Operations Research Letters 10 (1991) 193–202
North-Holland

Criteria for selecting the relaxation factor of the value iteration algorithm for undiscounted Markov and semi-Markov decision processes

Meir Herzberg

Engineering and Planning Division BEZEQ, The Israel Telecommunication Co., Tel Aviv 61290, Israel

Uri Yechiali

Department of Statistics, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel

Received September 1989
Revised May 1990

We present two criteria for selecting the adaptive relaxation factor being used in speeding-up the value iteration algorithm for undiscounted Markov decision processes. The criteria are: 1. Minimum Ratio, 2. Minimum Variance. For the problems tested it was found that the criterion of minimum variance is most effective for Markov models while an hybrid use of both criteria yields best results when solving semi-Markov decision problems. The advantage of using these criteria increases with the dimension of the models.

Markov and semi-Markov decision processes * value iteration algorithm * adaptive relaxation factor

1. Introduction

The value iteration algorithm, based on a dynamic programming approach, is one of the best computational methods for solving large scale undiscounted Markov decision models, avoiding either the repeated solution of a system of simultaneous linear equations (as required when adopting Howard's Policy Iteration Algorithm) or solving large scale Linear Programming problems (see, for example, Tijms [6]).

The algorithm may readily be applied to semi-Markov decision models (see Schweitzer [4]) by performing a direct data transformation on both the cost values and the one-step transition probabilities.

The value iteration algorithm achieves at each iteration a cost vector with lower and upper bounds on the optimal average cost per unit time. Under the assumption that the finite Markov chain is aperiodic for each stationary policy, these bounds converge geometrically to the optimal cost rate (see Schweitzer and Federgruen [5]). Varaiya [8] suggested changing the cost vector achieved at each iteration by introducing a modified parameter. This idea was applied by Popyack et al. [3] who introduced an Adaptive Relaxation Factor (ARF) by taking into consideration the lower and upper cost bounds and the associated states for which these bounds are obtained. It was demonstrated that introducing the above factor might greatly enhance and speed-up the convergence of the value iteration algorithm (see Tijms [6], Tijms and Eikeboom [7]).

The purpose of this paper is to propose criteria for an improved selection of the ARF in order to further speed-up the convergence of the algorithm. In Section 2, we discuss aspects of the value iteration algorithm

0167-6377/91/$03.50

and emphasize the one-step look-ahead approach. In Sections 3 and 4, we introduce the Minimum Ratio and Minimum Variance criteria, respectively. In Section 5, we present some numerical results and discuss the relative advantage of the proposed criteria.

2. The modified value iteration algorithm

The recursive equation on which the standard value iteration for undiscounted Markov decision processes is based can be presented (see Tijms [6]) by:

$$V_n(i) = \operatorname{Min}_{a \in A_i} \left\{ C_i(a) + \sum_{j \in I} P_{ij}(a) V_{n-1}(j) \right\}, \quad i \in I \tag{1}$$

where,

$V_0(i)$, $i \in I$, is an arbitrary chosen cost function, with $0 \le V_0(i) \le \operatorname{Min}_{a \in A_i} C_i(a)$,

$V_n(i)$ = minimal total expected cost when starting at state i, moving n steps and paying terminal cost $V_0(j)$ if the process ends up at state j,

A_i = set of possible actions (decisions) admissible in state i,

$C_i(a)$ = one-step expected cost associated with the selection of action a while in state i,

$P_{ij}(a)$ = one-step transition probability from state i to state j when selecting action a,

I = set of possible states of the analyzed system.

Without loss of generality we can assume that the values $C_i(a)$ ($i \in I$, $a \in A_i$) are positive, since adding a constant to every $C_i(a)$ merely adds the same constant to the cost rate. In addition, the selection of $V_0(i)$ is aimed at achieving a good starting point for the algorithm. (In fact, another direction of investigation might be how to select effectively the initial vector V_0.)

Consider now the differences,

$$\delta_n(i) = V_n(i) - V_{n-1}(i), \quad i \in I. \tag{2}$$

For large n the values $\delta_n(i)$ represent very closely the minimal average cost per unit time, as $V_n(i) \sim n \cdot \delta_n(i)$.

