EINDHOVEN UNIVERSITY OF TECHNOLOGY

Department of Mathematics and Computing Science

Memorandum COSOR 83-15

Application of Markov Programming

by

J. Wessels

Eindhoven, The Netherlands

September 1983

APPLICATION OF MARKOV PROGRAMMING

by

J. Wessels, Eindhoven

Abstract

The main topic of the paper is the relation between modelling and numerical analysis for Markov decision processes. It is demonstrated that the relation is very close, since numerical possibilities strongly depend on the structure of the model. This is even true for straightforward numerical techniques, but also for numerical analysis based on aggregation and/or decomposition.

Examples amplify the arguments.

1. Introduction

Markov decision processes are interesting from a practical and from a theoretical point of view. Most attention has been paid to theoretical aspects. Besides more insight in the nature of the processes, the study of theoretical issues has lead to elegant numerical approaches, both for total reward and for average reward decision processes.

Practically, Markov decision processes are interesting, since the concept does not put strong conditions on the analytic structures of process and criterion. The price to be paid, however, is expressed by the difficulties in analyzing the process. Solutions never take the form of explicit formulae and only seldomly the optimal strategies have a simple structure. These features would not be too bad if optimal solutions could be found by simple standard algorithms. Regrettably, however, this is only the case for relatively small problems. And here we arrive at a new difficulty: problems with, say, 100 states may seem medium-sized, but are relatively small. Namely, as in optimal control problems with an analytic structure, the state is usually expressed as a multi-dimensional variable. However, because of the lack of analytic structure, analysis of Markov decision processes has to be executed state by state and even a coarse grid gives 10.000 states in a 4-dimensional state space. A similar argument can be given for the decision variables.

So, practical problems have a tendency to produce relatively large models and large models are difficult to handle. This is a short formulation of the problem to be considered in this paper.
The treatment is as follows. In Section 2 a short description will be given of the models to be considered together with some properties and notations. In Sections 3, 4 and 5 numerical approaches will be described with their possibilities to handle large problems and to exploit the specific structure of the problem at hand. In Section 6 the three approaches are considered with respect to their possible use for aggregation and decomposition.

As problem structure is one of the essential features for solvability and choice of method, it is not sensible to compare methods via randomly drawn models. A useful comparison can only be obtained by considering problems which are akin to reallife problems.

2. The models and some properties

A process is observed at discrete time instants $t = 0,1,2,\dots$ and the state the process is in at time t is denoted by the random variable I_t with values in $N = \{1,\dots,N\}$. After the observation of the state some action has to be chosen from a set $A = \{1,\dots,A\}$. The action chosen at time t is denoted by the random variable A_t. If the process is at time t in state $i \in N$ and if action $a \in A$ is chosen, then $P(i,j;a)$ denotes the probability to observe the state j at time $t + 1$. Moreover, a reward $r(i;a)$ is cashed.
The actions A_t are generated in cooperation by the process I_t and the decision rule or strategy of the decision maker. A strategy s is supposed to be a sequence $(s_0,s_1,\dots)$ of policies s_t, where s_t maps N into A; $s_t(i)$ gives the action at time t if state i is observed.
In this way only decision rules without much memory are introduced, however, well-known results show that usually more general strategies are not necessary (for a recent overview of this aspect, cf. van der Wal/Wessels [27]).
If a strategy uses only one policy repeatedly, then it is called a stationary strategy. The term policy is also used deliberately for stationary strategies.
For given strategy s and starting state i, the random process (I_t,A_t), $t = 0,1,\dots$, is determined stochastically. Probabilities and expectations with respect to this process are denoted by $\mathbb{P}_{si}$ and $\mathbb{E}_{si}$ respectively; if the index i is deleted, then the vector for subsequent starting states $i = 1,2,\dots,N$ is meant.

