On Markov games

Citation for published version (APA):

Document status and date:
Published: 01/01/1975

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher’s website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:
www.tue.nl/taverne

Take down policy
If you believe that this document breaches copyright please contact us at:
openaccess@tue.nl
providing details and we will investigate your claim.
Memorandum COSOR 75-12

On Markov games

by

J. van der Wal and J. Wessels

Eindhoven, August 1975

The Netherlands
On Markov games

by

J. van der Wal and J. Wessels

Abstract: In the paper it is demonstrated, how a dynamic programming approach may be useful for the analysis of Markov games. Markov games with finitely many stages are dealt with extensively. The existence of optimal Markov strategies is proven for finite stage Markov games using a shortcut of a proof by Derman for the analogous result for Markov decision processes. For Markov games with a countably infinite number of stages some results are summarized. Here again the results and the methods of prove have much in common with results and proofs for Markov decision processes. Actually the theory of Markov games is a generalisation. The paper contains short introductions into the theories of matrix games and tree games.

1. Introduction: The theory of games is a part of mathematics. Its aim is the analysis of mathematical models for decision situations with at least two decision makers having more or less opposite goals. Examples of such situations are card games and board games. Other applications may be found in economic theory, military science, negotiation problems.

Many real-life situations with more than one decision maker involved (we will call them players henceforth), are typically dynamic in the following sense: the players have the possibility in subsequent moves to react on what happened so far (e.g. chess, bridge, bargaining, tankduels, economic competition).

An important tool in the theory of games is the replacement of a dynamic game by an equivalent one-stage game. For theoretical purposes such a replacement is very convenient: proofs of the existence of good strategies and values for one-stage games are mathematically extremely elegant (the notions mentioned will be defined in section 2; for a more elaborate treatment we refer to the epochmaking book on game theory by Von Neumann and Morgenstern [9], or the introduction to game theory by Owen [11]).

However, such a replacement is not very helpful if one is interested in the actual computation of values and good strategies and in the structure of
good strategies. For these reasons this paper emphasizes mainly on the use of the dynamic structure for the analysis of games.

In dynamic programming and in the strongly related theory of Markov decision processes (as initiated by Bellman in [1] and by Howard in [5], we encounter situations with one decisionmaker, in which the dynamic structure plays a basic role. In these situations it appeared to be possible to use the dynamic structure essentially for the analysis. So the question arises, which part of the tools developed in Markov programming may be extended for the use in situations with more than one player.

In the literature the connection between dynamic games and dynamic programming is not investigated systematically, although similar ideas appear in both fields. For example, Shapley's paper on Markov games has been published as early as 1953. However, it is not mentioned in papers on Markov programming, which have been published several years later, in which similar ideas are investigated in an essentially simpler situation. Even Von Neumann and Morgenstern already use a dynamic programming type of argument in a sketch of a proof [9, §15.8]. Namely for the assertion that in two-person games (with perfect information of what happened before and with the gain of one player equal to the loss of the other) only pure moves (see section 2) need to be considered. For a somewhat more formal proof using the same type of argument we refer to the book of Blackwell and Girshick [2, §1.7].

In this paper we will consider multi-stage games with two players, each having a finite number of possible moves at any stage of the game. Our games may be influenced by random events (like the tossing of coins and random distributing of cards). Furthermore it will be assumed that the gain of the first player has to be paid by the second one (in technical terms: we consider multi-stage two-person zero-sum games). All these assumptions may be weakened in some sense. However, in order to avoid distracting technical details, we will stick to the assumptions.

In section 2 some well-known facts about one-stage games (or matrix games) will be summarized. In section 3 games with a finite number of stages will be introduced using the concept of a treegame. The usefulness of a dynamic programming approach will be demonstrated for the perfect information case. In section 4 we introduce Markov games with a finite number of stages. This concept is slightly more general than the concept of a treegame with perfect
information. The analysis is completely based on dynamic programming ideas. The existence of good Markov strategies (for definition see section 4) is proven using a shortcut of a proof by Derman [4] for the analogous result for Markov decision processes. Section 5 is devoted to comments and extensions.

In section 6, Markov games with a countably infinite number of stages are considered. For the analysis of such games the analogy with Markov decision processes is extremely rewarding. Some results will be sketched, which have been obtained using some ideas about successive approximation procedures for Markov decision processes.

For an annotated bibliography on Markov games, we refer to [16].

2. Matrixgames: In this section we consider a very simple type of game. It is supposed, that both players (we call them P₁ and P₂) may select an action (or move) from a finite set of available actions: K in the set for P₁ and L for P₂. They are supposed to choose simultaneously without knowing what the competitor does.

The reward or gain for P₁ is equal to \( r(k, \ell) \) (possibly negative), if he chooses \( k \in K \) and P₂ selects \( \ell \in L \). The gain of P₁ is the loss of P₂.

Hence this type of game is completely characterized by the reward function \( r(k, \ell) \), which may be denoted in matrixform.

For a detailed analysis of two-person zero-sum matrixgames we may refer to any introductory book on game theory (e.g. [2,9,11]).

Example: \( K = L = \{1,2,3\} \), \( r(k, \ell) = |k - \ell| \).