The algorithm stops at a certain iteration n when the values $\delta_n(i)$, $i \in I$, become close enough to each other. Following Tijms [6, p. 192], we use a relative accuracy stopping criterion, i.e.,

$$\operatorname{Max}_{i \in I} \{ \delta_n(i) \} / \operatorname{Min}_{i \in I} \{ \delta_n(i) \} \le 1 + \varepsilon \tag{3}$$

where ε is a predetermined tolerance error, and where n must be chosen large enough that the denominator is positive. Note that if $V_0(i) = 0$ for all i, then $\operatorname{Min}_{i \in I} \delta_1(i) > 0$. The actions for which (1) is satisfied when the algorithm stops comprise the optimal stationary decision policy, provided that ε is chosen sufficiently small. Moreover, for any $\varepsilon > 0$, Tijms [6] showed that the actions for which (1) is satisfied comprise a policy which will achieve the optimal cost rate to within a relative error of ε. (An alternative stopping rule might be to apply an absolute accuracy criterion, such as $0 \le \operatorname{Max}_i \delta_n(i) - \operatorname{Min}_i \delta_n(i) \le \varepsilon$. However, this criterion seems to us to be particularly suitable for *discounted* MDPs.)

It should be noted that two assumptions are needed to ensure convergence of (1), in the sense that $\delta_n(i) \to$ (cost rate) as $n \to \infty$. First, for this to occur all components of the cost vector must be equal. This holds if the Markov chain is single-chained for every policy. Platzmann [2] relaxed this assumption and showed that the value-iteration procedure as described in Schweitzer [4] will converge if the minimal cost rate is state-invariant. This weaker condition may be verified by means of state connectivity tests. Second, as mentioned above, *aperiodicity* is needed. This can be achieved by *under-relaxation* (see [4]) which will force aperiodicity.

In order to speed-up the value iteration algorithm, modified values $\bar{V}_n(i)$, $i \in I$, are used for the next iteration as follows:

$$\bar{V}_n(i) = V_{n-1}(i) + w[V_n(i) - V_{n-1}(i)] = V_{n-1}(i) + w\delta_n(i), \quad i \in I \tag{4}$$

where w is an ARF. (Clearly, for $w = 1$, one gets the standard algorithm where $\bar{V}_n(i) = V_n(i)$.)

For the semi-Markov case a similar development can be achieved, starting with the modified recursive equation,

$$V_n(i) = \operatorname{Min}_{a \in A_i} \left\{ C_i(a)/\tau_i(a) + \sum_{j \in I} \tilde{P}_{ij}(a) V_{n-1}(j) \right\}, \quad i \in I, \tag{5}$$

where

$$\tilde{P}_{ij}(a) = \begin{cases} P_{ij}(a) * \tau/\tau_i(a), & i \neq j, \\ P_{ij}(a) * \tau/\tau_i(a) + (1 - \tau/\tau_i(a)), & i = j, \end{cases} \tag{6}$$

$\tau_i(a)$ = expected sojourn time of the system in state i when selecting action a, and

$$0 < \tau < \operatorname{Min}_{i \in I,\, a \in A_i} \{ \tau_i(a) \}. \tag{7}$$

In order to explore the effect of replacing the original $V_n(i)$'s with the modified values $\bar{V}_n(i)$, we perform a one-step look-ahead analysis. We start with the Markov decision processes and use the adaptive terms $\hat{V}_{n+1}(i)$ and $\hat{\delta}_{n+1}(i)$, $i \in I$, where,

$$\hat{V}_{n+1}(i) = C_i(R_i) + \sum_{j \in I} P_{ij}(R_i) \bar{V}_n(j), \tag{8}$$

R_i = selected action for state i, determined by (1) at iteration n, and

$$\hat{\delta}_{n+1}(i) = \hat{V}_{n+1}(i) - \bar{V}_n(i). \tag{9}$$

Then, from (4),

$$\begin{aligned} \hat{\delta}_{n+1}(i) &= C_i(R_i) + \sum_{j \in I} P_{ij}(R_i)[V_{n-1}(j) + w\delta_n(j)] - [V_{n-1}(i) + w\delta_n(i)] \\ &= C_i(R_i) + \sum_{j \in I} P_{ij}(R_i) V_{n-1}(j) + w\left[\sum_{j \in I} P_{ij}(R_i)\delta_n(j) - \delta_n(i) \right] - V_{n-1}(i). \end{aligned}$$

That is,

$$\hat{\delta}_{n+1}(i) = \delta_n(i) + w[\hat{g}_n(i) - \delta_n(i)], \tag{10}$$

where

$$\hat{g}_n(i) = \sum_{j \in I} P_{ij}(R_i)\delta_n(j). \tag{11}$$

$\hat{g}_n(i)$ will represent the value of $\delta_{n+1}(i)$ for the standard algorithm ($w = 1$), if there is no change in the decision for state i at the $(n+1)$-st iteration.

Performing a similar one-step look-ahead analysis for the semi-Markov case (10) is modified accordingly and becomes

$$\hat{\delta}_{n+1}(i) = \delta_n(i) + w[\tilde{g}_n(i) - \delta_n(i)], \tag{12}$$

where (11) is transformed into

$$\tilde{g}_n(i) = \sum_{j \in I} \tilde{P}_{ij}(R_i)\delta_n(j). \tag{13}$$

We shall use (10)–(13) to develop criteria for selecting 'good' ARFs in order to further speed-up the value iteration algorithm.