The most important criteria for choosing a strategy are expected total discounted rewards and average expected rewards. Formally, these criteria are represented respectively as

$$(2.1a)\qquad v(s;i) = \mathbb{E}_{si} \sum_{t=0}^{\infty} \beta^t r(I_t;A_t) \qquad (\beta = \text{discountfactor})$$

$$(2.1b)\qquad g(s;i) = \liminf_{T\to\infty} \frac{1}{T+1} \sum_{t=0}^{T} \mathbb{E}_{si}\, r(I_t;A_t)\ .$$

So, the problem becomes the computation of v^* and g^*, respectively, which satisfy

$$(2.2a)\qquad v^*(i) = \max_s v(s;i)$$

$$(2.2b)\qquad g^*(i) = \max_s g(s;i)$$

and to find strategies s' and s" satisfying

$$(2.3a)\qquad v^* = v(s') \qquad \text{or} \qquad v(s') \geq v^* - \varepsilon e$$

$$(2.3b)\qquad g^* = g(s'') \qquad \text{or} \qquad v(s'') \geq g^* - \varepsilon e$$

where $\varepsilon > 0$ and e the N-vector consisting of ones only. The basic properties for finding these optima are the so-called optimality equations, which are respectively

$$(2.4a)\qquad v^* = \max_{s_0} \{r(s_0) + \beta P(s_0)v^*\}$$

$$(2.4b)\qquad \begin{cases} g^* + w = \max_{s_0} \{r(s_0) + P(s_0)w\} \\ g^* = \max_{s_0} P(s_0)g^*\ . \end{cases}$$

Here $r(s_0)$, $P(s_0)$ are defined as vector and matrix satisfying

$$r(s_0)(i) = r(i;s_0(i)) \quad , \quad P(s_0)(i,j) = P(i,j;s_0(i)) \ .$$

If one tries to exploit these properties for finding v^* and g^* respectively, then a substantial reduction of (2.4b) can be obtained if one knows beforehand that g^* is a constant vector, namely in that case the second set of equations in (2.4b) can be deleted. Moreover, one extra condition on w can be added to make the system uniquely solvable.

If solutions v^* and g^*,w to (2.4a) and (2.4b) respectively would be known, then the maximizing policies in (2.4a) and (2.4b) would constitute optimal stationary strategies. It is particularly the latter property together with the equations themselves which provides ample opportunities to construct numerical algorithms.

3. Linear programming

The sets of equations (2.4a) and (2.4b) can be viewed as sets of parametrized linear equations in which the parameter needs some maximization operation. Therefore, it is appealing to try a linear programming approach. This becomes even more appealing if one knows the following results of Hordijk [9] and Hordijk/Kallenberg [10] respectively:

Lemma 1.

a. v^* is the smallest function (componentwize) satisfying

$$v \geq r(s_0) + \beta P(s_0)v \qquad \text{for all policies } s_0 .$$

b. g^* is the smallest function (componentwize) for which there is some function w such that

$$\begin{aligned} g + w &\geq r(s_0) + P(s_0)w \\ g &\geq P(s_0)g \qquad \text{for alle policies } s_0 . \end{aligned}$$

These properties lead directly to the following linear programs for the discounted and average reward criterion respectively.

a. $\min \frac{1}{N} \sum_i v(i)$

subject to

$$v(i) \geq r(i;a) + \beta \sum_j P(i,j;a)\, v(j) \qquad \text{for all } a \in A,\ i \in N .$$

b. $\min \frac{1}{N} \sum_i g(i)$

subject to

$$\begin{aligned} g(i) + w(i) &\geq r(i;a) + \sum_j P(i,j;a)\, w(j) \\ g(i) &\geq \sum_j P(i,j;a)\, g(j) \qquad \text{for all } a \in A,\ i \in N . \end{aligned}$$

In the latter program the w(i), $i \in N$, are also variables, although they don't appear in the objective function.

The duals to these linear programs are, respectively

a. $\max \sum_{i,a} r(i;a)\, x(i;a)$

subject to

$$\sum_a x(i;a) - \beta \sum_{j,a} x(j;a)\, P(j,i;a) = \frac{1}{N}$$

$$x(i;a) \geq 0 \qquad \text{for all } a \in A,\ i \in N .$$

b. $\max \sum_{i,a} r(i;a)\, x(i;a)$

subject to

$$\sum_a x(i;a) + \sum_a y(i;a) - \sum_{j,a} y(j;a)\, P(j,i;a) = \frac{1}{N}$$

$$\sum_a x(i;a) - \sum_{j,a} x(j;a)\, P(j,i;a) = 0$$

$$x(i;a),\ y(i;a) \geq 0 \qquad \text{for all } a \in A,\ i \in N .$$

If g^* is known to be a constant vector, then the primal and dual programs b simplify accordingly. The dual programs can also be developed directly when considering probabilistic properties of the decision process for given policies. In fact, in that way the linear programming approach for Markov decision processes has been developed originally (cf. [6], [28] for the nondiscounted case). A good overview of the linear programming approach can be obtained from the monograph of Kallenberg [14].