Hence this game is characterized by the following gainmatrix:

\[
\begin{pmatrix}
0 & 1 & 2 \\
1 & 0 & 1 \\
2 & 1 & 0
\end{pmatrix}
\]

We start the analysis of matrixgames by defining:

\[
\Lambda(k) := \min_{\ell \in L} r(k, \ell) , \quad \Omega(\ell) := \max_{k \in K} r(k, \ell)
\]

Here \( \Lambda(k) \) is the minimal reward for P₁, if he plays \( k \), whereas \( \Omega(\ell) \) is the maximal loss for P₂, if he plays \( \ell \).
Hence, by a sensible choice of $k$, $P_1$ may increase his guaranteed reward to

$$\lambda := \max_{k \in K} \Lambda(k),$$

whereas $P_2$ may decrease his most pessimistic loss to

$$w := \min_{k \in L} \Omega(k).$$

The interpretations of $\lambda$ and $w$ immediately imply: $\lambda \leq w$.

In the example $\Lambda(k) = 0$ for $k = 1, 2, 3$; $\Omega(k) = 2, 1, 2$ for $k = 1, 2, 3$ respectively. Hence $\lambda = 0$, $w = 1$.

A modification of the reward matrix such that $r(k, \ell)$ gets the value $k - \ell$ gives an example with $\lambda = n(= 0)$.

If $\lambda = w$, it is clear that, when both players attempt to maximize their most pessimistic reward, actions which maximize $\Lambda(k)$ and minimize $\Omega(k)$, respectively, are the best choices. In this case $\lambda$ is called the value of the game and the actions maximizing $\Lambda(k)$ and minimizing $\Omega(k)$, respectively, are called good pure strategies (the meaning of the adjective pure will become clear in the sequel).

If, on the other hand, $\lambda < w$, then it is not immediately clear which are the most sensible actions. We might again call actions which guarantee $P_1$ the reward $\lambda$ and $P_2$ the reward $-w$, respectively, good strategies, however, what happens with the still undivided amount $w - \lambda$?

This question is usually tackled by extending the game in such a sense, that so called mixed (or randomized) strategies are introduced: the players do not select an action from $K$ or $L$ respectively, but they select a probability distribution on $K$ or $L$ respectively. The interpretation of these strategies is, that the player selects a probability distribution and that a random experiment, according to the probability distribution selected, points out an action. The degenerated probability distributions coincide with the strategies in the non-extended game, which are called pure strategies henceforth.

If we introduce the expected reward as gain function, we obtain the following extended game:

The action sets for $P_1$ and $P_2$ are the sets $P$ and $Q$ of all probability distributions on $K$ and $L$ respectively. A typical element $p \in P$ is a set of non-negative numbers $p_k$ for $k \in K$, adding to one (analogously for $q \in Q$).

The gain function $R(p, q)$ is defined by
A basic theorem of game theory states, that for this new game—which possesses infinite action sets—the \(\lambda\)-value and \(\omega\)-value are equal:

\[
\begin{align*}
\lambda^* &= \max_{p} \min_{q} R(p,q) = \min_{q} \max_{p} R(p,q) =: \omega^*.
\end{align*}
\]

(One may easily verify that \(P\) and \(Q\) are compact subsets of \(\mathbb{R}^N\), for some \(N\), so that since \(R(p,q)\) is continuous we may write maxmin and minmax)

\(\lambda^*\) is called the (mixed) value of the original game. If \(\lambda = \omega\), then \(\lambda = \lambda^*\). The maximizing and minimizing strategies are called good (mixed) strategies. For proofs see the references [2, 9, 11] e.g.

Clearly a pair of good (mixed) strategies \((p^*, q^*)\) forms a saddlepoint of the function \(R(p,q)\):

\[
R(p,q^*) \leq R(p^*, q^*) \leq R(p^*, q^*) \text{ for all } p \in P, \ q \in Q.
\]

On the other hand one easily verifies, that if \((p^*, q^*)\) is a saddlepoint of \(R(p,q)\), then \(p^*\) and \(q^*\) are good (mixed) strategies and \(R(p^*, q^*)\) is the value of the game.

**Example:** If \(K = L = \{1, 2, 3\}\), we get:

\[
P = \{(p_1, p_2, p_3) \mid p_k \geq 0, \ p_1 + p_2 + p_3 = 1\} = \mathcal{Q}.
\]

In the matrix game with \(r(k, \xi) = |k-\xi|\) we find, as might be expected, that the strategies \(p^* = (\frac{1}{4}, 0, \frac{3}{4})\) and \(q^* = (\alpha, 1 - 2\alpha, \alpha)\), \(0 \leq \alpha \leq \frac{1}{4}\), are good for \(P_1\) and \(P_2\), respectively. (Notice that when \(r(3,3)\) is enlarged the saddlepoint becomes unique with \(\alpha = 0\))

Finding saddlepoints in a heuristic way is not always as easy as in the example. However, linear programming presents a systematic procedure, as indicated below.

For fixed \(p \in P\) we have obviously:

\[
\min_{q} \sum_{k, \xi} r(k, \xi) p_k q_\xi = \min_{\xi} \sum_{k} r(k, \xi) p_k.
\]

Hence the value \(\lambda^*\) of the game should satisfy for certain \(p\) and all \(\xi\):

\[
\sum_{k} r(k, \xi) p_k \geq \lambda^*.
\]
This implies, that the value of the game and a good strategy for $P_1$ may be obtained by solving the following linear programming problem in the variables $v$ and $p_k$ ($k \in K$):

$$\begin{align*}
\text{max } v \\
\text{subject to } & \sum_k p_k = 1 \\
& p_k \geq 0 \quad (k \in K), \\
& \sum_{k} r(k,\ell)p_k \geq v \quad (\ell \in L).
\end{align*}$$

In an optimal solution the $v$-value is equal to $\lambda^*$ and the $p$-part denotes a good strategy for $P_1$.

Note that the variable $v$ is not restricted to nonnegative numbers.