The ARF proposed in [3] is derived from the following equation:

$$\hat{\delta}_{n+1}(h) = \hat{\delta}_{n+1}(u) \tag{14}$$

where u and h are defined as the states achieving the minimum and maximum components of $\delta_n(\cdot)$, i.e.,

$$\delta_n(h) = \operatorname{Max}_{i \in I} \{ \delta_n(i) \} \quad \text{and} \quad \delta_n(u) = \operatorname{Min}_{i \in I} \{ \delta_n(i) \}. \tag{15}$$

Thus, from (10) and (14), one gets

$$\delta_n(h) + w[\hat{g}_n(h) - \delta_n(h)] = \delta_n(u) + w[\hat{g}_n(u) - \delta_n(u)],$$

or

$$w = \frac{\delta_n(h) - \delta_n(u)}{\delta_n(h) - \delta_n(u) + \hat{g}_n(u) - \hat{g}_n(h)}. \tag{16}$$

The above factor tries to reduce the gap between the values $\delta_{n+1}(h)$ and $\delta_{n+1}(u)$ and yields practically an accelerated convergence process of the value iteration algorithm. However, finding an ARF by considering only the states u and h (with the smallest and largest differences $\delta_n(i)$, respectively), neglects to take into account the influence of all *other* states and thus might not be effective in certain iterations.

The computational effort per iteration of the standard value iteration algorithm for fully-dense case is of order $A|I|^2$ (where A is the average number of actions per state). However, realistic large MDPs are usually sparse with an average, say, of $\bar{N} \ll |I|$ possible one-step state-to-state transitions, such that the effort per iteration is of order $\bar{N}A|I|$. The modified algorithm, which is aimed at reducing the number of iterations, achieves this goal at the expense of increasing the computational effort per iteration. In [3] this is done by increasing the effort per iteration by $2|I|$, which represents the computational effort of calculating the ARF w.

In this paper an additional reduction in the number of iterations is achieved at the expense of further increasing the computational effort of finding an improved ARF. This seems quite worthwhile in particular for cases where the number of states and actions is high, which is characteristic for large scale Markov decision models.

3. The minimum ratio criterion

Inequality (3) defines the stopping condition for the value iteration algorithm. If this condition has not been satisfied by iteration n, it seems plausible for the next iteration to find an ARF w^* that will *minimize* or at least *reduce* the term

$$R(w) = \frac{\pi_1(w)}{\pi_2(w)} \tag{17}$$

where $\pi_1(w) = \text{Max}_i\{\hat{\delta}_{n+1}(i)\}$ and $\pi_2(w) = \text{Min}_i\{\hat{\delta}_{n+1}(i)\}$. (Clearly, the numerator (denominator) in (17) estimates the highest (lowest) value of $\delta_{n+1}(i)$.) Note that this criterion is meaningful if we take w in the range where $\pi_2(w) > 0$.

For this purpose we define

$$\pi_1(w_1^*) = \min_w \{\pi_1(w)\} = \min_w \Big\{ \max_i \{\delta_n(i) + w\alpha_n(i)\} \Big\}, \tag{18}$$

where,

$$\alpha_n(i) = \hat{g}_n(i) - \delta_n(i). \tag{19}$$

(Observe that $\alpha_n(h) \le 0$ and $\alpha_n(u) \ge 0$.)

Similarly, let

$$\pi_2(w_2^*) = \max_w \{\pi_2(w)\} = \max_w \Big\{ \min_i \{\delta_n(i) + w\alpha_n(i)\} \Big\}. \tag{20}$$

Clearly, from (15) and (10), $\pi_1(0) = \delta_n(h)$, and $\pi_2(0) = \delta_n(u)$.

Observe that $\pi_1(w)$ is piecewise linear, and being the Max over a set of linear functions it is also convex (Figure 1), so its minimum w_1^* can be found by descending from one segment to the next. Similarly, $\pi_2(w)$ is piecewise linear and concave (Figure 1) and its maximum at w_2^* can be found by ascending from one segment to the next. Therefore, $R(w) = \pi_1(w)/\pi_2(w)$ has its minimum in between $w^- \equiv \text{Min}(w_1^*, w_2^*)$ and $w^+ \equiv \text{Max}(w_1^*, w_2^*)$.