The dual forms are best suited for the simplex method, since they have N constraints and $N * A$ variables (i.e. in the simplified form for the average reward case). Nevertheless, for standard linear programming algorithms, these problems are already large for Markov decision problems of moderate size.
For instance, a problem with $N = 100$, $A = 30$ gives a 100×3000 linear programming problem and a problem with $N = 500$, $A = 2$ gives a 500×1000 linear programming problem.

So. practically, the use of linear programming packages is not a solution for relatively large-sized problems. The only hope is that linear programming methods can be found which exploit the specific structure of the linear programs generated by Markov decision problems. The policy iteration technique, which is treated in the next section, can be viewed upon as such an approach. The only other approach in this style which is known to the author is based on the fact that linear programs describing the optimization of the flow in a processing network can be analyzed much more efficiently by adapted algorithms than by the standard simplex method. For recent developments on such adapted simplex algorithms, see Glover/Klingman [7]. It is very simple to formulate Markov decision problems as (generalized) network flow problems, namely by introducing distinct nodes to represent the state before the decision and before the chance move. This causes an increase in the number of states which can be quite considerable, namely amounting to N nodes representing states before the decision and $N \times A$ nodes respresenting states before the chance move. In the linear programming formulation the corresponding increase can be observed, leading from N to $N(A+1)$ restrictions and from NA variables to

$N(N+2)A$ variables. This is the general transformation. In practical situations, however, quite often the structure of the original problem permits a special transformation leading to simpler networks.

For instance, if each allowed decision amounts to a state transition, then it suffices to have N nodes to represent states before the chance move. Hence in this case the network only has 2N nodes instead of $N(A+1)$. It is quite often possible to model decisions in such a way that they are represented by state transitions: as examples think of inventory problems (decisions should be new stock levels and not order sizes) and also maintenance and replacement problems. For a more precise account of this remodelling, see van der Wal/Wessels [26].

At this moment it is not yet known how the numerical effects of this approach can be, but it is worthwhile to be investigated and it is sure that the effects depend on the extra structure of the problems.

4. Policy iteration

In the preceding section, the approach known as policy iteration was said to be a linear programming approach exploiting the particular structure of linear programs emerging from Markov decision problems. Indeed, the policy iteration method can be viewed upon as a linear programming method in which each iteration step consists of N exchanges of basisvariables in the dual linear programs (for the discounted case and the simplified undiscounted case). We will not treat this view extensively, since it does not help much in increasing numerical efficiency. For details, however, see [28] or [14].

For obtaining an idea of the numerical possibilities, it is more inspiring to base the policy iteration method directly on the optimality equations (2.4a) and (2.4b).

The main idea of the policy iteration method (Howard [11]) is the following: Take a policy which might be a maximizer in (2.4a) (or (2.4b)), compute $v(s)$ (or $g(s)$, $w(s)$) for $s = (s_0, s_0, \ldots)$ and check whether this vector could be v^* (or g^*, w) and if not try to find a better policy than s_0 and start again. Worked out for the discounted case, this would lead to the following simple algorithm:

1. (policy evaluation)

   compute $v(s)$ from

$$v(s) = r(s) + \beta P(s_0)\, v(s) \qquad \text{with } s = (s_0, s_0, \ldots) \ .$$

2. (policy improvement)

$$\text{compute } \max_{s_0'} \; r(s_0') + \beta P(s_0')\, v(s) \quad .$$

   If s_0 itself is a maximizer, then $v^* = v(s)$. Otherwise s_0 is replaced by some maximizer and the procedure is repeated.

For the undiscounted version, the reader is referred to van der Wal [25], however, in the simplified case the algorithm is essentially similar and only looks more complicated.
Practical use of this method is hampered by the fact that the method requires solution of a $N * N$ set of linear equations in each iteration step, although, usually the number of iteration steps is quite small. For really large N this is quite cumbersome. Therefore, usually the method can only be used for values of N up to a couple of hundreds.

The amount of work can often be diminished by slight changes in both parts of the iteration step. For the first part (the policy evaluation) it seems a bit overdue to aim at exact or very precise solution of the equation set, estimates might do well. However, we will not treat these points here, since they fit quite naturally into the set-up of the next section where successive approximation methods are treated.
But there is a set of problems, which can be solved very efficiently by the policy iteration method. Namely, the special structure of the problem may allow a very efficient execution of the policy evaluation step.
For example, in several problems one knows beforehand that a so-called control-limit policy will be optimal. It is often possible to solve the policy evaluation equations very easily for control-limit policies. If it is possible to restrict attention to control-limit policies in the policy improvement step, this gives an extremely efficient algorithm. Well-known examples can be found in inventory-control problems, where (s,S)-policies are often optimal (cf. Johnson [13]). Other examples may be found in maintenance problems or in queue-control problems (cf. Johansen/Slidham [12] and van Nunen/Puterman [18]). Regrettably, however, there are no high-dimensional problems known which can be solved in this way. On the other hand, this method may be used for intermediate problems in an aggregation or decomposition approach (cf. Section 6).

5. Successive approximations

Linear programming and policy iteration seem to provide the more advanced approaches for the numerical analysis of Markov decision problems. The most direct and primitive approach, however, appears to give the best results in terms of efficiency until now (only in cases where the solution of the set of linear equations becomes trivial, the policy iteration approach wins, cf. the final remarks of Section 4).
However, the successive approximation approach may be simple and effective, that does not imply that application is straightforward. There are very many variants of the successive approximation method and quite often only some of these variants are efficient. Moreover, it does not suffice to consider generally applicable variants, but it is also necessary to exploit the problem structure.
In this paper we will not give a detailed account of all variants of the successive approximation method. There are already reviewpapers where this topic is treated from different points of view (see van Nunen/Wessels [20], [21] and Hendrikx/van Nunen/Wessels [8]).

As said above, the approach is rather primitive and consists of guessing a solution v_0 for (2.4a) and computing the maximization in the right-hand side of (2.4a) with v_0 instead of the unknown v^*; the result, v, say, is then considered to be a better guess and the procedure is repeated.

For the undiscounted case the same is done with the first set of equations in (2.4b). This gives a procedure which is practically equal to the discounted case, at least in the simplifying situation. In this way we obtain the following structure for the n-th iteration step:

1. Approximation of $v(s^n)$ by v_n where $s^n = (s_0^n, s_0^n, \ldots)$ is the stationary strategy which emerges from iteration step $n-1$.

2. $$\max_s L(s)\, v_n \quad , \quad \text{leading to } s_0^{n+1}$$

   where $L(s)$ is some operator on $\mathbb{R}^N$, for instance

   $$L(s)\, v_n = r(s_0) + \beta P(s_0)\, v_n \, .$$

3. Stopping criterion.

In the undiscounted case the v's have to be replaced by w's and β can be deleted. Not only the procedure is completely similar in the undiscounted case (particularly in the simplifying situation with guaranteed aperiodicity), but also the convergence properties are in practice the same. Namely, for theoretical convergence proofs a discountfactor $\beta < 1$ is of great help. However, numerically convergence would usually take a long time if it depended on the discounting. Both cases need the mixing properties of the underlying processes for numerical convergence in a reasonable time. We will come back to this aspect later in this section.

By putting the structure of the iteration step as we have done, it is easy to characterize the variants by their choice of approximation in Part 1, choice of operator $L(s)$ and choice of stopping criterion.
For instance, if one takes $v_n = v(s^n)$, then one obtains a policy iteration method. However, if one takes

$$v_n = \max_s L(s)\, v_{n-1} \, ,$$

then one obtains the more standard form of a successive approximation method.