By computing $\min \max R(p,q)$ analogously, one finds the dual linear programming problem, which produces a good strategy for $P_2$. In this way the theory of matrix games is strongly related to (and in a sense even equivalent to) the duality theory of linear programming (see e.g. Dantzig [3]).

Example: In the example of this section we get the linear programming problem:

$$\begin{align*}
\text{max } v, \text{ subject to } & p_1 \geq 0, p_2 \geq 0, p_3 \geq 0 \\
& p_1 + p_2 + p_3 = 1 \\
& p_2 + 2p_3 - v \geq 0 \\
& p_1 + p_3 - v \geq 0 \\
& 2p_1 + p_2 - v \geq 0.
\end{align*}$$

Summarizing, the main assertion of this section, which will be used extensively in sections 4 - 6, is the following:

For any matrix game a value and good (mixed) strategies exist and may be computed (e.g. by linear programming).

3. Treegames: In many gaming situations (like card and board games) the players may subsequently choose a move. In this section, we will present a mathematical model for such games.
Firstly, we give a verbal description of the type of gaming situation this section treats:

1. for the start of the game, the rules specify whether one of the players makes the first move, or the first move is a chance move (like the throwing of a die, or the random distributing of cards). In the first case the rules specify which player makes the first move and which actions are available to him (e.g. in chess all allowed first moves for white, together with giving up and offering a draw). In the second case the rules specify the probability distribution of the outcome (e.g. the outcome of an honest throw with an honest die).

2. subsequent moves are specified by the rules of the game in a similar fashion. Viz., for the k-th move the same kind of specification is given as for the first one, however, in such a way that all features may depend on the realizations of the first k-1 moves (as in most gaming situations).

3. the rules should specify what information a player receives about what happened before (e.g. in board games all relevant information is always available, however, in many card games the players know exactly what has been done before by the players, but they don't know the complete outcome of the chance move the game started with viz. distributing cards).

4. the rules should specify when the game ends and which rewards are attached to any possible course of the game.

For the case of a finite two-person zero-sum game we will give a model using the concept of a tree. For a more detailed description for more general types of games the reader is referred to a paper by Kuhn [6].

A tree is a finite directed (connected) graph (or network), such that exactly one vertex does not have incoming branches and all the other vertices have exactly one incoming branch.

A treegame may be represented by a tree in which the set of vertices with outgoing branches is partitioned into three subsets \( V_0, V_1 \) and \( V_2 \). The sets \( V_1 \) and \( V_2 \) being again divided into sub-subsets consisting of vertices with an equal number of outgoing branches. And these sub-subsets are partitioned again into sub-sub-sub-subsets called information sets. To each vertex in \( V_0 \) a probability distribution on the outgoing branches has been attached. To each vertex without outgoing branches a number has been attached.
With this formal model games of the verbally described type may be handled, as appears from the interpretation, which will be given using the example of figure 1.

Fig. 1 points in \( V_0 \) are denoted by \( \square \), in \( V_1 \) by \( \square \), in \( V_2 \) by \( \square \), endpoints by \( \bullet \); information sets have been encircled. All branches are directed upwards.

The game is supposed to start in the bottom vertex, which is the only vertex without incoming branch. This vertex is an element of \( V_0 \), which means that the first move is a chance move. Suppose the outgoing branches are allotted with equal probabilities. If, say, 2 is allotted, then the gamesituation proceeds to the endpoint of the second branch, which is a vertex in \( V_1 \). Hence the second move is for \( P_1 \). Since this vertex and its left side neighbour belong to the same information set, \( P_1 \) does not know whether the chance move resulted in 1 or in 2. If \( P_1 \) chooses 1 (the outgoing branches denote his allowed actions), the third move is for \( P_2 \), since the outgoing branch 1 ends in a vertex of \( V_2 \). The information for \( P_2 \) is such that he does know that \( P_1 \) chose 1, but he does not know more about the chance move than \( P_1 \). His own possibilities are 1 and 2. If he chooses 2, he receives 5. On the other hand, if he had been in the other point of the information set - which he does not know - he would have to pay 4 with the same choice.

In this way, the playing of a game amounts to the successive determination of steps in a path leading from the bottom to one of the top vertices of a tree.
A treegame may be modelled as a matrix game by introducing the concept of a decision rule. For example in the game of figure 1, player \( P_1 \) has 4 decision rules, namely:

- choose \( i \) when the chance move is at most 2, choose \( j \) otherwise (where \( i \) and \( j \) may be 1 or 2).

In a similar way the \( 2^6 = 64 \) decision rules for \( P_2 \) may be given and for each pair of decision rules for \( P_1 \) and \( P_2 \) respectively the expected reward for \( P_1 \) may be computed. This generates a \( 4 \times 64 \) matrixgame.

How about finding sensible strategies in a treegame? Consider, for instance, the example of figure 1. Suppose that the game proceeded until the third move along the path on the extreme left. Then \( P_2 \) knows exactly what the situation is and which consequences his possible choices have: 1 will cost him 4 and 2 will bring him 5. So his choice won't be difficult. It is equally easy for him in the other vertices of \( V_2 \) which are alone in an information set.

It is much more difficult for \( P_2 \), if he knows only that he is in the second or third vertex from the left (on the third row). Then he should wonder, what will \( P_1 \) have thought on the second level, whereas \( P_1 \) should have asked himself how will \( P_2 \) react. This kind of interference, caused by the lack of information, makes the solution more difficult.

By the way, some actions for \( P_2 \) may be neglected because of the foregoing argument, while the game may be split up in two games (representing the right and the left half of the tree). Using these tricks the solution of the game only requires the solution of two \( 2 \times 2 \) - matrixgames instead of one \( 4 \times 64 \) - matrixgame. This is left as an exercise to the reader.