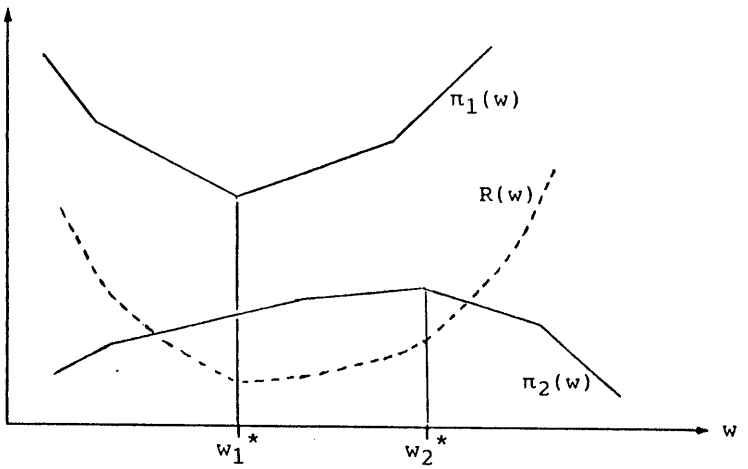

Fig. 1. Schematic illustration of $\pi_1(w)$, $\pi_2(w)$ and $R(w)$ when $w_1^* < w_2^*$

Moreover, $R(w)$ is piecewise *affine linear*, i.e., $R(w) = [A + Bw]/[C + Dw]$ and $R(w)$ is *monotone* on each segment:

$$\frac{\mathrm{d}}{\mathrm{d}w}\left[\frac{A + Bw}{C + Dw}\right] = \frac{BC - AD}{(C + Dw)^2}.$$

Therefore one is only interested in the *endpoints* of any segment. Furthermore, the endpoints of the segments correspond exactly with the breakpoints for $\pi_1(w)$ and $\pi_2(w)$.

Denote the breakpoints of $\pi_1(w)$ as $0 = x_0 < x_1 < x_2 < \cdots < x_M$, and let $A_m + B_m w$ be the line segment over the range $x_{m-1} \le w < x_m$ $(m = 1, 2, \ldots, M)$.

From the properties of $\pi_1(w)$ it follows that A_m is a decreasing sequence, while B_m is an increasing sequence. Similarly, if $0 = y_0 < y_1 < \cdots < y_N$ denote the breakpoints of $\pi_2(w)$, while $C_n + D_n w$ is the line segment over the range $y_{n-1} \le w < y_n$, then $C_n(D_n)$ is an increasing (decreasing) sequence. Let

$$X = \{x_m, 0 \le m \le M\}, \qquad Y = \{y_n, 0 \le n \le N\}$$

and

$$Z = X \cup Y = \{z_k, 0 \le k \le K\}.$$

Arrange the points of Z such that $0 = z_0 < z_1 < z_2 < \cdots < z_K$. Then the max of $R(w)$ must occur at one of the breakpoints $\{z_0, z_1, \ldots, z_K\}$. Hence, a reasonable heuristic is a one-pass scan along the z's until a change in sign is detected.

Another approach might be to try to retrieve *quickly* a 'good' starting point and then continue the search for w^*. Such a point could be either w_i^* which *minimizes* the function $\pi_1(w)$, or w_2^* which *maximizes* the function $\pi_2(w)$. However, there is a danger that by devoting too much effort to finding the optimal value w^* in every iteration we may lose on *overall* efficiency. It was found empirically that whenever $\delta_n(u) > 0$, the selection of a point w^* according to the *simple* rule

$$w^* = \begin{cases} w_1^* & \text{if } R(w_1^*) \le R(w_2^*), \\ w_2^* & \text{otherwise,} \end{cases} \tag{21}$$

usually yields either an optimal or near optimal relaxation factor for the Minimum Ratio criterion. Hence, it is desirable to develop an efficient procedure for finding w_1^* and w_2^*.

For this purpose consider again (18). This is a Minmax problem that can be formulated as a Linear Program:

Set $\text{Max}_i\{\hat{\delta}_{n+1}(i)\} \equiv \Theta$. Then the (Primal) problem is

$$\text{Min}\{\Theta\}$$

$$\text{subject to} \quad \Theta - w_1\alpha_n(i) \geq \delta_n(i), \quad i \in I,$$

$$w_1 \geq 0.$$

The corresponding Dual problem is

$$\text{Max}\left\{\sum_i \delta_n(i)\xi(i)\right\}$$

$$\text{subject to} \quad \sum_i \xi(i) = 1,$$

$$\sum_i \alpha_n(i)\xi(i) \geq 0,$$

$$\xi(i) \geq 0, \quad i \in I.$$

As there are only two constraints for the Dual, a basic feasible solution will have at most two positive $\xi(i)$'s. Furthermore, the optimal value of the objective function is obtained when the basis is composed of states j and i, say, such that $\alpha_n(j) \leq 0$ and $\alpha_n(i) > 0$. This follows since, if both $\alpha_n(j) < 0$ and $\alpha_n(i) < 0$, then the Dual problem is infeasible. On the other hand, if both $\alpha_n(j) > 0$ and $\alpha_n(i) > 0$, the value of the Dual objective function is no larger than $\text{Max}\{\delta_n(j), \delta_n(i)\}$, where a basis that represents either states j and h or states i and h yields a higher objective function value. (The intuitive explanation for the opposite signs of $\alpha_n(j)$ and $\alpha_n(i)$ is that the optimum w_1^* occurs where a downward-slopping line and an upward-slopping line cross, as depicted in Figure 1.)

