced from:

$$u_f(i) = q^k(i) + \alpha \sum_j p_{ij}^k \, u_f(j) \ ,$$

which yields:

$$(1 - \alpha p_{ii}^k) u_f(i) = q^k(i) + \alpha \sum_{j \neq i} p_{ij}^k \, u_f(j) \ ,$$

where (7) follows by dividing by $(1 - \alpha p_{ii}^k)$.
On the idea used in (7) we base for any $f \in K$ the mapping $L_0(f)$ defined by:

$$L_0(f)x(i) := \omega_i^k \, q^k(i) + \alpha\omega_i^k \sum_{j\neq i} p_{ij}^k \, x(j) \quad \text{with } k = f(i) \; .$$

Furthermore we define the mapping U_0 by:

$$U_0 x = \max_{f\in K} \{L_0(f)x\} \; .$$

Let

$$\omega^-(f) := \min_{i\in S} \{\omega_i^{f(i)}\} \; ,$$

$$\omega^+(f) := \max_{i\in S} \{\omega_i^{f(i)}\} \; ,$$

$$\gamma(\omega) := 1 - \omega(1-\alpha) \; .$$

Then $L_0(f)$ is a monotone $\gamma(\omega^-(f))$-contraction with fixed point u_f. It is easily verified that $\gamma(\omega^-(f)) \leq \alpha$.

Let $\omega^* := \min_{i,k} \{\omega_i^k\}$, then U_0 is a monotone $\gamma(\omega^*)$-contraction with fixed point u_{f_*} (see [8]).
We have the relation:

$$\gamma(\omega^-(f)) \leq \gamma(\omega^*) \leq \alpha \; .$$

Hence a successive approximation method might be based on

III
$$\begin{cases} x_0^0 := 0 \\ x_n^0 := U_0 x_{n-1}^0 =: L_0(f_n^0)x_{n-1}^0 \; , \end{cases}$$

where the following inequalities are easily proved:

(8) $$x_{n-1}^0 \leq x_n^0 \leq u_{f_n^0} \leq u_{f_*} \; ,$$

(9) $$x_n^b \leq x_n^0 \; .$$

2.4. It is also possible to simplify algorithm III by using the fixed overrelaxation factor ω^*, which means that the contribution to the expected reward until the system leaves state i is only estimated.

Then we define $L_s(f)$ by:

$$L_s(f)x(i) := \omega^* q^k(i) + \alpha\omega^* \sum_{j\in S} p_{ij}^k x(j) + (1-\omega^*)x(i) .$$

U_s is defined by:

$$U_s x := \max_{f\in K} L_s(f)x .$$

$L_s(f)$ and U_s are monotone $\gamma(\omega^*)$-contractions with fixed point u_f and u_{f_*}, respectively. So it is possible to construct an S.A. algorithm based on:

$$\text{IV} \begin{cases} x_0^s := 0 \\ x_n^s := U_s x_{n-1}^s =: L_s(f_n^s)x_{n-1}^s . \end{cases}$$

Again we have:

$$(10) \qquad x_{n-1}^s \le x_n^s \le u_{f_n^s} \le u_{f_*}$$

$$(11) \qquad x_n^b \le x_n^s .$$

§ 3. Combinations of S.A. algorithms.

In this section it will be shown that combinations of the mappings U_h, U_0, U_s lead to mappings U_{h0}, U_{hs}, with the same properties as the original mappings; i.e. U_{h0}, U_{hs} are monotone contractions with fixed point u_{f_*}.
First we want to combine the transformations U_0 and U_h as is done in a modified form by Reetz [8].
We define the transformation $L_{h0}(f)$ inductively by

$$L_{h0}(f)x(i) := \omega_i^k q^k(i) + \alpha\omega_i^k \sum_{j<i} p_{ij}^k L_{h0}(f)x(j) + \alpha\omega_i^k \sum_{j>i} p_{ij}^k x(j) ,$$

with $k = f(i)$,

and U_{h0} by:

$$U_{h0}x = \max_f L_{h0}(f)x .$$

Then, $L_{h0}(f)$ and U_{h0} are monotone and $\gamma(\omega^*)$-contractions with fixed point u_f and u_{f_*}, respectively.
So we have

$$V \quad \begin{cases} x_0^{h0} = 0 \\ x_n^{h0} := U_{h0} x_{n-1}^{h0} =: L_{h0}(f_n^{h0}) x_{n-1}^{h0} . \end{cases}$$