Let's modify the example in such a way, that both players are completely informed of the outcome of the chance move at the beginning of the game: then all information sets become one element sets. Such games are called: games with perfect information.

Now all sensible decisions for \( P_2 \) can easily be found: 2, 1, 2, 1, 1, 2, 1, 2 (from left to right). Knowing this we may attach the corresponding rewards to the vertices in the third row and skip all vertices and branches above them. Now the same argument gives for \( P_1 : 1, 1 \) or 2, 1, 1 or 2 (from left to right) as most sensible decisions. In this way we will surely find the best possible decisions by essentially using the dynamic
programming idea. The only difference being that in some steps one maximizes \( P_1 \) and in others one minimizes \( P_2 \). With perfect information the game is very unfavourable for \( P_1 \), the expected (optimal) reward being \( 1/-3-5-5-7 = -5 \) (how about the original game?).

So in perfect information treegames a dynamic programming approach may be very efficient. Notice that we don't need the mixing concept in perfect information treegames.

Although the strategies we found by dynamic programming are obviously the best possible ones, we did not prove that they are good in the sense of section 2. We will not prove this now, since this will be done in the next section for a somewhat larger class of dynamic games, viz. allowing some sort of imperfect information.

4. Finite-stage Markov games: In this section we will consider a special type of the dynamic games we introduced in the preceding section. However, the type we will consider is more general than the perfect information treegames.

It will be proved, that any game of this type has a value, that both players have good strategies of the so called Markov structure (essentially a lack of memory), and that value and good strategies may be determined by a dynamic programming approach.

The game we will consider is played as follows.

At each of a finite number of time instants, both players simultaneously choose an action out of some set of available actions. As a result of the two actions chosen, the state of the game changes and \( P_1 \) receives some (possibly negative) amount from \( P_2 \).

This type of game might be formalized as a treegame with subsequent moves instead of simultaneous ones, using an appropriate choice of information sets. However, we prefer a set up analogous to the usual set-up for Markov decision processes (see e.g. [4, 5, 19]):

Consider a dynamic system with finite state space \( S := \{1, \ldots, N\} \), which will be jointly controlled by the players \( P_1 \) and \( P_2 \) (the reader may think of a state of the system as of a position of the game).

For each state \( x \in S \) two finite, nonempty sets of actions (one set for each player) have been given, denoted by \( K_x \) for \( P_1 \) and by \( L_x \) for \( P_2 \),
At each of the times \( t = T, T - 1, \ldots, 1 \) both players have to select an action out of the set available to them. Suppose at a time \( t \) the system is in state \( x \), \( P_1 \) chooses action \( k \), and \( P_2 \) chooses action \( l \), then the system moves to the state \( y \) with probability \( p(y|x,k,l) \) and \( P_1 \) receives an amount \( r(x,k,l) \) from \( P_2 \). Moreover, we assume that, if the system moves to the state \( y \) as a result of the final players and chance moves, \( P_1 \) receives a terminal payoff \( q(y) \) from \( P_2 \).

This game will be called a \( T \)-stage Markov game with terminal payoff \( q \). Notice that we numbered the decision times in reversed order. It is more convenient for a dynamic programming approach to call the starting time \( T \) and the time for the final decisions 1.

Another observation is the following: if \( K_x \) consists of only one element for all \( x \in S \), then the Markov game reduces to a Markov decision process with essentially one player.

Example:

\[
\begin{align*}
S &= \{1, 2\}, K_1 = L_1 = \{1, 2\}, K_2 = L_2 = \{1\}, T = 2. \\
p(1|1,1,1) &= \frac{3}{4}, \quad p(1|1,2,1) = \frac{1}{8}, \quad p(1|2,1,1) = \frac{1}{2} \\
p(1|1,1,2) &= \frac{1}{4}, \quad p(1|1,2,2) = \frac{7}{8}, \\
p(2|x,k,l) &= -p(1|x,k,l). \\
r(1,1,1) &= 10, \quad r(1,2,1) = 5, \quad r(2,1,1) = -10 \\
r(1,1,2) &= 5, \quad r(1,2,2) = 8 \\
q(1) &= 6, \quad q(2) = -12.
\end{align*}
\]

In the example \( P_1 \) will try to keep the system in state 1, whereas \( P_2 \) prefers state 2. How they can achieve their goal in the best way will appear later on in this section.

Before starting with the analysis of Markov games, we will give a few definitions and notations.

A strategy \( \pi \) for \( P_1 \) in the \( T \)-stage Markov game is any function that specifies, for each time \( t = T, T - 1, \ldots, 1 \) for each state \( x \in S \), and for each history \( h_t \), the probability \( \pi(k|x,t,h_t) \) that action \( k \in K_x \)
will be chosen. By the history \( h_t \) we mean the history of the game up to reverse time \( t \), viz. \( h_t = ((x_T, k_T, l_T), \ldots, (x_{T+1}, k_{T+1}, l_{T+1})) \), the sequence of observed states and chosen actions at earlier times (formally we suppose \( h_1 \) to be the empty sequence).

We will call \( \pi \) a Markov strategy for \( P_1 \), if the \( \pi(k|x,t,h_t) \) do not depend on \( h_t \).

A policy for \( P_1 \) is any function \( f \) on \( S \), such that \( f(x) \) is a probability distribution on \( K_x \). Hence a Markov strategy \( \pi \) for \( P_1 \) consists of \( T \) policies for \( P_1 \). If \( \pi \) is a Markov strategy for \( P_1 \), we will denote it by \((f_T, f_{T-1}, \ldots, f_1)\), where \( f_t \) is the policy for \( P_1 \) to be used at reverse time \( t \).