We use this property in developing a 'greedy' procedure for finding the value of w_1^*. This procedure has the advantage that it *skips* some of the x_m values.

Step 0. Set $w_1^* = 0$, $\Omega = \delta_n(h)$, and $\beta = \alpha_n(h)$. (Clearly, $\delta_n(h) > 0$, $\alpha_n(h) \leq 0$.) If $\delta_n(h)$ is not unique select that state with the highest value of $\alpha_n(\cdot)$.

Step 1. Find

$$w_1 = \min_{i:\, \alpha_n(i) > 0} \left\{ \frac{\Omega - \delta_n(i)}{\alpha_n(i) - \beta} \right\}.$$

Step 2. Find

$$\Theta = \max_{i:\, \alpha_n(i) \leq 0} \{\delta_n(i) + w_1\alpha_n(i)\} = \delta_n(r) + w_1\alpha_n(r),$$

and set the new value of w_1^* to $w_1^* + w_1$. (Step 2 ensures feasibility. Together with Step 1, the procedure regenerates the initialization properties.)

Step 3. If $\Theta = \Omega + w_1\beta$ stop. w_1^* is the required optimal value. Otherwise, set $\Omega = \Theta$, $\beta = \alpha_n(r)$, and for all i update $\delta_n(i)$ to $\delta_n(i) + w_1\alpha_n(i)$. Go to Step 1.

A geometric illustration of the procedure is presented in Figure 2. $\pi_1(w)$ ($\pi_2(w)$) is depicted by the upper (lower) envelope. In this illustration the algorithm starts at $\delta_n(h)$, moves to B, then to C, and finally stops at D. B occurs at the earliest crossing with a positive-sloped line. The reinitialization move from B to C restores feasibility. The formulation of the Maxmin problem for finding the value w_2^* (see (20)) can be converted to an equivalent Minmax problems in the following way:

Set $\text{Min}_i\{\hat{\delta}_{n+1}(i)\} \equiv \phi$. Then the Linear Programming problem is

$$\text{Max}\{\phi\}$$

$$\text{subject to} \quad \phi - w_2\alpha_n(i) \leq \delta_n(i), \quad i \in I;$$

$$w_2 \geq 0.$$

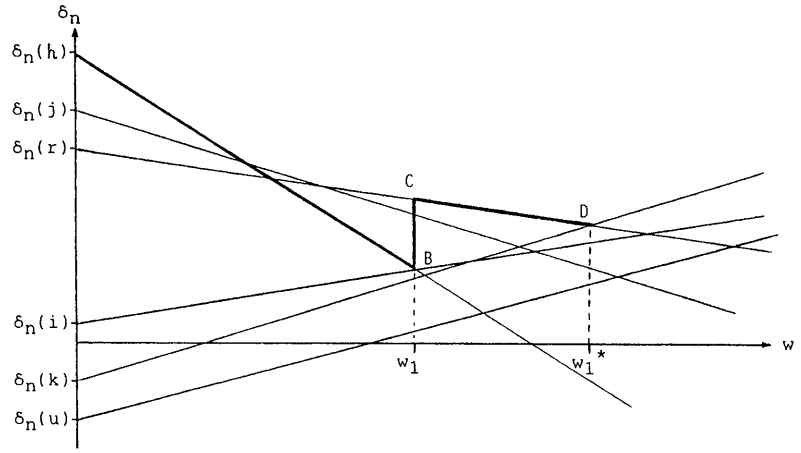

Fig. 2. A geometric illustration of finding the value w_1^*

This is equivalent to

$$-\text{Min}\{-\phi\}$$

$$\text{subject to} \quad -\phi + w_2\alpha_n(i) \geq -\delta_n(i),\ i \in I;$$

$$w_2 \geq 0.$$

Thus, by substituting the negative values of $\delta_n(i)$, $\alpha_n(i)$, $i \in I$, and starting at Step 0 with $\Omega = -\delta_n(u)$, $\beta = -\alpha_n(u)$, the algorithm of finding the relaxation factor w_2^* is identical to that of finding w_1^*. That is, the Minmax problem is a 'Mirror Reflection' of the Maxmin problem (Figure 2).