with

$$(12) \qquad x_{n-1}^{h0} \le x_n^{h0} \le u_{f_n^{h0}} \le u_{f_*} ,$$

$$\lim_{n\to\infty} x_n^{h0} = u_{f_*} .$$

Furthermore,

$$(13) \qquad x_n^0 \le x_n^{h0} ,$$

$$(14) \qquad x_n^h \le x_n^{h0} .$$

The original Reetz [8] algorithm can be found as a combination of the transformations U_s and U_h.
Let $L_{hs}(f)$ be given by

$$L_{hs}(f)x(i) := \omega^* q^k(i) + \alpha\omega^* \sum_{j<i} p_{ij}^k L_{hs}(f)x(j) + \alpha\omega^* \sum_{j\ge i} p_{ij}^k x(j) + (1-\omega^*)x(i)$$

and

$$U_{hs}(f)x = \max_f L_{hs}(f)x .$$

$L_{hs}(f)$ and U_{hs} are monotone and $\gamma(\omega^*)$-contractions with fixed point u_f and u_{f_*}, respectively.
We have:

$$VI \quad \begin{cases} x_0^{hs} = 0 \\ x_n^{hs} = U_{hs} x_{n-1}^{hs} =: L_{hs}(f_n^{hs}) x_{n-1}^{hs} \end{cases}$$

with

$$x_{n-1}^{hs} \leq x_n^{hs} \leq u_{f_n^{hs}} \leq u_{f_*} ,$$

$$\lim_{n\to\infty} x_n^{hs} = u_{f_*} ,$$

$$x_n^s \leq x_n^{hs} ,$$

$$x_n^h \leq x_n^{hs} .$$

§ 4. Extensions of S.A. algorithms.

A method to improve the estimations for u_{f_*} can also be found by inserting a number of value determination iteration steps in the S.A. algorithm based on U_β where $\beta \in T := \{b,h,0,s,hs,h0\}$, see [7].
This idea can also be introduced as the skipping of a number of policy improvement iteration steps in the S.A. algorithms.
We define for each $x \in \mathbb{R}^N$ and for finite $\lambda \in \mathbb{N}$ and for $\beta \in T$ the mapping $U_\beta^{(\lambda)}$ by:

$$U_\beta^{(\lambda)} x := L_\beta^{\lambda-1}(f^{\beta\lambda}) U_\beta x$$

where $f^{\beta\lambda}$ indicates the strategy that is found by applying U_β on x.
For $\lambda \in \mathbb{N}$, $\lambda > 1$, $U_\beta^{(\lambda)}$ is neither necessarily a contraction mapping nor a monotone mapping.
However, we may base an algorithm on such a mapping:

VII-XII
$$\begin{cases} x_0^{\beta\lambda} = 0 , & \beta \in T \\ x_n^{\beta\lambda} = U_\beta^{(\lambda)} x_{n-1}^{\beta\lambda} =: L_\beta^{\lambda}(f_n^{\beta\lambda}) x_{n-1}^{\beta\lambda} , & \beta \in T . \end{cases}$$

The monotone convergence of $x_n^{\beta\lambda}$ to u_{f_*} is preserved (see [7]) as follows from the monotonicity of U_β and $L_\beta(f^\beta)$, i.e.

$$\lim_{n\to\infty} x_n^{\beta\lambda} = u_{f_*} ,$$

$$x_{n-1}^{\beta\lambda} \le x_n^{\beta\lambda} \le u_{f_n^{\beta\lambda}} \le u_{f_*} .$$

A comparison of $x_n^{\beta\lambda}$ with x_n^{β} yields:

$$x_n^{\beta} \le x_n^{\beta\lambda} , \qquad n \in \mathbb{N},\ \beta \in T .$$

§ 5. Upper and lower bounds for u_{f_*}.

Successive approximation algorithms based on the ideas of the previous sections will converge. However, it will be necessary to construct upper and lower bounds for the current and the optimal strategy. Upper and lower bounds enable us to qualify the estimates for u_{f_n}, u_{f_*} and f_*, respectively, see for instance MacQueen [5].
Also upper and lower bounds enable us to incorporate a test for the suboptimality of decisions in an algorithm (see MacQueen [6]).
Let the upper bound $\bar{x}$ and the lower bound $\underline{x}$ for u_* be given, then we can state the following lemma:

Lemma 1. Strategy f is suboptimal if

$$L_\beta(f)\bar{x} < U_\beta \underline{x} , \qquad \beta \in T .$$

Proof.

$$u_* = U_\beta u_* \ge U_\beta \underline{x} > L_\beta(f)\bar{x} \ge L_\beta(f)u_* ,$$

where the monotonicity property of U_β and $L_\beta(f)$ is used. □

This lemma enables us to determine for each $i \in S$ decisions which are suboptimal (see for instance [6]). *

---

* Note that in the algorithms where U_s is used, ω^* can be redefined if the decision that causes ω^* is suboptimal.

If we want to compare two algorithms it will be necessary to compare the corresponding sequences of upper and lower bounds. However, where the estimates for u_{f_*} found in the n-th iteration step of a specific algorithm may be better than those of another algorithm (as shown in the previous sections), this doesn't mean unfortunately that it is possible to construct bounds that are "better" too.
We will illustrate this phenomenon with some examples (see section 6), However, we want to give without proof some general statements about upper and lower bounds first.

Lemma 2. For U_β, $\beta \in T$, the sequence

$$\bar{x}_n^\beta := x_{n-1}^\beta + \frac{1}{1 - c(\beta)} \|x_n^\beta - x_{n-1}^\beta\|_\infty , \qquad n \in \mathbb{N} ,$$

yields monotone nonincreasing upper bounds for u_{f_*}. Where $c(\beta)$ is the contraction factor corresponding with U_β and where

$$\|x - y\|_\infty := \max_i |x(i) - y(i)| , \qquad x,y \in \mathbb{R}^N .$$

Furthermore,

$$\lim_{n\to\infty} \bar{x}_n^\beta = u_{f_*} . \qquad \square$$

Lemma 3. For $U_\beta^{(\lambda)}$, $\beta \in T$, $\lambda \in \mathbb{N}$, the sequence

$$\bar{x}_n^{\beta\lambda} := \min \{\bar{x}_{n-1}^{\beta\lambda} , x_{n-1}^{\beta\lambda} + \frac{1}{1 - c(\beta)} \|U_\beta x_{n-1}^{\beta\lambda} - x_{n-1}^{\beta\lambda}\|_\infty$$

yields monotone nonincreasing upper bounds for u_{f_*}. Furthermore

$$\lim_{n\to\infty} \bar{x}_n^{\beta\lambda} = u_{f_*} . \qquad \square$$

It is also possible to construct a monotone nondecreasing sequence of lower bounds for $u_{f_n^\beta}$ and $u_{f_n^{\beta\lambda}}$, and so for u_{f_*}. Such a sequence can be formed trivially by using the x_n^β, $x_n^{\beta\lambda}$, $\beta \in T$, respectively.

We will now give sequences of lower bounds that might be used for the several methods described in the previous sections.

Lemma 4. For U_β, $\beta \in T$, the sequence

$$\underline{x}_n^\beta := x_{n-1}^\beta + \frac{1}{1-\delta(\beta)} \|x_n^\beta - x_{n-1}^\beta\|_{-\infty}$$

yields monotone nondecreasing lower bounds for $u_{f_n^\beta}$ and so for u_{f_*} and

$$\lim_{n\to\infty} \underline{x}_n^\beta = u_{f_*} ,$$

where $\delta(b) = \alpha$; $\delta(h) = \alpha^N$, $\delta(0) = \gamma(\omega^+(f_n^0))$, $\delta(s) = \gamma(\omega^*)$, $\delta(h0) = \gamma(\omega^+(f_n^{h0}))^N$, $\delta(hs) = (\gamma(\omega^*))^N$ and

$$\|x_n^\beta - x_{n-1}^\beta\|_{-\infty} := \min_{i\in S} (x_n^\beta(i) - x_{n-1}^\beta(i)) . \qquad \square$$

Lemma 5. For $U_\beta^{(\lambda)}$, $\beta \in T$, $\lambda \in \mathbf{N}$, the sequence

$$\underline{x}_n^{\beta\lambda} := x_{n-1}^{\beta\lambda} + \frac{1}{1-\delta(\beta)} \|U_\beta x_{n-1}^{\beta\lambda} - x_{n-1}^{\beta\lambda}\|_{-\infty}$$

yields monotone nondecreasing lower bounds for $u_{f_n^{\beta\lambda}}$ and so for u_{f_*}; furthermore

$$\lim_{n\to\infty} \underline{x}_n^{\beta\lambda} = u_{f_*} . \qquad \square$$

For all the bounds we have a monotone convergence to u_{f_*}. So each number of the indicated set of algorithms can be used to estimate the optimal policy f_* and the corresponding value vector u_{f_*}.

The examples in section 6 show that a choice for a specific algorithm may depend on the problem under consideration.

§ 6. Examples.

In this section we will give two simple examples to illustrate that the decision which algorithm has to be chosen, might depend on the problem under consideration.

Example 1. In this example we compare the distance between the upper and lower bounds in the n-th iteration step of algorithm I and this distance in the n-th iteration step of algorithm IV.
Consider a two state problem with in each state only one possible decision. Let the matrix of transition probabilities be given by:

$$P := \begin{pmatrix} p & 1-p \\ 1-p & p \end{pmatrix}$$

and the reward vector r by: $r := \begin{pmatrix} r_1 \\ r_2 \end{pmatrix}$, $(r_1 \geq r_2)$, with discount factor α.
Then algorithm I yields

$$x_n^b = \sum_{k=0}^{n-1} \alpha^k P^k r ,$$

so

$$x_n^b - x_{n-1}^b = \alpha^{n-1} P^{n-1} r = \alpha^{n-1} \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} \end{pmatrix} r + (-\alpha(1-2p))^{n-1} \begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix} r .$$

This yields:

$$D^b(n) := \|x_n^b - x_{n-1}^b\|_\infty - \|x_n^b - x_{n-1}^b\|_{-\infty} = (\alpha|1-2p|)^{n-1}(r_1 - r_2) .$$

Using algorithm IV yields in a similar way

$$D^s(n) := \|x_n^s - x_{n-1}^s\|_\infty - \|x_n^s - x_{n-1}^s\|_{-\infty} = \left(\frac{\alpha(1-p)}{1-\alpha p}\right)^{n-1} \frac{1}{1-\alpha p} (r_1 - r_2) .$$

Let A_n be defined by:

$$A_n := \frac{D^b(n)}{D^s(n)} = c \frac{|1-2p|^{n-1}(1-\alpha p)^{n-1}}{(1-p)^{n-1}} ,$$

where c is a constant which is independent of n.

Then

$$\lim_{n\to\infty} Q_n = \infty \quad \text{if} \quad p > p_1 = \left(\frac{3+\alpha}{4\alpha}\right) - \sqrt{\left(\frac{3+\alpha}{4\alpha}\right)^2 - 1}$$

$$\lim_{n\to\infty} Q_n = 0 \quad \text{if} \quad p < p_1 \ .$$

For this problem this leads to the conclusion that algorithm I is preferable if $p < p_1$.

Example 2. Consider a two state problem with $K(1) = \{1\}$, $K(2) = \{1,2\}$, $a = 0,9$, and

$$p_{11}^1 = 1 \ , \quad p_{12}^1 = 0 \ , \quad r^1(1) = 2$$

$$p_{21}^1 = 0 \ , \quad p_{22}^1 = 1 \ , \quad r^1(2) = 2$$

$$p_{22}^2 = 1 \ , \quad p_{22}^2 = 0 \ , \quad r^2(2) = 1,9 \ .$$

The Hastings algorithm II will start in state 2 with the suboptimal decision 2, while (MacQueen) algorithm I starts with selecting the optimal decision 1. Furthermore, the upper and lower bounds corresponding to algorithm I are equal, which means that the optimal values u_{f_*} are known in one step.

References.

[1] Blackwell, D., Discounted dynamic programming. Am. Math. Stat. 36 (1965), 226-234.

[2] Denardo, E.V., Contraction mappings in the theory underlying dynamic programming. SIAM Review 9 (1967), 165-177.

[3] Hastings, N., Some notes on Dynamic Programming and Replacement. Opl. Res. Q. 19 (1968), 453-464.

[4] Howard, R., Dynamic Programming and Markov Processes. M.I.T. Press, Cambridge (1960).

[5] MacQueen, J., A modified dynamic programming method for Markovian decision problems. J. Math. An. Appl. 14 (1966), 38-43.

[6] MacQueen, J., A test for suboptimal actions in Markovian decision problems. O.R. 15 (1967), 559-561.

[7] Nunen, J. van, A set of successive approximation methods for discounted Markovian decision problems. Memorandum COSOR 73-09, Department of Math., Techn. Univ. Eindhoven, Netherlands.

[8] Reetz, D., Solution of a Markovian Decision Problem by Successive Overrelaxation. Zeitschrift Operat. Res. 17 (1973), 29-32.

[9] Schellhaas, N., Zur Extrapolation in Markoffschen Entscheidungsmodellen mit Diskontierung (1973), preprint nr. 84. Technische Hochschule Darmstadt, West-Germany.

[10] Wessels, J. and Nunen, J. van, Discounted semi Markov decision processes: linear programming and policy iteration. Memorandum COSOR 74-01, Department of Math., Techn. Univ. Eindhoven, Netherlands.