Similarly strategies \( \rho \) and policies \( g \) for \( P_2 \) are defined.

As in the preceding sections, we suppose the expected reward for \( P_1 \) (or loss for \( P_2 \)) to be the criterion for choosing strategies. Therefore, we define 

\[ V(\pi, \rho)(x) \]

as the total expected payoff for \( P_1 \) over the duration of the game, when the game starts at \( t = T \) in state \( x \), while \( P_1 \) applies strategy \( \pi \), and \( P_2 \) applies strategy \( \rho \). \( V(\pi, \rho) \) then denotes a columnvector of size \( N \) with \( x \)-th coordinate \( V(\pi, \rho)(x) \).

By our definitions of strategy and policy we allow already for a kind of mixing.

Notice that \( V(\pi, \rho)(x) \) for fixed \( x \) characterizes a kind of matrixgame with moves for \( P_1 \) called \( \pi \) and for \( P_2 \) called \( \rho \). The only difference with the games of section 2 being that the numbers of allowed \( \pi \)'s and \( \rho \)'s are not finite.

**Example:** In the example of this section a policy for \( P_1 \) is completely characterized by: choose decision \( l \) with probability \( p \) if the system is in state \( l \). Hence a Markov strategy for \( P_1 \) is characterized by two numbers \( p_2 \) and \( p_1 \) with \( 0 \leq p_t \leq 1 \), \( t = 1,2 \).

If both players use the Markov policy "0,0", the corresponding \( V \)-values become: \( 16 \frac{21}{32} \) and \( -10 \frac{5}{8} \) for \( x = 1,2 \) respectively.

Similarly as in the matrixgames of section 2 our interest is in finding a saddlepoint of \( V(\pi, \rho) \), that means strategies \( \pi^* \) and \( \rho^* \) satisfying

\[ V(\pi^*, \rho^*) \leq V(\pi^*, \rho^*) \leq V(\pi^*, \rho^*) \]
where vector inequalities are supposed to be componentwise. If we can find such $\pi^*$ and $\rho^*$, we know (compare section 2) that $P_1$ can guarantee his expected reward at the level $V(\pi^*, \rho^*)$, whereas $P_2$ can prevent a larger expected loss than $V(\pi^*, \rho^*)$. If a saddlepoint exists, we call $V(\pi^*, \rho^*)$ the value of the game and $\pi^*, \rho^*$ good strategies for $P_1$ and $P_2$ respectively.

Our first aim will be to show the existence of good Markov strategies, since we know that in finite-stage Markov decision processes optimal Markov strategies exist (see e.g. Derman [4]).

In order to simplify the notations we introduce two operators on $\mathbb{R}^N$: let $f$ and $g$ be arbitrary policies for $P_1$ and $P_2$, then the operators $L(f, g)$ and $U$ on $\mathbb{R}^N$ are defined by

$$(L(f, g)v)(x) := \sum_k f^k(x) \sum_\ell g^\ell(x)[r(x, k, \ell) + \sum_y p(y|x, k, \ell)v(y)],$$

for $x \in S$, $v \in \mathbb{R}^N$, where $f^k(x)$ and $g^\ell(x)$ denote the probabilities for choosing actions $k$ and $\ell$ with policies $f$ and $g$ respectively.

$$(Uv)(x) := \max_{f} \min_{g} L(f, g)v,$$

where $\max_{f} \min_{g}$ is taken componentwise.

Hence $L(f, g)v$ is just the vector of expected payoffs for $P_1$ in the 1-stage Markov game with terminal payoff $v$, when $P_1$ uses policy (which is the same as a strategy in a 1-stage game) $f$ and $P_2$ uses $g$.

$(L(f, g)v)(x)$ may also be viewed as the expected reward in a matrixgame with action spaces $K_x$ and $L_x$, where $f(x)$ and $g(x)$ denote mixed strategies. Hence $(Uv)(x)$ is the value of the matrixgame with entries

$$r(x, k, \ell) + \sum_y p(y|x, k, \ell)v(y).$$

Now we may define the sequence $v_t$, $t = 0, 1, \ldots, T$, $v_t \in \mathbb{R}^N$ by

$$v_0(x) := q(x) \quad \text{for } x \in S,$$

$$v_t := Uv_{t-1} \quad \text{for } t = 1, \ldots, T.$$
Let \((f^*_t, g^*_t)\) be a saddlepoint of \(L(f, g) v_{t-1}\), hence

\[
L(f^*_t, g^*_t) v_{t-1} \leq L(f^*_t, g^*_t) v_{t-1} = v_t \leq L(f^*_t, g) \quad \text{for all } f, g.
\]

Interpretation: \(v_1\) is the value of the 1-stage Markov game with terminal payoff \(v_0 = q\); \(v_2\) is the value of a 1-stage Markov game with a terminal payoff determined by a pair of good strategies for the 1-stage Markov game with terminal payoff \(v_1\). Etc.

We expect that \(v_T\) will be the value of the T-stage Markov game with terminal payoff \(q\). Furthermore we expect \(\pi^*\) and \(\rho^*\) defined by \(\pi^* = (f^*_T, \ldots, f^*_1)\) and \(\rho^* = (g^*_T, \ldots, g^*_1)\) to be good strategies for the T-stage Markov game.

**Theorem:** The T-stage Markov game with terminal payoff \(q\) has value \(v_T\), furthermore the strategies \(\pi^*\) and \(\rho^*\) are good strategies for that game.