For obtaining an idea of all the possible choices we refer to the review papers mentioned. However, it is worthwhile to note, that it is not necessary to repeat the same iteration step over and over again. As argued in [8], it might be attractive to alternate different variants in order to combine the advantages of these variants. For instance, the choice of L(s) based on the Gauss-Seidel idea quite often gives good convergence of the v_n to v^*; regrettably, however, the Gauss-Seidel idea does not give a good stopping criterion and, therefore, after a couple of Gauss-Seidel iterations a standard Jacobi iteration can be well in place (cf. [8] for more details).

The last remark more strongly holds for the undiscounted case, since there only the Jacobi iteration is known to give a proper stopping criterion. The standard successive approximation method for the undiscounted case becomes:

$$w_{n+1} = \max_{s_0} \{r(s_0) + P(s_0)\, w_n\}$$

with a stopping criterion provided by

$$\min_i \{w_{n+1}^{(i)} - w_n^{(i)}\}\, e \le g^* \le \max \{w_{n+1}^{(i)} - w_n^{(i)}\}\, e$$

where e is the N-vector of only ones. Note that g^* is constant in the simplifying situation and then, indeed, also $w_{n+1} - w_n$ converges to a constant vector (if, at least, aperiodicity is guaranteed).
So, in the undiscounted case, the operator can be replaced by a different one and also w_n may be some approximation of $w(s^n)$, but one needs a standard Jacobi step now and then in order to obtain a stopping criterion.

Also in the discounted case $v_{n+1} - v_n$ provides a basis for a stopping criterion and also here convergence stems from the fact that $v_{n+1} - v_n$ tends to become a constant vector. This is caused by the same mixing properties of the process as in the undiscounted case and, usually, only slightly accelerated by the discounting. In [23] one can find an example in which convergence has practically been completed after a number of iterations corresponding to a time-interval of two weeks, whereas the discounting with a realistic factor does not have a significant influence in such a time-interval. This effect is rather a usual than an occasional phenomenon.

The choice of a variant requires a lot of skill and experience, since the criteria remain vague. This is caused by the fact that several features are important and these features are influenced differently by one variant. Alternate use of different variants can give a partial answer to this question. The situation is further complicated by the fact that a specific structure of the problem can make some variants more easy to use. In [8] it is demonstrated that some sort of separability of the problem can be very helpful. If one has

$$P(i,j;a) = P(j;a)$$

then

$$\beta \sum_j P(i,j;a)\, v_n(j) = d_n(a)$$

independent of i. And if, moreover,

$$r(i;a) = r(i) + \rho(a)$$

then

$$\max_a \{r(i;a) + \beta \sum_j P(i,j;a)\, v_n(j)\} = r(i) + \max_a \{\rho(a) + d_n(a)\} =$$

$$= r(i) + \delta_n(a) .$$

In this way a very substantial simplification of the computations in the policy improvement part (Part 2) of the algorithm can be obtained. By proper choice of the model, one can often obtain this sort of separability (take the new inventory level as possible action instead of order quantity). This approach also works quite often in high-dimensional problems.

A lot more can be said about successive approximations, but we will stop here with the final observation that, practically, it is possible to solve problems with a couple of thousands states using the successive approximations. In specific cases we have experienced that also tens of thousands of states can be workable. So it is known (cf. [23]), that an extra dimension in the state space because of cyclically varying transition probabilities does not enlarge the numerical complexity.

6. Aggregation and decomposition

A conclusion from the three preceding sections may be that the numerical treatment of relatively small Markov decision problems can easily be undertaken by any of the three main approaches as sketched. For problems with an intermediate size, the successive approximatio approach often indicates the road to be taken, with policy iteration as a better choice in particular cases. For really large problems, however, the three approaches don't give the right cure. Although in very specific cases the successive approximation approach can still be used. Regrettably, large problems are the rule rather than the exception and therefore a numerical approach for large problems is necessary.

Even for problems with an intermediate size, the exploitation of the particular problem structure is a necessity in order to obtain efficient procedures. Therefore, one cannot hope that for large problems one efficient standard method will present itself.

It is a wide-spread belief that for really large Markov decision problems aggregation/disaggregation and decomposition/composition approaches point out the most promising ways for the analysis. Although there is quite some literature on aggregation in Markov decision processes (see, for instance, Whitt [29]), not much attention has been paid in the literature to numerical methods based on aggregation or decomposition.
In this section a rough review will be given of attempts to exploit aggregation or decomposition for numerical purposes. Therefore, we first consider the linear programming approach and afterwards aggregation or decomposition of the problem directly.

6.1. The linear programming approach

Decomposition and aggregation both have been worked out for linear programming problems. The oldest and most well-known are the decomposition/composition approaches of Dantzig/Wolfe [3] and of Benders [2]. The author does not know of any successful attempt to apply these approaches for the linear programs generated by Markov decision problems. The main difficulty in such an application would be that the linear programs themselves (dual or primal) usually do not show much of the structure desirable for one of the decomposition approaches. Nevertheless, for a given feasible basis the part of the matrix corresponding to such a basis may have a better structure. Perhaps this feature might be used in some way to develop a decomposition/composition approach for the particular linear programs arising for Markov decision processes. In recent years also an aggregation/disaggregation approach has been worked out for linear programming problems (cf. Vakkutinskii/Dudkin/Ryvkin [22], Zipkin [30]). This approach is iterative in the sense that it starts with the solution of a highly aggregated problem and that this solution helps in defining a less aggregated problem etc. This approach has been specialized to the linear programs arising from Markov decision problems by Mendelssohn [17]. By this approach the linear programming approach may be made somewhat more efficient, however, it is not to be believed that this approach will make linear programming the right method for large problems, unless there would be some device which would make it possible to stop the iteration process relatively soon.

6.2. Aggregation or decomposition of the model

It seems to be much more sensible to consider aggregation or decomposition within the context of the model itself. Numerical approaches can then be chosen later. A simple example shows already that the choice of decomposition or aggregation does not only depend on the structure of the model, but also on the actual numbers. Consider, for instance, some simple one-machine/ multi-product production planning problem with random demands. The natural decomposition approach will lead to dealing with the individual products first and later try to solve capacity problems. In the natural aggregation approach one will aggregate the demand for all products and treat the capacity part first. It will be apparent that an aggregation approach will work best if capacity is a real bottleneck, otherwise decomposition is the best choice (cf. Bemelmans [1] for this sort of comparison and van Nunen/Wessels [19] for a worked out example or this type of decomposition).
Particularly in the area of multi-product and/or multi-stock production and inventory problems there have been several attempts recently using model aggregation or decomposition (cf. Federgruen/Zipkin [5]).

All attempts in this direction are heuristic. Usually, these attempts lead to rather efficient methods. However, there are no proofs of convergence or estimates of deviations from the optimal solution.
Even for the simplest type of aggregation/disaggregation approach, namely via aggregation in the action space only, there is no efficient estimate of deviations. However, particularly in that case it will be possible to find good estimates if transition probabilities and costs have a simple structure (e.g. inventory type models). This type of aggregation/disaggregation has been de-

scribed in Veugen/van der Wal/Wessels [24]. In that paper it is also shown that usually aggregation in the action space is not necessary (see also Section 5). However, if old decisions have to be retained in the state description, then it is shown that this type of aggregation-disaggregation can be worthwhile.

Finally, we want to mention a hybrid approach using a highly aggregate decision model. This decision model is analyzed by one of the techniques of Sections 3 - 5. The detailed process is then simulated with the strategy resulting from the optimization. The simulation results are used to redefine the aggregate decision model which is analyzed anew, etc. (cf. Lenssen/van der Wal/Wessels [15], Demandt/van der Wal [4]). Although, again, a good theory for this type of approach is lacking, the experiences are quite promising.

References

[1] R. Bemelmans, Aggregation and decomposition in one-machine, multi-product planning problems.
Operations Research Proceedings 1982, Springer-Verlag, Berlin 1983, p. 29 - 38.

[2] J.F. Benders, Partitioning procedures for solving mixed-variables programming problems.
Numer. Math. 4 (1962) 238 - 252.

[3] G.B. Dantzig, Ph. Wolfe, Decomposition principle for linear programs.
Oper. Res. 8 (1960) 101 - 111.

[4] H.M. Demandt, J. van der Wal, Manpower planning in a general purpose shipterminal - an iterative aggregation/disaggregation approach.
This volume.

[5] A. Federgruen, P. Zipkin, Several recent reports on this topic have been written by these authors in the series of Research Working Papers, Graduate School of Business, Columbia University, New York.

[6] G.T. de Ghellinck, G.D. Eppen, Linear programming solutions for separable Markovian decision problems.
Management Science 13 (1967) 371 - 394.

[7] F. Glover, D. Klingman, The simplex SON algorithm for LP/embedded network problems.
Mathematical Programming Study 15 (1981) 148 - 176.

[8] M.H.M. Hendrikx, J.A.E.E. van Nunen, J. Wessels, Some notes on iterative optimization of structured Markov decision processes with discounted rewards.
Mathematische Operationsforschung und Statistik, Ser. Optimization (to appear).

[9] A. Hordijk, Dynamic programming and Markov potential theory.
Mathematical Centre Tract 51, Amsterdam 1974.

[10] A. Hordijk, L.C.M. Kallenberg, Linear programming and Markov decision chains.
Management Science 25 (1979) 352 - 362.

[11] R.A. Howard, Dynamic programming and Markov processes.
John Wiley, New York 1960.

[12] S.G. Johansen, S. Stidham Jr., Control of arrivals to a stochastic input-output system.
Adv. Appl. Probab. 12 (1980) 972 - 999.

[13] E.L. Johnson, On (s,S) policies.
Manag. Sci. 15 (1968) 80 - 101.

[14] L.C.M. Kallenberg, Linear programming and finite Markovian control problems.
Mathematical Centre Tract 148, Amsterdam 1983.

[15] M.J.G. Lenssen, J. van der Wal, J. Wessels, Markov decision processes and ship handling: an exercise in aggregation.
Operations Research Proceedings 1982. Springer, Berlin 1983, p. 476 - 482.

[16] A.S. Manne, Linear programming and sequential decisions.
Management Science 6 (1960) 259 - 267.

[17] R. Mendelssohn, An iterative aggregation process for Markov decision processes.
Administrative report no. H - 80 - 3, National Oceanic and Atmospheric Administration, Honolulu 1980.

[18] J.A.E.E. van Nunen, M. Puterman, On computing optimal policies for G/M/s-queueing systems.
Management Sci. 29 (1983) 725 - 734.

[19] J.A.E.E. van Nunen, J. Wessels, Multi-item lot size determination and scheduling under capacity constraints 2 (1978) 36 - 41.

[20] J.A.E.E. van Nunen, J. Wessels, Successive approximations for Markov decision processes and Markov games with unbounded rewards.
Mathematische Operationsforschung und Statistik, Ser. Optimization 10 (1979) 431 - 455.

[21] J.A.E.E. van Nunen, J. Wessels, On theory and algorithms for Markov decision problems with the total reward criterion.
OR Spektrum 1 (1979) 57 - 67.

[22] I.Ya. Vakkutinskii, L.M. Dudkin, A.A. Ryvkin, Iterative Aggregation - a new approach to the solution of large-scale problems.
Econometrika 47 (1979) 821 - 841.

[23] L.M.M. Veugen, J. van der Wal, J. Wessels, The numerical exploitation of periodicity in Markov decision processes.
OR Spektrum 5 (1983) 97 - 103.

[24] L.M.M. Veugen, J. van der Wal, J. Wessels, Aggregation and disaggregation in Markov decision models for inventory control.
Europ. J. Oper. Res. (to appear).

[25] J. van der Wal, Stochastic dynamic programming.
Mathematical Centre Tract 139, Amsterdam 1981.

[26] J. van der Wal, J. Wessels, De formulering van een Markov beslissingsprobleem als proces netwerk (in Dutch).
Kwantitatieve Methoden 4 (1983) 99 - 108.

[27] J. van der Wal, J. Wessels, On the use of information in Markov decision processes.
Statistics and Decisions (to appear).

[28] J. Wessels, J.A.E.E. van Nunen, Discounted semi-Markov decision processes: linear programming and policy iteration.
Statistica Neerlandica 29 (1975) 1 - 7.

[29] W. Whitt, Approximations of dynamic programs I, II.
Math. Oper. Res. 3 (1978) 231 - 243, 4 (1979) 179 - 185.

[30] P. Zipkin, Bounds for row-aggregation in linear programming.
Oper. Res. 28 (1980) 903 - 916.