Before concluding this section we note that our procedure makes an implicit assumption that the same policy R_i recurs twice in a row. This is usually not expected to occur for small n while the iterative scheme is still hunting for good policies. Policy recurrence is almost certain to occur for moderate n, as evidenced by the observed usually-good performance of the ARF method. However, there is a rare chance – illustrated in the Federgruen and Schweitzer survey [1] – that in bizarre cases the policy can *fail to converge*, in which case the ARF might fail. But this has never been observed in a real problem.

4. The minimum variance criterion

In this section we consider another criterion for selecting the ARF so as to reduce the number of iterations of the algorithm. By this criterion we select the value w^* that minimizes the *variance* of the terms $\hat{\delta}_{n+1}(i)$, $i \in I$. This criterion differs from previous criteria by considering the *entire* set of $\hat{\delta}_{n+1}(i)$'s and not only its extreme components.

Consider the values $\delta_n = \{\delta_n(i),\ i \in I\}$ and $\alpha_n = \{\alpha_n(i),\ i \in I\}$. Then the vector $\hat{\Delta}_{n+1}(w) = \delta_n + w\alpha_n$ has components $\{\delta_n(i) + w\alpha_n(i)\}$. Clearly,

$$\text{Var}\left[\hat{\Delta}_{n+1}(w)\right] = \text{Var}[\delta_n] + w^2\text{Var}[\alpha_n] + 2w\text{Cov}[\delta_n, \alpha_n]. \tag{22}$$

Setting the derivative of $\text{Var}[\hat{\Delta}_{n+1}(w)]$ to zero, one gets

$$w^* = \frac{-\text{Cov}[\delta_n, \alpha_n]}{\text{Var}[\alpha_n]}. \tag{23}$$

Usually $\text{Cov}[\delta_n, \alpha_n] \leq 0$ and consequently $w^* \geq 0$ (see Section 5). This happens as $\alpha_n(i) = \Sigma_j P_{ij}(R_i)\delta_n(j) - \delta_n(i)$, and $\delta_n(i)$ increasing (decreasing) usually results in $\alpha_n(i)$ decreasing (increasing).

Since $d^2/dw^2\{\text{Var}[\hat{\Delta}_{n+1}(w)]\} = 2\text{Var}[\alpha_n] > 0$, the optimal value w^* minimizes indeed the variance of $\hat{\Delta}_{n+1}$.

In fact, w^* represents the value of the regression coefficient in the linear regression of δ_n on $(-\alpha_n)$, and is easily calculated using equation (24):

$$w^* = \frac{-\left\{\sum_i \delta_n(i)\alpha_n(i) - \left[\sum_i \delta_n(i)\right]\left[\sum_i \alpha_n(i)\right]/|I|\right\}}{\sum_i [\alpha_n(i)]^2 - \left[\sum_i \alpha_n(i)\right]^2/|I|}. \tag{24}$$

5. Computational considerations and numerical results

The effectiveness of the proposed criteria for selecting an ARF was tested on several problems. It was revealed that the Minimum Ratio criterion is sometimes accompanied by two types of 'congestion' or 'jamming', which represent occasions with limited sensitivity (see also Varaiya [8]). *Congestion-type-1* may occur in iterations when the change in $\pi_1(w)$ is only marginally sensitive to changes in w over the range where $\pi_1(w)$ is decreasing. That is, when $\pi_1(w^*)$ is close to the value of $\delta_n(h)$. Similarly, *congestion-type-2* is associated with the function $\pi_2(w)$. Congestion-type-1 may be detected by testing whether there exists a state i such that

$$|\delta_n(i) - \delta_n(h)| \le \varepsilon_1, \qquad \{|\alpha_n(i)| \le \varepsilon_2 \quad \text{or} \quad \alpha_n(i) > 0\},$$

where ε_1 and ε_2 are predetermined tolerance values. A similar test is applied for detecting congestion-type-2, using $\delta_n(u)$.

When both types of congestion occur at a certain iteration, the Minimum Variance criterion tends to be more effective than the Minimum Ratio criterion. Predetection of congestion cases has the potential of saving computational efforts and leads to an hybrid use of Minimum Ratio and Minimum Variance criteria as presented in the flow chart in Figure 3.

In turns out that the Minimum Variance approach becomes less effective for small values of w^*. Therefore, for practical considerations, it is recommended to predetermine a value $w_{\min}$, and use (24) if $w^* > w_{\min}$. (In our computations we used the value $w_{\min} = 0.3$.) It should also be noted that for both

Table 1
Number of iterations and CPU time units for various MDP and SMDP problems tested

Adaptive relaxation factor criteria	Prob. no. 1 $A=2$, $\|I\|=8$		Prob. no. 2 $A=4$, $\|I\|=10$		Prob. no. 3 $A=8$, $\|I\|=12$		Prob. no. 4 $A=16$, $\|I\|=15$	
	MDP	SMDP	MDP	SMDP	MDP	SMDP	MDP	SMDP
Regular value iteration algorithm ($w=1$)	25 14.4	27 17.2	50 63.5	53 74.8	52 161.2	77 250.2	64 507.3	86 948.7
Popyack, Brown and White [1]	19 11.5	17 11.2	38 50.7	40 57.8	61 193.2	35 118.5	not convergent	46 419.4
Minimum variance	10 11.1	15 18.1	21 47.5	30 70.2	27 131.4	29 141.3	28 266.8	38 445.3
Hybrid use of Mininum Ratio and Minimum Variance	15 20.8	8 12.1	27 62.6	23 58.2	31 150.1	22 110.9	36 376.1	31 373.6

MDP = Markov Decision Process; SMDP = Semi-MDP; A = number of actions per state; $|I|$ = number of states

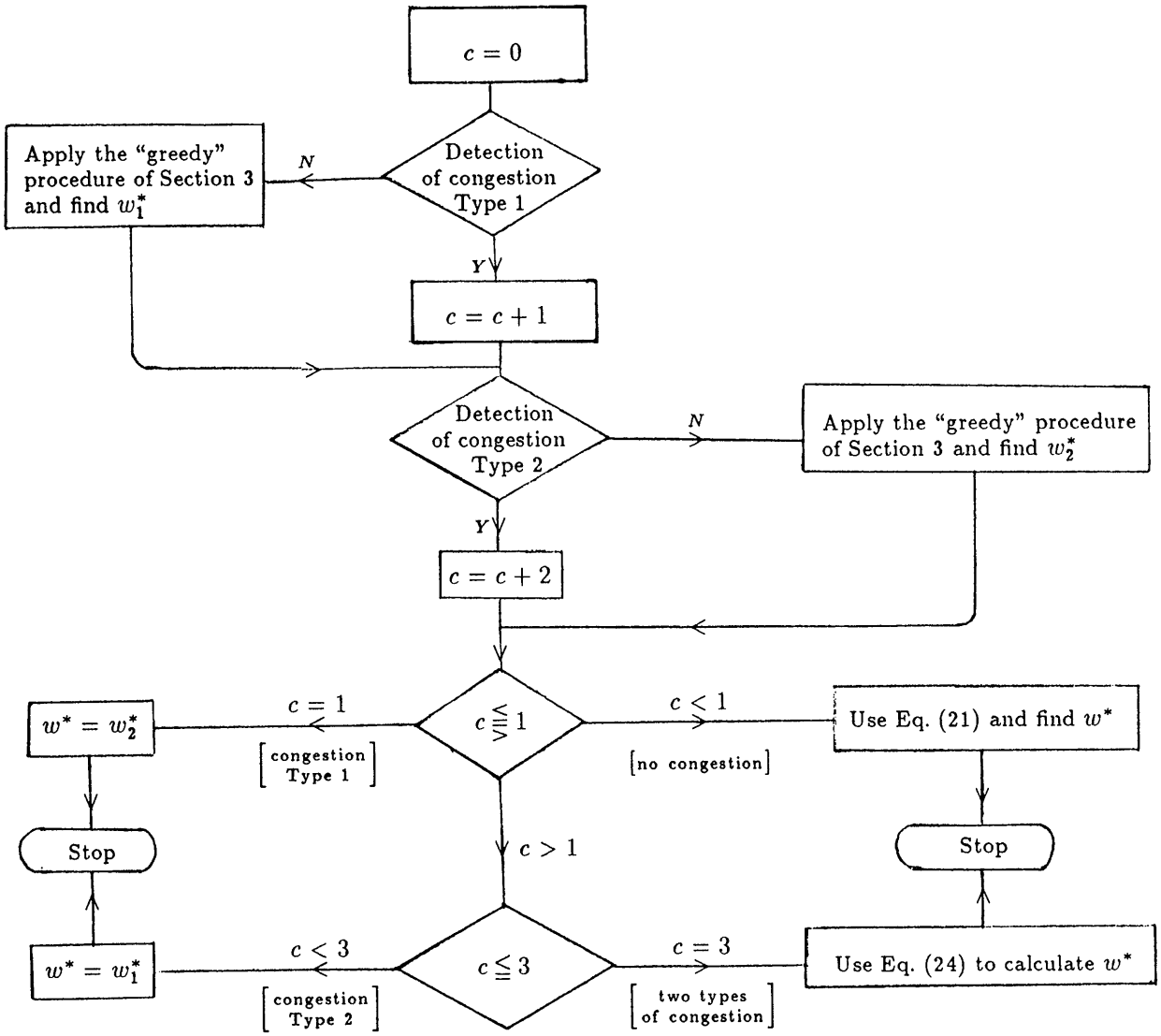

Fig. 3. Hybrid use of Minimum Ratio and Minimum Variance criteria

criteria the most time consuming factor in the process of finding w^* is the calculation of the $\{\hat{g}_n(i)\}$ values, which requires computational effort of order $O(|I|^2)$.

Several real problems dealing with optimal dimensioning of telecommunication networks were tested on an IBM XT personal computer. In Table 1 we present numerical results for 4 selected problems which differ from each other mainly by dimension. The tolerance error used for the problems tested was $\varepsilon = 10^{-3}$, with $2 \leq \bar{N} \leq 4$. As expected, the relative advantage of the proposed criteria rises with the increase of the dimension of the problems. The table suggests that for the kind of problems tested the Minimum Variance criterion is more effective than the other procedures for Markov decision models, while an hybrid use of Minimum Ratio and Minimum Variance criteria yields best results for semi-Markov decision models.

In order to investigate these findings, we consider again (10) and (12), which are related to the Markov and semi-Markov models, respectively.

Using the explicit expression for $\tilde{g}_n(i)$, as determined by (6) and (13), (12) becomes:

$$\hat{\delta}_{n+1}(i) = \delta_n(i) + w\left\{\left[\tau/\tau_i(R_i)\right]\sum_j P_{ij}(R_i)\delta_n(j) + \left[1 - \tau/\tau_i(R_i)\right]\delta_n(i) - \delta_n(i)\right\}$$

or

$$\hat{\delta}_{n+1}(i) = \delta_n(i) + w\left[\tau/\tau_i(R_i)\right]\left[\hat{g}_n(i) - \delta_n(i)\right] = \delta_n(i) + w\alpha_n(i)\tau/\tau_i(R_i). \tag{25}$$

Equation (25) demonstrates the role played by the mean sojourn times in the determination of w for the SMDP. Indeed, the slope of each linear function $\hat{\delta}_{n+1}(i)$ depends not only on $\alpha_n(i)$ but also on $\tau_i(R_i)$. With respect to the Markov models this slope is restrained by a state-dependent factor $0 < \tau/\tau_i(R_i) \leq 1$.

It should be pointed out that selection of different values of τ *does not* affect the convergence rate of the algorithm, as the adaptive coefficient $w\tau$, appearing in (25), does not vary with changes in τ.

Conclusion

In this paper we have proposed promising ARF methods which were tested on several problems and reduced the unrelaxed computation time and number of iterations by a factor of 2 or 3. A limited investigation of sensitivity to the selection criteria and to problem size has been carried out. It appears that slight increases in the work per iteration can significantly decrease the total number of iterations. Further research may be directed into choice of criteria for the over-relaxation factor, into achieving robustness in behavior, and into better understanding of sensitivity to problem size, structure and sparsity.

Acknowledgement

We would like to thank the referee for many helpful suggestions which led to improvements in the paper.

References

[1] A. Federgruen and P.J. Schweitzer, "A survey of asymptotic value-iteration for undiscounted Markovian decision processes", in: R. Hartley, L.C. Thomas and D.J. White, (eds.), *Recent Developments in Markov Decision Processes*, Academic Press, New York, 73–109 (1980).

[2] L. Platzman, "Improved conditions for convergence in undiscounted Markov renewal programming", *Oper.Res.* **25**, 529–533 (1977).

[3] J.L. Popyack, R.L. Brown and C.C. White III, "Discrete versions of an algorithm due to Varaiya", *IEEE Trans. Automat. Control* **24**, 503–504 (1979).

[4] P.J. Schweitzer, "Iterative solution of the functional equations of undiscounted Markov renewal programming", *J. Math. Anal. Appl.* **34**, 495–501 (1971).

[5] P.J. Schweitzer and A. Federgruen, "Geometric convergence of value iteration in multichain Markov decision problems", *Adv. in Appl. Probab.* **11**, 188–217 (1979).

[6] H.C. Tijms, *Stochastic Modelling and Analysis: A Computational Approach*, John Wiley and Sons, Chichester, UK (1986).

[7] H.C. Tijms and A.M. Eikeboom, "A simple technique in Markovian control with applications to resources allocation in communication networks", *Oper. Res. Lett.* **5**, 25–32 (1986).

[8] P. Varaiya, "Optimal and suboptimal stationary control for Markov chains", *IEEE Trans. Automat. Control* **23**, 388–394 (1978).