**Proof:** As a first part of the saddlepoint property, we will show

\[ V(\pi, \rho^*) \leq V(\pi^*, \rho^*) = v_T \quad \text{for any } \pi. \]

Denote by \(v(t, x, \pi)\) (for \(t = 1, \ldots, T\)) the conditional expected reward for \(P_1\) from reverse time \(t\) towards, if the system is in state \(x\) at time \(t\), history \(h_t\) has been observed and the strategies \(\pi\) and \(\rho^*\) are played.

Define: \(v_0(h_0, x, \pi) := q(x)\).

We will prove the assertion by induction. For \(t = 0\) we have by definition for all \(h_0\) and \(x\):

\[ v_0(h_0, x, \pi) \geq v_0(h_0, x, \pi^*) = v_0(x). \]

Now assume for \(t = 0, \ldots, n\) and for all \(\pi, h_t, x\):

\[ v_t(h_t, x, \pi) \leq v_t(h_t, x, \pi^*) = v_t(x). \]

So we have for all \(h_{n+1}, x\) and \(\pi\):

\[
\begin{align*}
v_{n+1}(h_{n+1}, x, \pi) &= \sum_k \pi(k | x, n, h_{n+1}) \sum_l g_{n+1}^*(x) [r(x, k, \ell) + \sum_y p(y | x, k, \ell) v_n(\pi^*, h_{n+1}^* (x, k, \ell), y)] \\
&\leq \sum_k \pi(k | x, n, h_{n+1}) \sum_l g_{n+1}^*(x) [r(x, k, \ell) + \sum_y p(y | x, k, \ell) v_n(y)] \\
&\leq v_{n+1}(x) = v_{n+1}(h_{n+1}, x, \pi^*),
\end{align*}
\]
where \( h_{n+1}^{(x,k,l)} \) denotes the concatenation of \( h_{n+1} \) with \((x,k,l)\) resulting in a history \( h_n \). The first inequality follows immediately from the induction assumption for \( t = n \), and the latter one follows from the definition of \( v_{n+1}^{(x)} \) and \( g_n^* \). The final equality follows from

\[
v_{n+1} = L(f_{n+1}^*, g_{n+1}^*)v_n
\]

and the induction assumption.

Hence for all \( x \in S \):

\[
v_T(h_T, x, \pi) \leq v_T(h_T, x, \pi^*) = v_T(x) \text{ or } V(\pi, \rho^*) \leq V(\pi, \rho^*) = V_T.
\]

The other part of the saddlepoint property may be shown analogously.

The proof of the preceding theorem is a shortcut of the proof given by Derman [4] for the existence of optimal Markov strategies in finite-stage Markov decision processes.

We have permitted the players to use non-Markov strategies, however, we showed that a player may just as well restrict himself to the use of Markov strategies. Zachrisson [20], who also considered this type of game, silently assumed that both players would only use Markov strategies. Under this condition he proved that the game has a value. Furthermore he proved, that this value and good strategies may be determined by the dynamic programming approach we also used here.

Summarizing, we see that we have shown in this section, that the following algorithm provides the value of the game and good Markov strategies for both players:

Algorithm:

(i) set \( v_T(x) := q(x) \) for \( x = 1, \ldots, N \).

(ii) determine for \( t = 1, \ldots, T \) policies \( f_t^*, g_t^* \) satisfying for all \( f \) and \( g \)

\[
L(f_t^*, g_t^*)v_{t-1} \leq L(f_t^*, g_t^*)v_{t-1} \leq L(f_t^*, g_t^*)v_{t-1}
\]

and set

\[
 v_T := L(f_T^*, g_T^*)v_{t-1}.
\]

(iii) \( v_T \) is the value of the game and \( \pi^* := (f_T^*, \ldots, f_1^*) \), \( \rho^* := (g_T^*, \ldots, g_1^*) \) are good strategies for \( P_1 \), \( P_2 \) respectively.
Example: using the algorithm for the example of this section gives the following results:

\[ v_0(1) = 6, \quad v_0(2) = -12. \]

The computation of \( v_1(1) \) requires the solution of the matrix game

\[
\begin{pmatrix}
\frac{23}{2} & -\frac{5}{2} \\
-\frac{19}{4} & \frac{47}{4}
\end{pmatrix}.
\]

By the method of section 2 or by elementary geometry (since the matrix is \( 2 \times 2 \)) one finds: \( f^*_1(1) = \frac{33}{61} \), \( g^*_1(1) = \frac{57}{122} \) and \( v_1(1) = \frac{493}{122} = 4.04 \), \( v_1(2) = -13. \)

The computation of \( v_2(1) \) requires the solution of the matrix game

\[
\begin{pmatrix}
9.78 & -3.74 \\
-5.87 & 9.91
\end{pmatrix}.
\]

This gives: \( f^*_2(1) = 0.54 \), \( g^*_2(1) = 0.47 \), and \( v_2(1) = 2.56 \), \( v_2(2) = -14.48. \)

5. Comments and extensions: The first point to comment on, is the amount of work involved in using the algorithm of the previous section. For each stage we have to solve a matrix game for each state. Hence all together \( NT \) matrix games. These matrix games may be solved by linear programming as shown in section 2. The size of the linear programming problems depends heavily on the size of \( K_x \) and \( L_x \). If one of the sets \( K_x, L_x \) contains only one element, the linear programming problem may be replaced by a maximum or minimum operation. Moreover one may choose degenerated distributions for \( f^*_t(x) \) and \( g^*_t(x) \), thus for state \( x \) mixing won't be necessary. If both \( K_x \) and \( L_x \) contain only one element, then there is no problem at all (as for state 2 in the example of section 4).

If for all \( x \) at least one of the sets \( K_x, L_x \) contains only one element (in which case game has perfect information) we need not consider mixed actions, at all. Furthermore no nontrivial matrix games have to be solved: only minimization and maximization will be required. As a result the amount of work that has to be done to solve the T-stage Markov game will be the same as for a T-stage Markov decision process of the same size, the only
difference being that some stated require minimization instead of maximization.

Now we will make some remarks on the possibilities of weakening the conditions in section 4:

a. We supposed that \( S, K, L, p(y|x,k,q), r(x,k,q) \) were the same for all time instants. We did not use this supposition in the proof of our theorem in any essential way. The weakening of this condition is not very essential, since any \( T \)-stage Markov game without these stationarity property may easily be transformed (by introducing some new states) in an equivalent \( T \)-stage Markov game satisfying the requirements of section 4.

b. We supposed for all \( x, k, \) and \( q \)

\[
\sum_y p(y|x,k,q) = 1.
\]

However if we allow for some or all \( x, k, \) and \( q \)

\[
\sum_y p(y|x,k,q) < 1,
\]

only trivial changes of the proof are needed to obtain the result of section 4.

One might interprete

\[
1 - \sum_y p(y|x,k,q)
\]

as the probability of a premature ending of the game.

c. One easily interpretes our finite number of time instants as deterministic equidistant points of time. However, this is not the only possibility. Actually they may be stochastic, e.g. the transition time may be distributed according to a distribution function \( F(.|x,y,k,q) \) where \( k \) and \( q \) are the selected actions in the starting point \( x \) of the transition, and \( y \) is the result of the transition (semi-Markov behaviour).

Of course one might introduce more intricate decision rules by taking into account the transition times. However, one easily argues that this cannot alter the value of the game.

d. Instead of considering the criterion of total expected rewards, one might prefer to use total expected discounted rewards. For the game with equidistant time instants this just requires replacement of \( p(y|x,k,q) \) by \( \beta p(y|x,k,q) \) in most places. For \( \beta \) all nonnegative real values are allowed. For the semi-
Markov type of transition times we may use any $\beta \in [0,1]$; if we want to use $\beta > 1$, we should require for all $y,x,k,l$

$$\int_0^{\infty} \beta^t dF(t|y,x,k,l) < \infty.$$ e. the rewards $r(x,k,l)$ need not be deterministic (given $x,k,l$), but may as well be expectations. For instance, if the amount $P_1$ receives from $P_2$ is equal to $r(y|x,k,l)$ when the transition from $x$ to $y$ is made under the actions $k$ and $l$, then $r(x,k,l)$ may be defined by

$$r(x,k,l) := \sum_y p(y|x,k,l)r(y,x,k,l).$$

Most of the extension possibilities have been worked out in more detail in [14].

The final section of this paper will be devoted to another (actually the most interesting) extension, namely the case $T = \infty$.

6. Infinite-horizon Markov games: So far we considered finite-stage games only: the treegames of section 3 and the Markov games of section 4 all had finite time horizons.

The results of section 4 may be of help for some infinite horizon Markov games as well.

One of the difficulties of infinite-horizon Markov games is the fact that dynamic programming procedures need endpoints to start the backwards induction. Another difficulty is that for infinite horizon Markov games the total expected rewards need not converge.

We will handle these difficulties by considering some types of infinite-horizon Markov games, each with special extra requirements.

Throughout this section the basic assumptions are the assumptions of section 4 with $T = \infty$, (the terminal payoff $q$ is obsolete now). So we start with the concept of an $\infty$-stage Markov game. We will consider 4 types of $\infty$-stage Markov games, by making successively 4 extra assumptions. The time is considered as normal order time again: $t = 1,2,\ldots$. Strategies, policies, and Markov strategies for both players may be defined as in section 4.

a. $\infty$-stage discounted Markov games: the special assumption here is, that $V(\pi,\rho)$ is replaced by $V_\beta(\pi,\rho)$ with $0 < \beta < 1$, where $V_\beta(\pi,\rho)$ denotes the
column vector of the total expected discounted rewards (using discount-factor $\beta$) for $P_1$ for the different starting states $x$. Hence, if $P_1$ receives a reward $A$ at time $t$, it is only evaluated at $\beta^{t-1}A$.

We will demonstrate that this game has a value.

Consider the $T$-stage Markov game with terminal payoff 0. As we have seen in section 4, this game has a value and both players have good Markov strategies. This remains true when we discount rewards (section 5).

Now, let $P_1$ play in the $\infty$-stage discounted game a good strategy for the $T$-stage game amplified by arbitrary decisions after time point $T$. Then his expected discounted rewards will be at least equal to

$$v_T = \frac{\beta^T}{1-\beta} M_e,$$

where $v_T$ is the value (vector) for the $T$-stage (discounted) game, $e$ is a column vector consisting of ones, and

$$M := \max_{k,\ell,x} |r(x,k,\ell)|.$$

In a similar way, $P_2$ may restrict his expected discounted losses to

$$v_T + \frac{\beta^T}{1-\beta} M_e.$$

Notice, that the difference between the two boundaries vanishes if $T$ tends to infinity.

Again we consider the operators $L(f,g)$ and $U$ of section 4, however with the slight modification of all transition probabilities $p(y|x,k,\ell)$ being replaced by $\beta p(y|x,k,\ell)$.

For this modified operator $U$ one easily verifies

$$\max_x |(Uv)(x) - (Uw)(x)| \leq \beta \max_x |v(x) - w(x)| \quad (v,w \in \mathbb{R}^N)$$

So $U$ is a contractive operator on $\mathbb{R}^N$ (with maximum norm) with contraction radius $\beta$ (for details see [15]).

As a consequence we have

1. The set of equations $Uv = v$ with $v \in \mathbb{R}^N$ has only one solution (let us call this solution $v_\beta$);
2. For any $v \in \mathbb{R}^N$ we have

$$\lim_{T \to \infty} U^T v = v_\beta \quad \text{(componentwise convergence)}$$
Thus, since $v_T = U^0$, we have
\[
\lim_{T \to \infty} v_T = v_\beta.
\]

Hence $v_T + \frac{\beta^T}{1 - \beta} M e$ and $v_T - \frac{\beta^T}{1 - \beta} M e$ tend to $v_\beta$, which is consequently the value (vector of the $\infty$-stage discounted Markov game).

If $T$ is chosen in such a way that
\[
\frac{2\beta^T}{1 - \beta} M \leq \varepsilon,
\]
for a prescribed $\varepsilon > 0$, then good strategies for the $T$-stage game with arbitrary continuations are $\infty$-good strategies for both players (a strategy $\pi_\varepsilon$ for $P_1$ is called $\varepsilon$-good strategy for player $P_1$ if for all:
\[
V_\beta(\pi_\varepsilon, \rho) \geq v_\beta - \varepsilon \varepsilon).
\]

Hence $\varepsilon$-optimal Markov strategies for both players may be determined by a dynamic programming type of procedure, which at the same time gives a $\varepsilon$-approximation of $v_\beta$.

Like in the theory of Markov decision processes a more efficient use of the dynamic programming procedure $\varepsilon$-good stationary Markov strategies nearly always in much less steps than needed to reach a $T$ with
\[
\frac{2\beta^T}{1 - \beta} M \leq \varepsilon.
\]

This will not be worked out here. We refer to [15] for details in the case of Markov games and we refer to Macqueen [8] and van Nunen [10] for the same ideas in Markov decision processes.

Alternative types of dynamic programming procedures (based on other $U$-operators) may be used as well as in Markov decision processes. This has been worked out in [15] using the concept of stoppingtime-based $L(f, g)$-operators as introduced in [17].

b. Shapley's $\infty$-stage Markov games: Shapley [12] did not use discounting of rewards in order to obtain a properly defined criterion function, but supposed (with some purpose) for all $x,k,\ell$
\[
\sum_{y} p(y|x,k,\ell) < 1.
\]
As a result of the assumption, the operator $U$ again becomes a contracting operator. Hence the same arguments as in the discounted case ($\beta < 1$) lead to the same results. The refinements in order to obtain a more efficient procedure require somewhat more details. Actually this case is equivalent to the discounted case with semi-Markov behaviour of transitions. Compare [14]

c. $\infty$-stage Markov games with probabilistic termination: For this type of games we suppose that for certain $T_0$ and $\epsilon > 0$ any pair of strategies incurs a probability of at least $\epsilon$ that the game ends before or on $T_0$. This condition is weaker than the condition in the preceding type. For Markov games with probabilistic termination the operator $U$ is not necessarily contracting. However $U^{T_0}$ is contracting with a contraction radius of at most $1 - \epsilon$. In this case the proofs for the properties of dynamic programming type of procedures may be easily adapted from the discounted case. (Compare [7] and [14].) Another way of treating this situation is by introducing another norm in $\mathbb{R}^N$ than the maximum norm: there always exists a norm such that $U$ is strictly contracting with respect to that norm. Then the proofs for the discounted case may be rewritten in the new norm. See for details [18].

d. $\infty$-stage Markov games with decisional termination: In this game it is supposed that all immediate rewards $r(x,k,l)$ are strictly positive and that the minimizing player may restrict his losses to some finite amount. See [7, 14]. In [14] the minimizing player has in each state the possibility of terminating the game immediately against some terminal (state and/or action dependent) loss. And it is shown that though $U^T$ might not be contracting for any $T$ at all the dynamic programming approach still gives interesting results.

Example: $s = K_1 = L_1 = K_2 = \{1,2\}$, $L_2 = \{1,2,3\}$. Furthermore 
$r(1,1,1) = r(1,2,1) = 3, r(1,1,2) = r(1,2,2) = r(2,1,1) = r(2,1,2) = r(2,1,3) = 1, r(2,2,1) = 8, r(2,2,2) = 1, r(2,2,3) = 2,$
$p(1|1,2,2) = p(2|2,2,2) = p(1|2,2,3) = 1$ and $p(x|y,k,l) = 0$ if either $k$ or $l$ equals 1. Thus whenever $P_1$ or $P_2$ takes action 1 the game terminates immediately.

It is easily seen that $v_1 = \binom{1}{1}, v_2 = \binom{2}{2}, v_3 = \binom{3}{3}, v_4 = \binom{3}{4}$ and for $n \geq 5$ $v_n = \binom{3}{5}$. So $\binom{3}{5}$ is the value of the game. For $P_2$ the strategy which
prescribes action 1 in state 1 and action 3 in state 2 is good and for \( P_1 \) any strategy which prescribes action 2 in state 2 is good.

Another way of attacking \( \infty \)-stage Markov games may be the replacement of total expected (possibly discounted) rewards by average payoff over a long time. One may show, that, when for each pair of pure (= nonrandomized) policies the corresponding transition probabilities yield a Markov chain with only one communicating class of states, the game possesses a value and both players have good stationary strategies (see e.g. theorem 2 in Sobel [13]).

References: