Markov Decision Processes and Regular Events

Costas Courcoubetis and Mihalis Yannakakis

***Abstract*— Desirable properties of the infinite histories of a finite-state Markov decision process are specified in terms of a finite number of events represented as ω-regular sets. An infinite history of the process produces a reward which depends on the properties it satisfies. The authors investigate the existence of optimal policies and provide algorithms for the construction of such policies.**

***Index Terms*—Complexity, discrete-event systems, Markov decision processes, ω-automata, probabilistic games, regular languages, verification.**

I. Introduction

MARKOV decision processes (MDP's) have been widely used to model systems in which the evolution is controlled by available decisions which can be taken when the system is in certain states. Such processes with finite sets of states and decisions have been widely investigated; see [9], [23], [11], [5], [24], and [19]. A finite-state MDP consists of a Markov chain with transition probabilities controlled by the available decisions. The traditional cost structure associated with such models is to assume that at each time the process is in a particular state and a certain action is taken, a reward depending on that state and action is obtained. There is a large body of theory developed for investigating the optimal operation of such systems, where one seeks to maximize performance measures such as the expected reward per unit time or the total discounted reward for some given discount. The formulation of such cost structures makes dynamic programming a natural framework for solving the above optimization problems.

In this paper we are interested in designing optimal decision strategies for MDP's under a new type of cost structure. Consider a system that is modeled as a finite-state MDP. We would like to control the operation of the system so that it satisfies certain properties. Our definition of "property" is quite general: we view the infinite history of the MDP as an infinite word over the alphabet consisting of the state and action space of the MDP and allow a property to be any ω-regular (or rational) set over this alphabet. In general, we may have a number of such properties that we would like to satisfy. We do not assume that the properties are mutually consistent, that is, in general it may be impossible to satisfy all of them simultaneously, and one could assign a different importance in the system having histories satisfying the different properties. Then, if one associates with each infinite history a reward which depends on the properties that the history satisfies, one would like to design a decision policy that maximizes the expected reward. This is the problem we investigate in this paper.

We are given a finite state MDP and a finite number of ω-regular properties. We assume that these properties are given in terms of the corresponding automata on infinite words, called Buchi automata. Every set of properties has an associated (nonnegative) reward, which is produced by a history if it satisfies the properties in the set. We assume that the reward structure satisfies the following monotonicity constraint: if $R \subset Q$ are two sets of properties, then the reward associated with Q is at least as large as the reward associated with R. This constraint corresponds to the fact that we would like to satisfy as much as possible the given properties. We seek a policy which maximizes the expected reward. The class of policies over which we optimize includes policies that randomize and are allowed to use complete information about the past history of the system. We devise an algorithm which computes the maximum reward that can be produced and constructs an optimal policy. The time complexity of our algorithm is polynomial in the size of the MDP and exponential in the sizes of the automata that represent the properties. The exponential dependence of the algorithm is unavoidable; even a very special case of our problem is hard for deterministic exponential time [7].

The optimal decision policy does not necessarily take the same action every time it passes through the same state, that is, the best possible action at every time depends not only on the current state of the process but also on the past history. However, we show that it suffices to remember only a bounded amount of information about the past history. From the given MDP M and the given set of properties, we construct a new equivalent MDP M' which is defined on a larger state space, is coupled with M (in the sense that it is defined on the same probability space), and M' has a memoryless optimal decision policy, i.e., the policy depends on the history of the process only through the current state of M'. One can view M' as a *refinement* of M, on which a stationary optimal policy exists, which is also optimal for M.

Some related work which does not address optimization issues and randomization, can be found in the control of discrete event systems; see [6], [25], [34], [13], [29], and [22]. A discrete-event system models a plant, with a controller attached to it which enables or disables certain controlled transitions based on the history of the system, so that the evolution of the system satisfies certain correctness properties. These properties are expressed in a way similar to ours

Manuscript received August 11, 1992. Recommended by Associate Editor, P. J. Ramadge. This work was supported in part by ESPRIT BRA under project SPEC.

C. Courcoubetis is with the Computer Science Department, University of Crete, and Computer Science Institute, Forth, Heraklion, Crete, Greece.

M. Yannakakis is with Bell Laboratories, Murray Hill, NJ 07974 USA.

Publisher Item Identifier S 0018-9286(98)07512-6.

in terms of regular sets. A related approach is the one followed in [16], [3], [17], and [32], where the authors consider the problem of designing "reactive modules," satisfying a certain specification in terms of an ω-regular set. In this approach a module corresponds to the controller and the environment of the module corresponds to the plant. An infinite history in this system corresponds to an interleaving of actions from the environment and the module; the module must choose its actions in such a way so that for any actions of the environment, the module can succeed in producing a history in the set given by the specification. Both the discrete-event system approach and the reactive module approach derive conditions under which such a controller exists and provide algorithms for constructing one. There is an important difference between the problems considered in the above papers and the problem in this paper. There the problem can be formulated as a game between the controller and the plant: a controller (module) exists if it can win over any strategy of the plant (environment). Here there is no game involved. Our problem cast in the above framework corresponds to the plant (environment) using randomization for choosing its actions, and the controller (module) using a strategy for maximizing its probability to win.

Another application of the optimization problem that we solve in this paper (which was in fact our original motivation) is in the *automated analysis* of the correctness of finite-state programs which include randomization and nondeterminism (the so-called "concurrent probabilistic programs" in the literature), such as computer-communication protocols. Randomization is by now an important tool in the design of efficient algorithms and protocols. It has been realized that randomization may often lead to algorithms with better complexity than deterministic ones, or even allow for the solution of problems (especially in distributed computation) that cannot be solved deterministically. Such programs and protocols are typically modeled as finite-state transition systems that include both nondeterministic and probabilistic transitions. The nondeterminism is due to the asynchronous execution of the processes, or because some transitions are controlled by the environment (users, scheduler of the CPU, etc.). The probabilistic transitions are due to randomization in the protocol. Even in the case of protocols that do not use randomization, one may want to regard certain transitions as being probabilistic in order to analyze the behavior of the protocol under some probabilistic assumptions about the environment (for example, a message is corrupted with certain probability, a buffer can be allocated with certain probability, etc.). We will define formally the model for such programs in the next section and observe that it is closely related to MDP's.

The wide applicability and use of concurrent probabilistic programs has motivated the interest in the development of formal automatic methods to analyze such programs and verify that their computations meet their specifications. A popular language for specifying temporal properties of computations is propositional linear time temporal logic (PTL) [15]. More powerful languages, such as the extended temporal logics (ETL) of [33] and [35] have been also proposed. It is known that the sets of computations (infinite histories) that satisfy properties expressed in the above logics are ω-regular: from a given formula ϕ of PTL (or ETL) one can construct an ω-automaton A (of size exponential in the size of ϕ) which accepts the computations that satisfy ϕ [35]. To verify whether a concurrent probabilistic program P meets a specification ϕ, one assumes a worst case scenario for the nondeterministic transitions controlled by the environment. That is, one considers the environment as a malicious adversary who tries to cause the program to violate its specification. Such a program is correct if for any strategy of the environment, its histories satisfy ϕ with probability one.

A number of papers in recent years have investigated the verification problem for such programs, i.e., testing whether the program meets a desired specification with probability one under worst case nondeterminism, and have introduced methods for solving this problem, proving tight upper and lower bounds on its complexity [12], [10], [18], [30], [31], [7], [8]. However, many times probabilistic programs are supposed to satisfy certain requirements only with some given probability p which may be smaller than one. For example, a certain protocol might be allowed to lose a message with probability $1-p$. The existing techniques deal only with the *qualitative* aspects of probabilistic verification, and thus are not capable of verifying such programs. This problem can be posed naturally as an optimization problem for MDP's: given an MDP P (the program) and a specification ϕ with a desired probability of satisfaction p, find the optimal decision policy (worst possible nondeterminism) that maximizes the probability that the infinity history satisfies $\neg\phi$ (i.e., the process misbehaves), and check that the above probability is less than $1-p$. We can solve this problem by first constructing an automaton A corresponding to $\neg\phi$ and then applying our algorithm for maximizing the probability of a (single) regular event. The complexity of this algorithm is exponential in the size of A and polynomial in the size of the program. Similarly, our methods can be used to perform a quantitative analysis on the behavior of a protocol under probabilistic assumptions on certain parts of the environment.

The present paper is organized as follows. In Section II we review briefly the necessary background on ω-regular sets, automata over infinite words, and MDP's. We define a model for an MDP which has the actions as explicit transitions; we call this model a *controlled Markov chain*. In Section III we deal with the case of a single regular event. We prove some structural properties of automata on infinite words and controlled Markov chains and then use these properties to solve the optimization problem. In Section IV we deal with the case of many events. First, we solve the special case when the events are disjoint and then finally address the general case. In Section V we analyze the complexity of our algorithm, in Section VI we provide some extensions of the results, and finally in Section VII we point out interesting directions for further research. For facilitating the reading of the paper, we included all the proofs of the paper in the Appendix.

II. Background

This section introduces the basic background information about ω-automata and MDP's. In this section as well as in

the rest of the paper we will obey the following notational convention as much as possible. We use lowercase letters such as x, x_1, to denote actual values from the set X. We use uppercase letters such as X_k to refer to random variables; in this case this corresponds to the kth value of the chain defined over the set X. We use boldface letters to refer to infinite sequences; $\mathbf{X}$ denotes an infinite sequence with elements from X. We use symbols like $\mathbf{A}$ to refer to sets of infinite sequences.

A. Automata on Infinite Words and ω-Regular Sets

In this section we define the ω-automata and ω-regular languages and introduce the necessary notation. For a more elaborate introduction to this subject, the reader is referred to [26], [31], [14], and [27].

The ω-automata are extensions of the finite-state automata over finite words into automata over infinite ones. This extension greatly increases the power of our specifications. With automata over finite words one can specify properties of systems involving finite histories, and hence of systems which eventually terminate. Many systems of interest such as protocols and manufacturing plants do not terminate for any practical purpose, and correctness involves properties of infinite histories which cannot be expressed in terms of finite ones. For example, a property such as "if a message is transmitted infinitely often, it will be eventually received," or "always, a lossless communication channel when it receives a message it will eventually deliver it," cannot be expressed by a regular set and hence by an automaton over finite words. Such properties can easily be expressed as ω-regular sets and hence by an ω-automaton. A natural question the reader might ask is why use ω-regular sets and not allow even more powerful specification languages, such as ω-context-free languages. The answer is that in this case the optimization problem we consider becomes undecidable, and hence no algorithm to solve a general instance of the problem exists. Note that even if we consider sets of finite strings, the problem of determining if a regular set is included in a context-free set is undecidable. We continue with the formal definition of ω-automata.

A *transition table* is a tuple $\tau = (\Sigma, S, \rho)$, where Σ is the finite alphabet, S is a finite set of states, and $\rho : S \times \Sigma \to 2^S$ is the transition function. A state s is *deterministic* if $|\rho(s,a)| \leq 1$ for all letters $a \in \Sigma$; the table τ is deterministic if all the states are. A run of τ over a finite word $w = a_1 \cdots a_n$ over Σ is a sequence of states $\mathbf{s} = s_0, \cdots, s_n$ such that $s_{i+1} \in \rho(s_i, a_i)$ for $0 \leq i \leq n-1$. A *run* of τ over an infinite word $w = a_1 a_2 \cdots$ is an infinite sequence of states $\mathbf{s} = s_0, s_1, \cdots$ such that $s_{i+1} \in \rho(s_i, a_i)$ for $i \geq 0$. For an infinite run $\mathbf{s}$, the set $\inf(\mathbf{s})$ is the set of states that repeat infinitely often in $\mathbf{s}$, i.e., $\inf(\mathbf{s}) = \{v : |\{i : s_i = v\}| = \infty\}$.

Transition tables are often represented by directed graphs with labeled edges. The graph G_τ for a given table $\tau = (\Sigma, S, \rho)$ has one node for every state of τ and has an edge from state s_i to state s_j labeled by a letter a if and only if $s_j \in \rho(s_i, a)$. There is a correspondence between runs of the table τ and paths of its graph G_τ. Standard graph theoretic concepts and algorithms are often used to analyze transition tables and automata. A *strongly connected subset* of states of a transition table τ is a set C of states such that for any two states y_1, y_2 of C, there is a path going from y_1 to y_2 and a path going from y_2 to y_1. A strongly connected subset which is maximal is called a *strongly connected component*, (s.c.c). The states of a transition table (directed graph) can be partitioned in linear time into strongly connected components (see, e.g., [2]). An s.c.c. is called *trivial* if it does not contain any edges, i.e., it consists of a single state without a self-loop; otherwise, it is called *nontrivial*. An s.c.c. that does not have any edges leaving it, is called a *bottom strongly connected component* (b.s.c.c).

For a transition table $\tau = (\Sigma, S, \rho)$, we denote by $\det(\tau)$ the deterministic table that results from applying the usual subset construction to τ. That is, the state space of $\det(\tau)$ is the powerset 2^S (set of all subsets of states of τ), the alphabet is the same, Σ, and for every subset $Q \subseteq S$ and letter $a \in \Sigma$, the table $\det(\tau)$ has a transition on letter a from Q to $\cup_{q \in Q} \rho(q, a)$. For a state s of τ, we use $\det(\tau, s)$ to denote the table obtained from $\det(\tau)$ by restricting to the states that are reachable from $\{s\}$.

Let Σ^ω be the set of infinite words (strings) over the alphabet Σ and Σ^* be the set of finite words over the alphabet Σ including the empty string (ω and * denote infinite and finite repetition, respectively). A *language* of infinite words (ω-language) over Σ is any subset of Σ^ω. An *ω-automaton* A consists of a table $\tau_A = (\Sigma, S, \rho)$, a set of starting states $S_0 \subseteq S$, and an acceptance condition. The automaton is deterministic if τ is deterministic and $|S_0| = 1$. In the usual automata on finite strings, acceptance of a run is determined by the final state of the run; in the case of ω-automata and infinite runs $\mathbf{s}$, acceptance is determined by its infinity set $\inf(\mathbf{s})$. There are several ways in which one can specify which infinity sets are accepting and which ones are not. The simplest one is the Buchi acceptance condition. A Buchi automaton $A = (\tau_A, S_0, F), F \subseteq S$ is an ω-automaton with the following acceptance condition. A accepts an infinite word w if there is a run $\mathbf{s}$ of τ_A over w such that $\mathbf{s}$ starts with a state in S_0 and some state in F repeats in $\mathbf{s}$ infinitely often, that is, $\inf(\mathbf{s}) \cap F \neq \emptyset$.

We define as $\mathbf{A} \subseteq \Sigma^\omega$ the set of infinite words accepted by the ω-automaton A. Nondeterministic Buchi automata accept exactly the ω-regular languages; deterministic Buchi automata accept a proper subset. Since in the rest of the paper we will be using only automata with the Buchi type of acceptance condition, we will refer to them as simply automata. We will usually denote an automaton and the language that it accepts by the same symbol, such as A and $\mathbf{A}$, respectively.

B. MDP's and Controlled Markov Chains

For an MDP we use the standard definition found in [9]. Let Y be the finite set of states of the system. Associated with every state $i \in Y$ there is a (finite) set K_i of possible actions, which are available when the process is in state i. A decision policy is a prescription for choosing actions at each point in time. Such a policy must select an action from the set K_i when the process is in state i, and it can base its selection on the history of the process up to that time. To allow

greater generality, the selection policy can use randomization: it selects action $a \in K_i$ with probability p_a, where p_a depends on the history. The actions influence the evolution of the process Y_t. Once action a has been selected, the process transitions with transition probabilities depending on a. We formalize the above discussion as follows.

Let $\{Y_t, t = 0,1,\cdots\}$ be the sequence of states of the process, let $\{a_t, t = 0,1,\cdots\}$ be the sequence of actions taken, and let the history of the system up to time t be $\mathbf{H}_t = Y_0, a_0, \cdots, Y_t, a_t$. Then a policy u is a set of functions $\{p_a(\mathbf{H}_{t-1}, Y_t), a \in K_{Y_t}, t = 0,1,\cdots\}$ satisfying $0 \leq p_a(\mathbf{H}_{t-1}, Y_t) \leq 1$ and $\sum_a p_a(\mathbf{H}_{t-1}, Y_t) = 1$. That is, a policy chooses the probability of the next action based on the past history of the system. If the system is in state i and action a is chosen, then the system transitions according to a given set of transition probabilities $\{q_{ij}(a) : j \in Y\}$. We are also given the initial probability distribution $P_0(Y_0 = i)$. To summarize, an MDP consists of a set of states Y, a set of actions K_i, $i \in Y$, a set of transition probabilities $\{q_{ij}(a) : i, j \in Y, a \in K_i\}$, and an initial probability distribution $\{P_0(Y_0 = i) : i \in Y\}$.

A policy u induces a probability measure P_u for the stochastic process $\{(Y_t, a_t), t = 0,1,\cdots\}$. Formally, there is a probability space $(\Omega, \mathbf{F}, P_u)$ where the sample space Ω is the set of all infinite histories, i.e., sequences of alternating states and actions $y_0, a_0, y_1, a_1, \cdots$ such that $a_i \in K_{y_i}$ for all i. The family of measurable sets $\mathbf{F}$ is the Borel field generated by the *basic cylinder sets* $C(h)$, where h ranges over all finite histories of length one or more, and $C(h)$ is the set of infinite histories $\mathbf{H}$ with prefix h. (Note that Ω and $\mathbf{F}$ do not depend on the policy.) The measure P_u is defined on the basic cylinder sets (and can then be extended uniquely to the rest of $\mathbf{F}$) as follows: 1) if h is of length one, i.e., it is a state $y_0 \in Y$, then $P_u(C(h))$ is the initial probability $P_0(Y_0 = y_0)$; 2) if h ends with an action, i.e., $h = y_0, a_0, \cdots, y_t, a_t$, then $P_u(C(h)) = P_u(C(h')) \cdot p_{a_t}(h')$ where $h' = y_0, a_0, \cdots, y_t$ and $p_{a_t}(h')$ is the probability that policy u chooses action a_t given history h'; 3) if h is of length two or more and ends with a state, i.e., $h = y_0, a_0, \cdots, y_t, a_t, y_{t+1}$, then $P_u(C(h)) = P_u(C(h')) \cdots q_{y_t y_{t+1}}(a_t)$ where $h' = y_0, a_0, \cdots, a_t$ and $q_{y_t y_{t+1}}(a_t)$ is the transition probability from y_t to y_{t+1} if action a_t is chosen. For more background on MDP's see [9] and [19].

Let $K = \cup_i K_i$ be the set of all actions and consider any mapping $V : Y \cup K \to \Sigma$, where Σ is some finite alphabet. Let $\mathbf{H} = Y_0, a_0, Y_1, a_1, \cdots$, denote an infinite history of the system and denote by $V(\mathbf{H})$ the sequence $V(Y_0), V(a_0), \cdots$. We call the above sequence the *valuation* of the sequence $\mathbf{H}$. The use of V is to define the observables of the system. For example, we might "paint" the states with red and green and the actions with black and white. In this case our alphabet Σ is the set of colors red, green, black, white, and the observable history of the above system consists of an infinite interleaving of the above colors. Note also that Σ can be much smaller than Y.

Desired properties of the MDP are specified in terms of the ω-regular languages $\mathbf{A}_i$, $i = 1, \cdots, m$, over the alphabet Σ (over the observables). If the valuation $V(\mathbf{H})$ of a history $\mathbf{H}$ is in the language $\mathbf{A}_i$, we will say that the history $\mathbf{H}$ satisfies property $\mathbf{A}_i$. For every ω-regular property $\mathbf{A}_i$, the set of histories $\mathbf{H}$ of the MDP that satisfy property $\mathbf{A}_i$ is measurable [30]. The cost structure we use is defined as follows. Let R denote any subset of the set of indexes $\{1, \cdots, m\}$, and define the set

$$\mathbf{A}_R = \bigcap_{i \in R} \mathbf{A}_i \bigcap_{i \notin R} \mathbf{A}_i^c \tag{1}$$

where $\mathbf{A}^c$ is the complement of $\mathbf{A}$. Clearly, the sets $\mathbf{A}_R$ corresponding to different R's are disjoint, and a history in $\mathbf{A}_R$ does not satisfy property A_j if j is not in R. For every subset R of indexes there is an associated reward k_R. A history $\mathbf{H}$ produces reward k_R iff its valuation is in $\mathbf{A}_R$, i.e., the set of (indexes of) properties it satisfies is exactly R. We assume that the k_R's are nonnegative and satisfy the following monotonicity property:

$$R \subseteq Q \quad \text{implies} \quad k_R \leq k_Q. \tag{2}$$

In measuring the complexity of our algorithm, we do not assume that the rewards k_R are listed as part of the input for all possible 2^m subsets of indexes, but allow that they be implicitly specified. For example, a special case is when there is a positive weight w_i associated with every property $\mathbf{A}_i$, and the reward produced by a history is the sum of the weights of the properties that it satisfies. Our problem is to design an optimal policy u which maximizes the expected reward $\sum_R k_R P_u(V(\mathbf{H}) \in \mathbf{A}_R)$, where P_u is the probability measure defined by the MDP with policy u.

In this paper we will use an equivalent formulation for an MDP. In this formulation, actions are explicitly modeled as special transitions. This provides for a uniform treatment of actions and state transitions in the above problem. We use the term controlled Markov chain to refer to this formulation, which is very similar to the notion of concurrent Markov chain introduced in [30] to model finite state concurrent probabilistic programs.

Informally, a controlled Markov chain has two types of states, the randomizing states and the control states. When the chain is at a randomizing state, a transition is chosen randomly according to a probability transition matrix; when the chain is at a control state, a transition is chosen by a *policy*, possibly depending on the history of the chain. Formally, a *controlled Markov chain* M consists of a finite set X of states, which is partitioned into a set N_1 of *control* states and a set N_2 of *randomizing* states; a set $E \subseteq X \times X$ of arcs (transitions); an assignment of nonzero probabilities to arcs coming out of randomizing states, so that the probabilities on the arcs coming out of each randomizing state sum to one; an initial probability distribution, again summing to one. Let $\{X_t, t = 0,1,\cdots\}$ be the sequence of observed states, and let $\mathbf{H}_t = X_0, \cdots, X_t$ be the history of the system up to time t. A *policy* is a set of functions $\{p_{ij}(\mathbf{H}_t), i = X_t \in N_1, (i,j) \in E, t = 0,1,\cdots\}$, satisfying $0 \leq p_{ij}(\mathbf{H}_t) \leq 1$, and $\sum_j p_{ij}(\mathbf{H}_t) = 1$. The interpretation of this definition is that at each time t, if the system is in a control state, the policy chooses based on the past history the probabilities for the transitions out of the present state. For each policy u, the process $X_t, t = 0,1,\cdots$, is not Markov since at time t the distribution of X_t depends

on the entire past history of the process. (We can view the process as a Markov chain on an infinite state space, namely the set of all finite histories.)

Clearly, a controlled Markov chain corresponds to an MDP. Conversely, from an MDP we can construct an equivalent controlled Markov chain as follows. For each state $i \in Y$ of the MDP with associated set of actions $K_i = \{a_{i1}Q, \cdots, a_{ik_i}\}$, we have a control state i, randomizing states $(i, a_{i1}), \cdots, (i, a_{ik_i})$, and the control transitions $i \rightarrow (i, a_{i1}), \cdots, i \rightarrow (i, a_{ik_i})$. From every randomizing state (i, a_{ik}) we have randomizing transitions to all control states j with transition probabilities $p((i, a_{ik}) \rightarrow j) = q_{ij}(a_{ik})$. A valuation mapping V from the states and actions of the MDP to an alphabet Σ can be translated in a straightforward way to a valuation from the states of the controlled Markov chain to Σ.

Let A be an automaton over the alphabet Σ, and let M be a controlled Markov chain with state space X and with valuation mapping $V : X \rightarrow \Sigma$. We can define another automaton A' over the alphabet X such that any infinite trajectory $\mathbf{X}$ of M is accepted by A' if and only if its valuation $V(\mathbf{X})$ is accepted by A; the automaton A' has the same states as A, and for each state s and letter $x \in X$ its transition function is defined to be $\rho'_A(s, x) = \rho_A(s, V(x))$ where ρ_A is the transition function of A. Thus, for any policy u, $P_u(\mathbf{X} \in \mathbf{A}) = P_u(\mathbf{X} \in \mathbf{A}')$. A consequence of this observation is that we can assume without loss of generality that the alphabet Σ coincides with the state space X of the controlled Markov chain. We can now state our optimization problem as follows.

Problem: Given a controlled Markov chain over the state space X, the automata A_i, $i = 1, \cdots, m$, over the alphabet X, and the nonnegative constants k_R, $R \subset \{1, \cdots, m\}$, satisfying (2), find a policy u which maximizes the reward function $\sum_R k_R P_u(\mathbf{X} \in \mathbf{A}_R)$.

This is the problem which we investigate in the subsequent sections. In the rest of the paper we refer to controlled Markov chains as simply Markov chains, and we denote $P_u(\mathbf{X} \in \mathbf{A})$ by $P_u(\mathbf{A})$.

C. Some Examples

We give some examples of ω-regular sets and the corresponding automata. Let X be the alphabet, $x_1, x_2 \in X$, and as before $\mathbf{X}$ denotes an infinite word over X.

$A_1 = \{\mathbf{X} \mid x_1$ appears infinitely often$\}$.

$A_2 = \{\mathbf{X} \mid$ at all times, if x_1 appears, then x_2 eventually appears at some future time$\}$.

$A_3 = \{\mathbf{X} \mid$ every four steps, x_1 appears at least once$\}$.

The next set A_4 describes the erroneous behavior of a communication protocol and is borrowed from [1]. This behavior corresponds to the sequence of messages which the receiver part of the protocol gives as an output to the end user. There is an arbitrary number of messages two of which are distinct, say msg_1 and msg_2, the rest of the messages being undistinguishable. At every step, the receiver outputs the message it received or performs a "pause" operation. The output of the receiver takes values from the alphabet $\{r_1, r_2, r,$ pause$\}$, where r_1 and r_2 correspond to the reception of the two distinguished messages msg_1 and msg_2, respectively, whereas the reception of all other messages is mapped to the letter r. The pause operation corresponds to the letter "pause." The sender sends an infinite number of messages to the receiver as follows. It starts by sending an arbitrary number of undistinguishable messages, then it transmits msg_1 followed by an arbitrary number of undistinguishable messages followed by msg_2, and then it continues forever transmitting undistinguishable messages. The set of correct output histories for this system is the set of all histories for which r_1 and r_2 occur exactly once, and r_1 occurs before r_2. The set A_4 is the complement of the above set, i.e., a history in that set corresponds to a loss of r_1 or r_2, or to the reception of duplicates of the above messages, or to the case that these two messages are received out of order. If we model the protocol as an MDP, then, maximizing the probability of A_4 in this context will provide the upper bound for the probability of incorrect behavior of the protocol under all possible decisions (by the users, the CPU scheduler, and all other nondeterministic factors like buffer overflows at the worst possible moment, etc.).

The automata for the above sets are shown in Fig. 1.

III. Maximizing the Probability of a Regular Event

In this section we deal with a controlled Markov chain M having state space X and a single automaton A defined over the alphabet $\Sigma = X$. We want to construct a policy that maximizes the probability that the history is accepted by A. We shall construct a new controlled Markov chain M' and we shall identify a distinguished set T of *target* states of M' such that the maximum probability (over all policies of M) that a history of M is accepted by the automaton A is equal to the maximum probability (over all policies of M') that a history of M' hits a state from T; the latter probability can be computed by linear programming techniques. It is useful to think of M' as being a *refinement* of M, in the sense that a state of M' records state information for M as well as some additional information. We control M by controlling M'; its larger state space contains the necessary information to derive our optimal policy. Furthermore, by the way M and M' are constructed, a policy defined on M' is directly translatable into a policy for M. The optimal policy works in two phases. In the first phase it tries to make M' hit a target state; if this phase succeeds, then it switches to another strategy which ensures with probability one that the history produced is accepted by A.

The technical development involves a detailed analysis of the structure of ω-automata and the interaction with controlled Markov chains. The details are rather involved; we will outline in the following section the basic concepts involved.

A. The Construction of M'

In this section we define some important graph-theoretic concepts necessary for the construction of M'. We can assume without loss of generality that the automaton A has a unique initial state, say s_0, and that it is completely specified, i.e., has a transition from every state on each letter of the alphabet X. We view M also as an automaton with state space X over the same alphabet $\Sigma = X$ which generates infinite words. All states of this automaton are accepting. A transition of M from

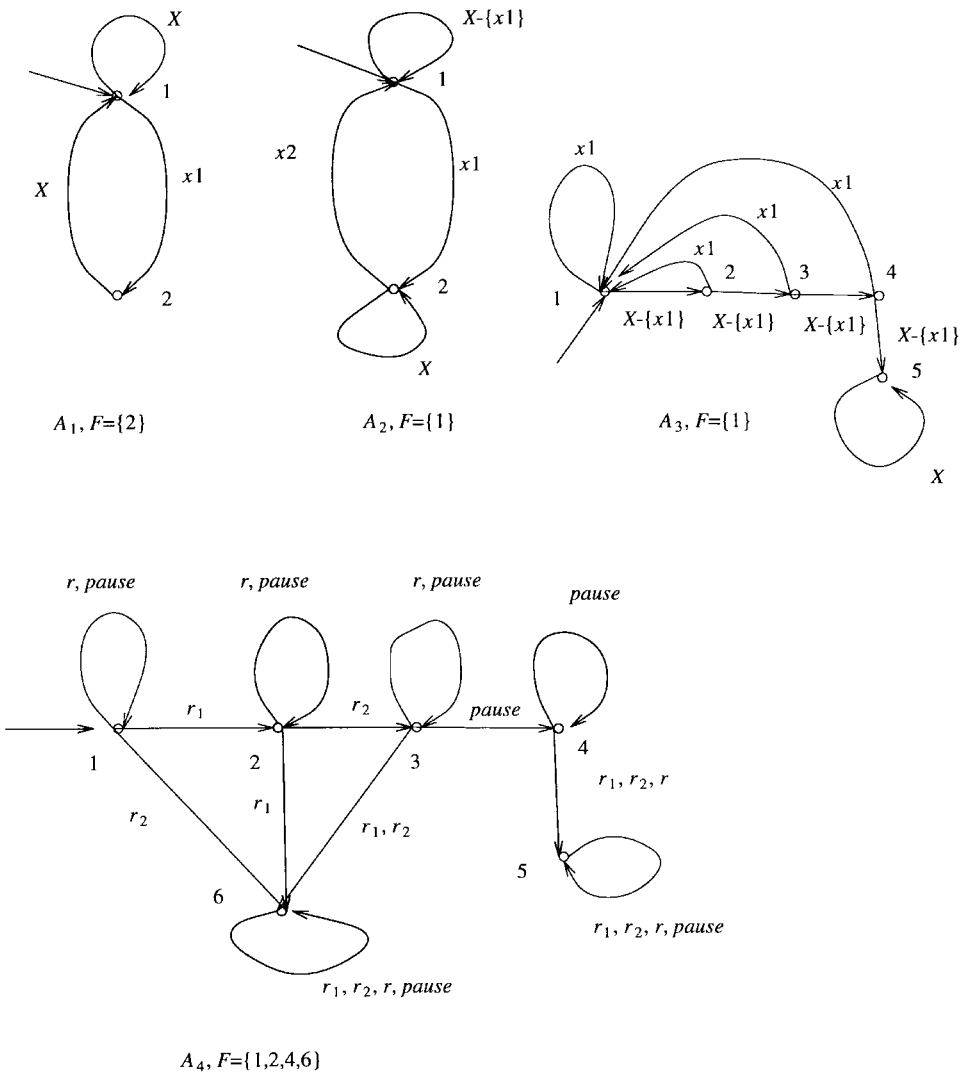

Fig. 1. Example automata.

state v to a state x is labeled with the letter x (the tail of the transition); that is, the transition function ρ_M of M from state v on letter $x \in X$ is $\rho_M(v,x) = \{x\}$ if M has a transition from v to x, and is Ø otherwise. Note that under this definition, M is a deterministic automaton, but it may not be completely specified.

Observe that the above definitions imply that the infinite word (over the alphabet X) generated by a trajectory of M is identical to the trajectory, except that the first state of the trajectory is missing. For this reason, we first add a new dummy initial state x_0 and add a transition from x_0 to every state v with the probability of the transition equal to the probability of v in the (old) initial distribution (of course, if this probability is zero, then the transition $x_0 \to v$ is not included). We want to maximize the probability that the new chain M generates a word accepted by A, starting from x_0.

We can take now the product of M and A as two automata over the same alphabet X. If S is the set of states of A, then the set of states of the product transition table $\tau_{M\times A}$ is $X \times S$. On letter $x \in X$, a state (v,s) of the product goes to all states (x,s') such that $s' \in \rho_A(s,x)$ if M has a transition from v to x, and goes nowhere otherwise. Note that all transitions of $\tau_{M\times A}$ coming into a state with the first component x are on letter x. Every (finite or infinite) run of $\tau_{M\times A}$ corresponds to a unique path in M; just project every state of the run on its first component. Conversely, every path of M corresponds to at least one run of $\tau_{M\times A}$. We consider $\tau_{M\times A}$ as an automaton with initial state (x_0, s_0) and with set of accepting states $X \times F$, where F is the set of accepting states of A. Observe that a trajectory $\mathbf{X} = x_0x_1x_2\cdots$ of M (starting from the new initial state x_0) corresponds to some accepting run of $\tau_{M\times A}$ starting from state (x_0, s_0) if and only if the infinite word $x_1x_2\cdots$ that it generates is accepted by the automaton A. The automaton $\tau_{M\times A}$ accepts the intersection of the languages of M and A. This allows us to reduce the problem of producing a trajectory in M accepted by A into the problem of producing a trajectory accepted by $\tau_{M\times A}$.

Consider the deterministic table $L = \tau_M \times \det(\tau_A)$ obtained by taking the product of τ_M and $\det(\tau_A)$. By construction every state of this deterministic table is a pair (x,Q) that has as first component some state $x \in X$ of the Markov chain and as a second component some set Q of states of the automaton

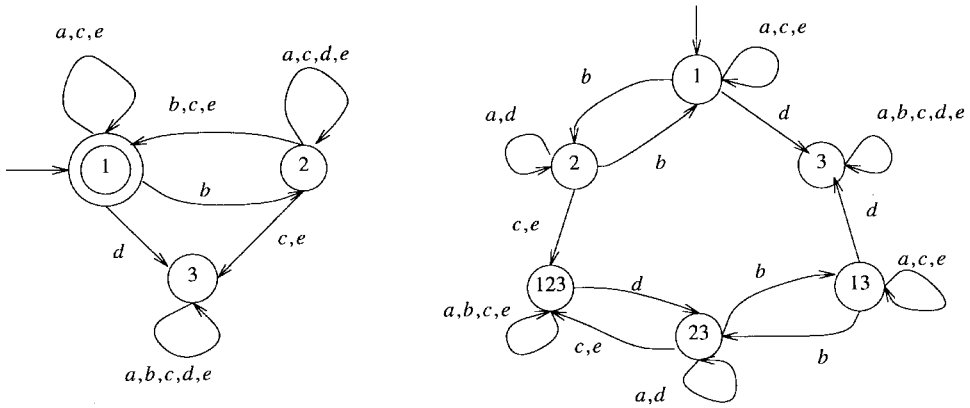

Fig. 2. (a) A Buchi automaton and (b) the corresponding deterministic automaton.

A. Note that this table is defined over all states in $X \times 2^S$. We define the Markov chain M' on this transition table; that is, the states and transitions of M' are those of L. A state (x, Q) of M' is a control state or a randomizing state depending on whether x is a control or randomizing state of M. If x is a randomizing state and has a transition to a state x' in M, then the state (x, Q) of M' has a corresponding transition to a unique state (x', Q') with first component x'; we assign to this transition of M' the same probability as the transition $x \to x'$ of M. The initial state of M' is $(x_0, \{s_0\})$.

Observe that for every state x of M and every state (x, Q) of M' with first component x, there is a one-to-one correspondence between the infinite trajectories $\mathbf{X}$ of M starting at x and the infinite trajectories $\mathbf{Y}$ of M' starting at (x, Q), such that $\mathbf{X}$ is the projection of $\mathbf{Y}$ on the first component; there is a similar correspondence between finite trajectories of the same length. That is, every trajectory $\mathbf{X} = X_0, X_1, \cdots, X_0 = x$, of M has a unique corresponding trajectory $\mathbf{Y} = Y_0, Y_1, \cdots, Y_0 = (x, Q)$, of M' and vice versa: $\mathbf{Y}$ is the run of L over $X_1, X_2, \cdots$ starting from (x, Q) and conversely, for a given $\mathbf{Y}$, the corresponding trajectory $\mathbf{X}$ is simply the projection of $\mathbf{Y}$ on the first component. We say that $\mathbf{X}$ and $\mathbf{Y}$ are *coupled*. We can think of M and M' as having the same sample space, where a sample point corresponds to a pair $(\mathbf{X}, \mathbf{Y})$ of coupled trajectories of M and M' starting at their respective initial states.

Similarly, there is a one-to-one correspondence between policies u in M and policies u' in M'. Recall that a policy maps each finite history (trajectory) ending at a control state to a set of probabilities for the transitions coming out of that state. We only need to consider trajectories starting at the respective initial states of M and M'. If u is a policy in M, then the corresponding policy u' in M' is defined as follows: if $\mathbf{Y}$ is a finite trajectory of M' ending at a control state (x, Q), then u' assigns to an outgoing transition $(x, Q) \to (x', Q')$ the same probability that u assigns to the transition $x \to x'$ given the history $\mathbf{X}$ coupled with $\mathbf{Y}$. Conversely, from a policy u' of M' we can define a corresponding policy u of M as follows: given a finite trajectory $\mathbf{X}$ of M ending at a control state x, if the corresponding coupled trajectory $\mathbf{Y}$ of M' ends in state (x, Q), then we let the policy u assign to a transition $x \to x'$ the same probability that u' assigns given $\mathbf{Y}$ to the unique transition from (x, Q) to a state (x', Q') with first component x'. By construction, corresponding policies of M and M' induce the same probability measure on the common sample space of coupled trajectories $(\mathbf{X}, \mathbf{Y})$ of M and M'.

As we have already mentioned, we will define our control policy in terms of M'. A state of M' summarizes the information from the past necessary for controlling the actual chain M. Hence when our policy produces a trajectory $\mathbf{Y}$ by controlling M', this corresponds to producing the trajectory $\mathbf{X}$ of M, where $(\mathbf{X}, \mathbf{Y})$ are coupled as described before.

In the previous definitions the table L is defined without specifying an initial state. There will be cases in which we will need to define the reachable part of L from a particular initial state (x, Q) of L; in this case $\langle L, (x, Q)\rangle$ will denote the above table. Clearly, this table is smaller than L since it contains the reachable part of L from state (x, Q). We will also use the notation $(x, s) \in y$ where y is a state in L if $y = (x, Q)$ and $s \in Q$. Similarly, we denote by $y \subseteq y'$ the case that $y = (x, Q)$, $y' = (x, Q')$, and $Q \subseteq Q'$.

Example 3.1: Figs. 2–4 illustrate the construction of the controlled Markov chain M' from the chain M. Fig. 2(a) shows a Buchi automaton A over the alphabet $\Sigma = \{a, b, c, d, e\}$. The automaton has three states numbered 1, 2, 3. State 1 is the initial and the only accepting state of A. The automaton is nondeterministic. Fig. 2(b) shows the corresponding deterministic table $\det(\tau_A)$; more precisely, we only show the part of the $\det(\tau_A)$ that is reachable from the initial state 1. Each state of $\det(\tau_A)$ is a set Q of states of A, which is shown in the figure as the concatenation of the elements of Q.

Fig. 3 shows a controlled Markov chain M with set of states $X = \Sigma$. The initial state is a, which is a control state and all the other states are randomizing. In the figure we have not included the dummy initial state x_0. Fig. 4 shows the (table of the) new controlled Markov chain M'. We have labeled each state (x, Q) of M', where $x \in X$ is a state of M and Q is a set of states of A, by the concatenation of x and the elements of Q. States of M' with first component a are control states, and the rest are randomizing. Ignore for now the dotted box in the figure; we will explain its meaning later.

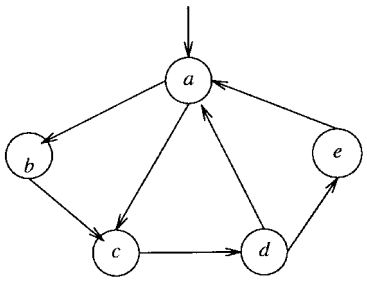

Fig. 3. A controlled Markov chain.

B. The Optimal Strategy

Our optimal strategy will be defined in terms of the following two simple strategies, which are defined for arbitrary controlled Markov chains. Consider any finite sequence of states $\gamma = x(0)\cdots x(m) \in X^*$ with transitions consistent with τ_M. We define first as $\theta(\gamma)$ the *strategy favoring* γ under which the event that the chain starting from state $x(0)$ will perform the sequence of transitions in γ has positive probability. Observe first that some states in γ are randomizing and some are control states. The goal of this strategy is to make γ occur. To achieve this, when the chain is at a control state it chooses the transitions out of this state in a way consistent with γ. Then if the outcome of the randomized transitions turns out to be favorable, γ will be achieved; otherwise the policy will fail. Note that if all states are control states, then $\theta(\gamma)$ will succeed with probability one; if all the states are randomizing states, then the strategy is trivial and has no effect. Note also that the duration of the above strategy is at most m steps and that the probability of success is equal to the product of the probabilities of the randomized transitions in γ. Let $p_\gamma > 0$ denote the above probability.

The second strategy we need to define is the *alternating strategy*. This strategy keeps a history of the transitions taken out of each control state. Every time such a state is visited, an outgoing transition is picked by rotating through the set of possible transitions so that each transition is scheduled in a round-robin way. From the above definition it follows that when the alternating strategy is used, if a control state is visited infinitely often then all transitions out of this state will also occur infinitely often.

To illustrate the difficulty of the general case we consider first the case in which A is deterministic. We remind the reader that our goal is to create a trajectory $\mathbf{X}$ of M for which there exists a run of $\tau_{M\times A}$ which repeats some accepting state of the form (x, f), $f \in F$, infinitely often. Since τ_A is deterministic it follows that for each $\mathbf{X}$ there is a unique corresponding run of $\tau_{M\times A}$ (which is deterministic as well and coincides with the table L defined in the previous section), being equal to the corresponding $\mathbf{Y}$ of M'. Hence if we control M' instead of M so that we maximize the probability that $\mathbf{Y}$ hits some accepting state in $\tau_{M\times A}$ infinitely often, we have solved our initial problem. The value of this maximum probability corresponds to the $\max_u P_u(\mathbf{A})$, and the optimal policy u' for M' is directly translated into the optimal policy u for M; $u(X_0,\cdots,X_n) = u'(Y_0,\cdots,Y_n)$, $n = 1,2,\cdots$, where X_i is the projection of Y_i onto its first component. An optimal policy for the deterministic case can be easily constructed as follows. Let C be a strongly connected subset of states in L reachable from (x_0, s_0), such that there are no randomizing transitions out of C, and C contains some state $y = (x, f)$. Then conditional on the event that Y_t eventually hits a state $y' \in C$, if we use the alternating policy and exclude transitions out of C, then Y_t will hit y infinitely often with probability one (by ergodicity and by the definition of the alternating policy). This observation leads us to the following construction. We define as our target set T the union of the sets of states of all the strongly connected subsets in L satisfying the properties defined previously. Then our optimal policy is performed in two steps. First it uses the policy which maximizes the probability that Y_t eventually hits T; then when a state in such a set C is reached, it switches to the alternating strategy that does not leave C. Note that $\mathbf{Y}$ is the run of $\tau_{M\times A}$ over $\mathbf{X}$, and it repeats (x, f) at the random times $\tau_1, \tau_2, \cdots$, each such time being almost surely finite.

Consider now the case in which τ_A is nondeterministic. A naive approach would be to extend the reasoning of the deterministic case as follows. Consider again M' defined over $L = \tau_M \times \det(A)$ starting from $(x_0, \{s_0\})$, and assume that we find a policy which maximizes the probability that some state (x, Q), such that $f \in Q$, is repeated infinitely often by Y_t. Will this policy solve our optimization problem? The answer is no! The reason is the following. Suppose that $\mathbf{Y}$ repeats (x, Q) infinitely often at times $\tau_1, \tau_2, \cdots$. This does not mean that the corresponding $\mathbf{X}$ has a run of $\tau_{M\times A}$ which repeats (x, f) at the above times $\tau_1, \tau_2, \cdots$. It only implies that for each τ_k, there is a run of $\tau_{M\times A}$ over $X_0, \cdots, X_{\tau_k}$ starting from (x_0, s_0) which *ends* in (x, f), and which does not necessarily pass through (x, f) during the times $\tau_1, \cdots, \tau_{k-1}$. This example must also convince the reader that determinizing Buchi automata is a highly nontrivial task. We return now to the general case.

As a first step we would like to characterize those states of the (nondeterministic) automaton $\tau_{M\times A}$ that can appear infinitely often in runs of $\tau_{M\times A}$ over sets of infinite trajectories of the chain having positive probability for some policy u. These states we call controllably recurrent states. Not all states of $\tau_{M\times A}$ have this property. There are states (x, f) such that no run of $\tau_{M\times A}$ over $\mathbf{X}$ repeats (x, f) infinitely often, and this happens P_u-almost surely, for all policies u. Clearly we are interested for states satisfying the complement of this property. We will show that such states are characterized by some graph-theoretic property. In the following definition we introduce the property of controllable recurrence as a graph-theoretic property, and then we prove that it is equivalent with the infinite repetition property mentioned above. We will show that if (x, s) is controllably recurrent in the graph-theoretic sense, then there exists some policy u which generates with positive probability a $\mathbf{X}$ starting from x for which there is a run of $\tau_{M\times A}$ starting from (x, s) which repeats (x, s) infinitely often.

Definition 3.1: A state $(x, s), x \in X, s \in S$ is *controllably recurrent* if in L there exists a strongly connected subset C with a state $y \in C, (x, s) \in y$, such that:

1) there is no transition out of C corresponding to a randomizing transition in M;

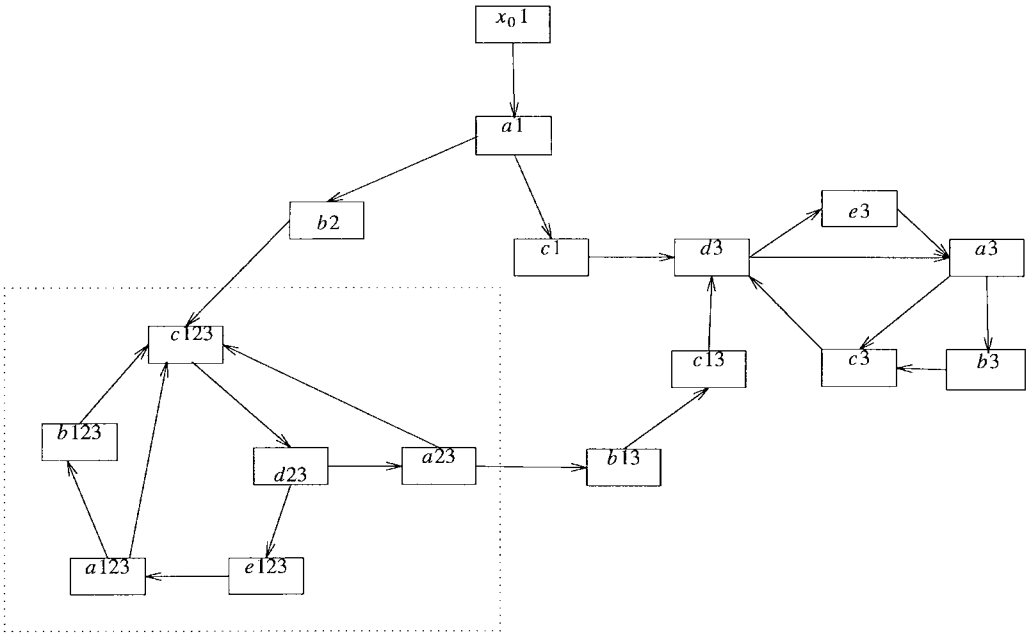

Fig. 4. The new controlled Markov chain derived from the chain of Fig. 3 and the automaton of Fig. 2.

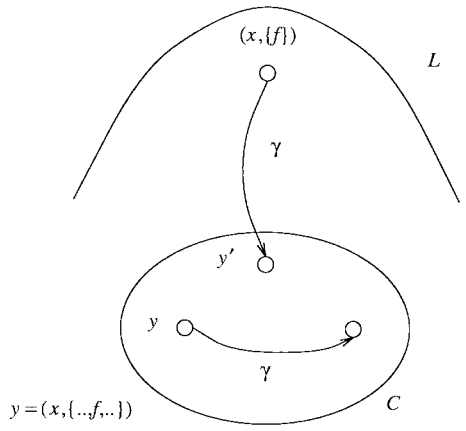

Fig. 5. Controllably recurrent state.

2) there is a nonempty finite string $\gamma \in X^*$ such that the run of L over γ starting at y visits only states in C, and the run of L over γ starting at (x, s) ends inside C. □

The definition for a state to be controllably recurrent is illustrated in Fig. 5.

Example 3.1 (Continued): Consider the example of the automaton A and chain M of Figs. 2 and 3, respectively. Recall that a is the control state of M and the rest are randomizing states. Consider the state $(a, 1)$ of $\tau_{M\times A}$. Fig. 4 includes the part of the transition table L that is reachable from that state (which is labeled a_1 in the figure). The set C of states that lie within the dotted rectangle of the figure is clearly strongly connected. The set C includes a state $y = a123$ (i.e., $y = \{(a, 1), (a, 2), (a, 3)\}$) that contains $(a, 1)$. Condition 1) of the definition is satisfied because the only transition out of C leaves state $a23$, which is a control state. The string $\gamma = bc$ fulfills condition 2) of the definition: the run of L over γ starting from y stays in C and the run starting from $(a, 1)$ ends at state $c123 \in C$. Therefore, $(a, 1)$ is a controllably recurrent state. In contrast, state $(c, 1)$ is not controllably recurrent. □

The string γ associated with a controllably recurrent state (x, s) in the previous definition has the following interesting property as shown in the Appendix: starting from x, if M performs first the transitions in γ, then there is a policy which from then on can complete γ into an infinite trajectory which has with probability one a run of $\tau_{M\times A}$ starting from (x, s) which repeats (x, s) infinitely often. The following proposition summarizes the important properties of controllably recurrent states.

Proposition 3.1: Let (x, s) be any state of $\tau_{M\times A}$, and let $\mathbf{B}$ denote the event that a trajectory of M starting from s_0 has a corresponding run of $\tau_{M\times A}$ starting from (x_0, s_0) that repeats (x, s) infinitely often. There is a policy u such that $P_u(\mathbf{B}) > 0$ if and only if (x, s) is controllably recurrent and is reachable from the initial state (x_0, s_0) of $\tau_{M\times A}$. □

The proof of this proposition uses the previous remark concerning the properties of the controllable recurrent states together with the fact that if (x, s) is controllably recurrent and reachable in $\tau_{M\times A}$ from (x_0, s_0), then there is a policy which with positive probability constructs an initial prefix $\delta\gamma$ of $\mathbf{X}$ with the property that 1) there exist a run of $\tau_{M\times A}$ over δ starting from (x_0, s_0) which ends in (x, s) and 2) γ is the "favorable" string of (x, s) in the previous definition.

A corollary of this proposition is that the maximum probability $\max_u P_u(\mathbf{A})$ that M generates a trajectory in $\mathbf{A}$ is positive if and only if the initial state (x_0, s_0) of $\tau_{M\times A}$ can reach an accepting controllably recurrent state (x, f) (i.e., with $f \in F$). We define now the set T of target states of M'.

Definition 3.2: If (x, f) is an accepting controllably recurrent state of $\tau_{M\times A}$ reachable from (x_0, s_0), then let $V(x, f)$ be the set of those states of L which belong to some strongly connected subset C such that the conditions of Definition 3.1 are satisfied for (x, f) and C. Let V be the union of the sets $V(x, f)$ for all accepting controllably recurrent states (x, f). A state z of M' is a *target state* iff there is a member z' of V such that $z' \subseteq z$. □

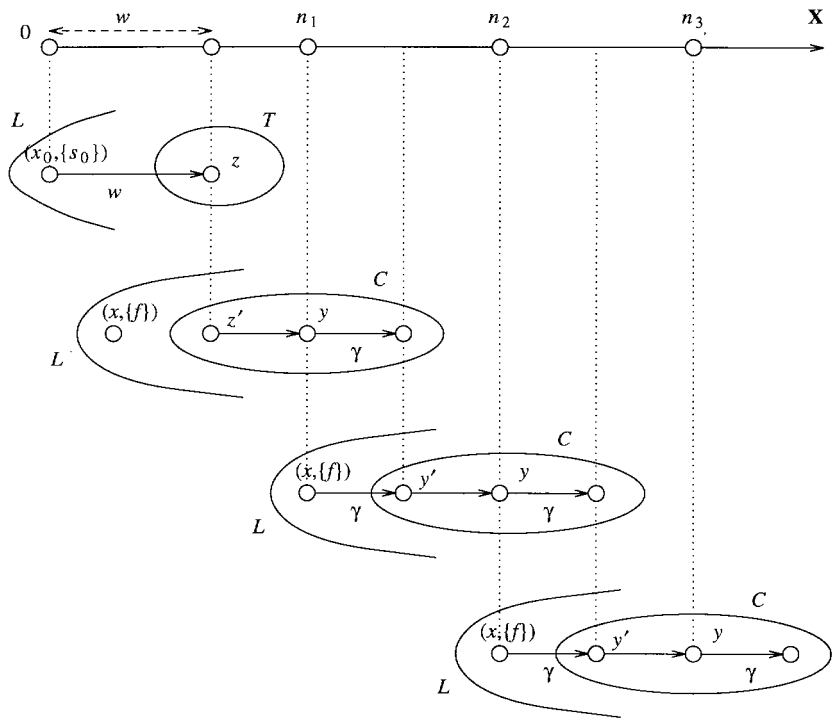

Fig. 6. The strategy of Lemma 3.1.

An important property of such a state z is that it offers at least as many possibilities for controlling the future of M' as a state $z' \subseteq z$. Basically, any sequence of transitions starting from z' can be performed by starting from z while preserving the $\subseteq$ ordering between the corresponding states of the two trajectories. Hence being at state z we have all control capabilities corresponding to a state $z' \subseteq z$.

Example 3.1 (Continued): State $(a, 1)$ is an accepting controllably recurrent state of $\tau_{M\times A}$ with the corresponding strongly connected subset C within the dotted rectangle satisfying Definition 3.1. This is the only such subset. Therefore, $V(a, 1) = C$. If we restrict attention only to those states of L that are reachable from the initial state, the set of target states coincides with C. There are more target states that are unreachable and thus do not interest us; for example $d123$ is such a target state because it contains $d23$. □

The following two lemmas give the important properties of the set T of target states. The first lemma says that if a policy manages to make the chain M' hit a target state, then it can switch to a strategy that almost certainly ensures a good trajectory. The reason this works is the following. If a target state is reached, then there is a policy which with probability one ensures that eventually the initial prefix of the trajectory $\mathbf{X}$ will become of the form $\delta\gamma$, where δ and γ are related with some controllable recurrent state (x, f) as in the remark following Proposition 3.1.

Lemma 3.1: Let $w = x(1)\cdots x(n)$ be any finite sequence of states of M for which the run of L starting from $(x_0, \{s_0\})$ ends in some state in T. Then, if the Markov chain M performs as its first n transitions the transitions corresponding to w, there exists a strategy μ which is used from time n onwards and produces with probability one an infinite trajectory accepted by the automaton A (this trajectory has w as its prefix). □

We describe now the strategy μ which is constructed in the proof of Lemma 3.1 and combines the simple strategies introduced earlier. Assume that the run of L starting from $(x_0, \{s_0\})$ over $x(1)\cdots x(n)$ ends in state $z \in T$, and let z', y, (x, f), C, γ be the corresponding entities (with z) according to Definitions 3.1 and 3.2. (Note that $z' \subseteq z$, $z', y \in C$; see Fig. 6). Suppose also that the run of L over γ starting from $(x, \{f\})$ ends in state $y' \in C$.

The strategy μ will be defined in terms of controlling the evolution of M' (and hence the evolution of M since M' refines M) so that with probability one the trajectory with initial prefix $x(1)\cdots x(n)$ will be accepted by A, and its construction is depicted in Fig. 6. In order to achieve this we must control the way the remaining part of the trajectory will be constructed. To do this, we start M' in state $z' = (x(n), Q) \in C$, and we control the chain so that eventually 1) state $y \in C$ is reached and 2) the next $|\gamma|$ transitions performed immediately following y constitute the string γ; this can be achieved with probability one in finite time since the component C has no outgoing randomizing edges. Assume now that at the time we visited y prior to the appearance of γ was n_1 (time is measured from the beginning of the complete trajectory), and that at this time we started a new copy M'_1 of M' on the same probability space, with initial state $(x, \{f\})$. Now, at time $n_1 + |\gamma|$, M'_1 will be in state $y' \in C$. We now forget M' and we control from this time on M'_1 until a similar condition as in the case of M' is met: state $y \in C$ is reached and the transitions immediately following y constitute the string γ. Let n_2 be the time y was visited by M'_1. At this time we start an other copy M'_2 from state $(x, \{f\})$, and we

keep repeating this procedure forever. One can easily convince oneself that the trajectory $\mathbf{X}$ constructed by this procedure has with probability one a run in $\tau_{M\times A}$ starting from (x_0, s_0) which repeats (x, f) at the times $n_1, n_2, \cdots$.

The policy is formally defined as follows. The states y, y', z' have been introduced previously and correspond to the initial prefix $w = x(1)\cdots x(n)$ of the trajectory which satisfies the conditions of Lemma 3.1.

Start M' at state z'.

Step 0: Use the alternating strategy (on M') by avoiding control transitions out of C until state y is reached; then go to step 1.

Step 1: Use the strategy favoring γ; if this strategy fails, go to Step 0, else restart M' at state y' and go to Step 0.

Example 3.1 (Continued): Consider the policy with state a_1 in the role of the controllably recurrent state (x, f), state $a123$ in the role of y, and state $c123$ in the role of z, z' and y'. When the chain M' is in state $y = a123$, then (Step 1) the policy uses the strategy favoring γ and thus chooses the transition of M' out of state $a123$ that goes to $b123$ (i.e., the transition of the original chain M from a to b). When the chain M' is in state $a23$, then (Step 0) the policy uses the alternating strategy and thus chooses the transition that goes to $c123$ in order to stay within the subset C, i.e., the policy chooses the transition of the original chain M from a to c. Note that the latter choice is always "safe" at state $a123$ as well. However, if the policy was always choosing that transition, then it would not generate a word accepted by the automaton A. □

For the purposes of this section, we need only the converse of Lemma 3.1; i.e., under any policy u, the trajectories of M that are accepted by A almost surely correspond to trajectories of M' that hit T. The following stronger converse is true and will be needed in the general case of many events.

Lemma 3.2: Let $\mathbf{X}$ be a trajectory of M starting from x_0, and let $\mathbf{Y}$ be the corresponding trajectory of M' starting from $(x_0, \{s_0\})$. Then, for any policy u, the probability that $\mathbf{X} \in \mathbf{A}$ and that $\mathbf{Y}$ does not hit some state in T infinitely often is equal to zero. □

We can define now the optimal policy as follows: first use a strategy which maximizes the probability that the chain M' eventually hits the set T. When a trajectory of M' hits for the first time some state in T, then the policy switches to the strategy μ defined in Lemma 3.1 from this time onwards. The optimality of such a policy can be easily deduced from Lemmas 3.1 and 3.2 as follows.

For any policy u, let $\zeta(T)$ be the first time that the trajectory $\mathbf{Y}$ of M' hits T (recall that M' is defined on the same probability space as M). From Lemma 3.2 it follows that $P_u(A) \le P_u(\zeta(T) < \infty)$. For any policy π and its corresponding stopping time $\zeta(T)$, define $u(\pi)$ to be the policy which imitates π until the stopping time $\zeta(T)$ (forever if $\zeta(T) = \infty$) and then uses strategy μ from that time on. It follows from Lemma 3.1 that for any policy π we have $P_{u(\pi)}(\mathbf{A}) = P_\pi(\zeta(T) < \infty)$. Combining this with the previous inequality, we have the following optimality theorem.

Theorem 3.1: If π^* is a policy that maximizes $P_\pi(\zeta(T) < \infty)$ over all policies π, then the corresponding policy $u(\pi^*)$ maximizes $P_u(\mathbf{A})$. □

The maximum probability $p^* = \max_u P_u(\mathbf{A})$ and the actions of the optimal policy during the first phase can be computed from the solution of a linear program constructed as follows. Let Y be the set of states of the chain M', and let $Y_N \subseteq Y$ denote the subset of the control states, $E(i)$ be the set of states to which there is a transition from state i, and $V(i)$ be the maximum probability over all policies that the chain starting from i will eventually hit T. We define the linear program LP1:

$$\min \sum_{i\in Y} v(i)$$

$$\begin{aligned}
&v(i) \ge \sum_{j\in E(i)} p_{ij} v(j), \qquad i \in Y - Y_N, \quad i \notin T\\
&v(i) \ge v(j), \qquad j \in E(i), \qquad i \in Y_N, \qquad i \notin T\\
&v(i) \ge 0, \qquad i \in Y\\
&v(i) = 1, \qquad i \in T.
\end{aligned}$$

Proposition 3.2: V is the optimal solution of LP1. In particular, $\max_u P_u(A)$ is given by the value of $v(x_0, s_0)$ in the optimal solution. □

In Example 3.1 we may choose the transition out of state a_1 going to state $b2$. From that point on we can apply the previously described policy to ensure acceptance with probability one.

IV. The Optimization Problem for Many Events

We will first analyze the case that the events (the desired properties) are disjoint. This case contains most of the probability theoretic subtleties of the problem. Then we will proceed to the general case which requires further structural analysis of automata. The optimal policies of this section operate also on a new controlled Markov chain with a larger state space and work in two phases. In the first phase the policy solves an optimal stopping time problem; when this phase finishes, the policy has decided on a subset of properties that it can probably achieve, and has given up on the rest of them. In the second phase, the policy switches to a strategy that ensures that the properties in the subset are satisfied with probability one. It is critical that we can stop the first phase after finite expected duration, even though we deal with properties that depend on the entire infinite history, i.e., observing the system for any finite time may not be enough to tell whether it will satisfy a property or not.

A. Disjoint Events

We are given a controlled Markov chain M', automata A_i, $i = 1, \cdots, m$, corresponding to m disjoint regular sets and the positive scalars $k_1 \le \cdots \le k_m$. We want to find a policy which maximizes the reward $R_u = \sum_i k_i P_u(\mathbf{A}_i)$.

For each automaton A_i let S_i be its state space and s_0^i its initial state. Let L be the transition table corresponding to the product of the tables $\tau_M, \det(\tau_{A_1}), \cdots, \det(\tau_{A_m})$. Clearly L is

also deterministic and its states can be written in the form $z = (x, w_1, \cdots, w_m)$, $w_i \in 2^{S_i}$, with the interpretation that if the run of L over some word a starting from $(x_0, \{s_0^1\}, \cdots, \{s_0^m\})$ ends on z, then for each i, the run of $L_i = \tau_M \times \det(A_i)$ over a will end on (x, w_i). We define the projection on the ith component of the state z of L to be the state (x, w_i) of the corresponding table L_i. We let M' be the controlled Markov chain defined on the transition table L by associating the transition probabilities of M as we did in the case of one automaton in the previous section. We consider M and M' as being defined on the same probability space. When we refer to the reward R_u of a policy in M' or deal with the probability of events $P_u(\mathbf{A}_i)$, since these events are in the probability space of the original chain M, we implicitly project the trajectories of M' on the component of the state corresponding to M.

Let Y be the state space of M'. We construct the sets $T_i \subseteq Y$, $i = 1, \cdots, m$, such that a state z of M' is in T_i if the projection of z on its ith component is one of the target states of the table L_i corresponding to M and A_i as defined in the previous section. Although the events are disjoint, it may be the case that some states have their projections on two or more components belonging to the target sets of the corresponding tables; we make the T_i's disjoint by associating any state in Y which belongs to more than one T_i's by the previous construction, to the T_i with the largest index (i.e., the one with the largest reward).

We define the following two systems: O_1, O_2. The first system O_1 is the original system consisting of the chain M' and the cost structure $R_u^1 = \sum_i k_i P_u(\mathbf{A}_i)$. A policy in this system will be denoted by u and Y_t denotes the state of the chain at time t.

The second system O_2 consists of a different controlled Markov chain M'' and a new cost structure defined as follows. The new chain is identical to M' with the addition of the m absorbing states $y_1^*, \cdots, y_m^*$, and at each state in T_i we add a new decision for transitioning to y_i^*. In this system we associate k_i units of reward with the transitions $y \to y_i^*$, $y \neq y_i^*$, $i = 1, \cdots, m$, and zero reward with all remaining transitions. This implies that on every trajectory, reward will be produced only once, and this will occur when the system will for the first time transition to some state in $y_1^*, \cdots, y_m^*$. A policy in this system is denoted by π and Z_t denotes the state of the chain at time t. The reward structure in this system is the total expected reward obtained during the infinite execution of the system.

As we will show, an optimal policy in O_2 gives rise to an optimal policy in O_1. For each policy π of O_2, let ζ_π be the stopping time (in O_2) corresponding to the first time Z_t visits any state in the set $\{y_1^*, \cdots, y_m^*\}$. Define the policy $u(\pi)$ in O_1 as follows. For $t < \zeta_\pi$, the policy $u(\pi)$ imitates π; if π visits state y_i^*, then for $t \geq \zeta_\pi$, $u(\pi)$ uses strategy μ_i, as defined before. We shall show that if π^* is an optimal policy in O_2, then $u(\pi^*)$ is an optimal policy for O_1.

Lemma 4.1: For all policies π of O_2 we have $R_{u(\pi)}^1 \geq R_\pi^2$.

To prove a converse, we will argue that given any policy u in O_1 we can construct a policy π in O_2 which produces expected rewards arbitrarily close to the reward produced by u. First, we need to introduce some new quantities. Let $\mathbf{Y}_s$ denote the history $Y_0, \cdots, Y_s$, and let $T_{>i} = \cup_{j>i} T_j$ ($T_{>m} = \emptyset$). For every policy u we define the function $g_u(\mathbf{Y}_s) = \sum_i k_i P_u$ (after time s, $\mathbf{Y}$ will hit eventually some state in T_i and never visit any state in $T_{>i} \mid \mathbf{Y}_s$). Using the analysis of the previous section we can show that the reward is bounded from above by g_u applied to the empty history.

Lemma 4.2: Let s be an arbitrary time. Then the following holds:

$$\begin{aligned}
&\sum_i k_i P_u(\mathbf{A}_i) \\
&\quad \leq \sum_i k_i P_u(\mathbf{Y} \text{ hits i.o. } T_i \\
&\qquad\qquad \text{and does not hit i.o. } T_{>i}) \\
&\quad \leq \sum_i k_i P_u(\text{after time } s, \mathbf{Y} \text{ hits} \\
&\qquad\qquad \text{eventually } T_i \text{ and never again } T_{>i}).
\end{aligned}$$

The same inequalities also hold if we consider conditional probabilities on some arbitrary event. □

Given any positive scalar ε we construct the stopping time ψ of Y_t as follows. For every time s during which M' visits a state in $\cup_i T_i$, we compare $g_u(\mathbf{Y}_s)$ with $k_i + \varepsilon$, where i is such that $Y_s \in T_i$. If $g_u(\mathbf{Y}_s) < k_i + \varepsilon$ we define $\psi = s$ and stop. If no such s exists, then $\psi = \infty$. We call such a $\mathbf{Y}_\psi$ a *stopping history*. We define now the policy $\pi(u, \varepsilon)$ in O_2. This policy is identical to u up to time ψ. Then at time ψ it makes Z_t jump to the state y_i^* where i is such that $Y_\psi \in T_i$.

Lemma 4.3: For any policy u in O_1 and any positive scalar ε, we have $R_{\pi(u,\varepsilon)}^2 \geq R_u^1 - \varepsilon$. □

We can prove now the optimality theorem.

Theorem 4.1: If π^* is a policy which maximizes R_π^2 in O_2, then the policy $u(\pi^*)$ maximizes R_u^1 in O_1. □

The policy π^* and the optimal reward can be computed by solving the following Linear Program LP2:

$$\min \sum_{i \in Y} v(i)$$

$$\begin{aligned}
&v(i) \geq \sum_{j \in E(i)} p_{ij} v(j), \qquad i \in Y - Y_N \\
&v(i) \geq v(j), \qquad j \in E(i), \qquad i \in Y_N \\
&v(i) \geq 0, \qquad i \in Y \\
&v(i) \geq k_j, \qquad j \in T_i.
\end{aligned}$$

Proposition 4.1: V is the optimal solution of LP2. In particular, the optimum reward is given by the value in the optimal solution of the variable corresponding to the initial state of M'. □

B. The General Case

We are given the sets Ai, $i = 1, \cdots, m$, in terms of their corresponding automata, and the nonnegative constants k_R, $R \subseteq \{1, \cdots, m\}$, satisfying (2). In principle we could solve the general case by reducing it to the disjoint case: for every subset R of indexes we could have a property $\mathbf{A}_R$ as in (1) with associated reward k_R. Since the $\mathbf{A}_R$'s are obviously regular disjoint events, we could then use the method of the

previous subsection. However, this approach would increase drastically the complexity: there are 2^m new properties, and for each one of them we would have to construct an accepting automaton; this would involve complementing some of the original automata and taking their product. In this subsection we will explore further the structure of automata to solve the general case with essentially the same complexity as the disjoint case.

Let L be the deterministic transition table introduced in the previous section, corresponding to the product $\tau_M \times \det(\tau_{A_1}) \times \cdots \times \det(\tau_{A_m})$, and let M' be the corresponding Markov chain.

Definition 4.1: A strongly connected subset C of the table L *favors* the set $\mathbf{A}_R$, $R \subseteq \{1, \cdots, m\}$, if the following conditions hold.

1) There is no randomizing transition out of C.
2) For each $i \in R$, there exists a state $y^i = (x, y^i_1, \cdots, y^i_m)$, of C, an accepting state f^i of the automaton A_i, and a nonempty word $w^i \in X^*$, with the following properties:
   a) $f^i \in y^i_i$;
   b) the run of L over w^i starting from y^i stays in C;
   c) the run of L over w^i starting from $(x, y^i_1, \cdots, \{f^i\}, \cdots, y^i_m)$ ends inside C. ($(x, y^i_1, \cdots, \{f^i\}, \cdots, y^i_m)$ is the state which is identical to y^i except on the ith component, which is $\{f^i\}$). □

We will say that a state $z = (x, z_1, \cdots, z_m)$ of M' *contains* another state $z' = (x, z'_1, \cdots, z'_m)$ if $z'_i \subseteq z_i$, $i = 1, \cdots, m$, and we denote this by $z' \subseteq z$. We define now the sets T_R of states of M' as follows. For each $R \subseteq \{1, \cdots, m\}$, T_R is the set of all states z of L such that z contains some state of a strongly connected subset favoring $\mathbf{A}_R$. A state may belong to more than one set T_R. The important fact is that the sets T_R have properties similar to those of the target sets T_i of Section 4.1. The following two lemmas generalize appropriately Lemmas 3.1 and 3.2.

Lemma 4.4: Let $\gamma = x(1) \cdots x(n)$ be any finite sequence of states of M for which the run of L starting from $(x_0, \{s^1_0\}, \cdots, \{s^m_0\})$ ends in some state in T_R. Then, if the Markov chain performs as its first n transitions the transitions corresponding to γ, there exists a strategy μ_R which is used from time n onwards and produces with probability one an infinite trajectory in the set $\cup\{\mathbf{A}_Q \mid R \subseteq Q\}$ (this trajectory has γ as its prefix). □

The next lemma generalizes Lemma 3.2.

Lemma 4.5: Let $\mathbf{X}$ be a trajectory of M starting from x_0, and let $\mathbf{Y}$ be the corresponding trajectory of M' starting from $(x_0, \{s^1_0\}, \cdots, \{s^m_0\})$. Then, for any policy u, the probability that $\mathbf{X} \in \mathbf{A}_R$ and that $\mathbf{Y}$ does not hit some state in T_R infinitely often is equal to zero. □

We can repeat now the same construction we did in the previous section and use the sets T_R, $R \subseteq \{1, \cdots, m\}$, instead of the sets T_i, $i = 1, \cdots, m$. In this construction we associate with each set T_R an absorbing state y^*_R, and the transition to this state produces reward k_R. We define a system O_2 as in Section 4.1 and solve the optimal stopping time problem for this system. If π^* is the optimal policy for O_2, then the optimal policy $u(\pi^*)$ imitates π^* until it reaches some state y^*_R, and then switches to the corresponding strategy μ_R of Lemma 4.4. The properties of the T_i's used in the proofs of the previous section is that these sets satisfy Lemmas 3.1 and 3.2. By using Lemmas 4.4 and 4.5 instead of Lemmas 3.1 and 3.2 we can carry out a similar analysis and prove that the optimality Theorem 4.1 holds in the general case as well. We can compute the maximum reward and the optimal policy that achieves it by solving a Linear Program. Let Y be the set of states of M', let $Y_N \subseteq Y$ be the subset of the control states, let $E(i)$ be the set of states to which there is a transition from state i, and let $V(i)$ be the maximum reward that can be achieved over all policies when the chain starts from state $i \in Y$. Let LP3 be the following linear program:

$$\min \sum_{i \in Y} v(i)$$

$$\begin{aligned}
v(i) &\geq \sum_{j \in E(i)} p_{ij} v(j), \qquad i \in Y - Y_N \\
v(i) &\geq v(j), \qquad j \in E(i), \qquad i \in Y_N \\
v(i) &\geq 0, \qquad i \in Y \\
v(i) &\geq k_R, \qquad i \in T_R.
\end{aligned}$$

Proposition 4.2: V is the optimal solution of LP3. In particular, the optimum reward is given by the value in the optimal solution of the variable corresponding to the initial state of M'. □

V. Complexity

We analyze first the case of one automaton (Section III). We need to determine first the accepting controllably recurrent states, i.e., states (x, f) of $\tau_{M \times A}$ with $f \in F$ that satisfy Definition 3.1. From them we can determine the set T of target states using Definition 3.2 and then construct and solve the linear program LP1.

We can determine the states (x, f) that are controllably recurrent as follows. Let us say that a subset C of nodes of $L = \tau_M \times \det(\tau_A)$ is *closed* if it is strongly connected, nontrivial (i.e., contains at least one edge), and it satisfies condition 1) of Definition 3.1 (i.e., there is no transition out of C corresponding to a randomizing transition of M). First, observe that if C_1 and C_2 are two closed subsets of L with a nonempty intersection, then their union $C_1 \cup C_2$ is also closed: it is strongly connected because there is at least one node in the intersection $C_1 \cup C_2$ and such a node can reach and can be reached by every node of C_1 and C_2 (since C_1 and C_2 are strongly connected), and $C_1 \cup C_2$ has no outgoing transitions corresponding to randomizing transitions of M because C_1 and C_2 do not have such transitions. Thus, for every state z of L, either there is no closed subset that contains z, or there is a unique such maximal closed subset $C(z)$. Note that if the strongly connected subset C and the state y of L satisfy condition 2) in Definition 3.1, then the maximal subset $C(y)$ and y also satisfy this condition. The maximal closed sets of L are disjoint; we call them the *closed components* of L.

We can determine the set $C(z)$ for every state z of L as follows. Take each strongly connected component D of L. If

D is trivial then delete it. If a state of D whose first component is a randomizing state of M has a transition out of D, then remove the node from D. As a result of this, D may now break into smaller strongly connected components. Continue this process on every new s.c.c., until the final s.c.c.'s satisfy condition 1). These are the closed components of L. Delete all the edges that connect different components. Let $\tilde{L}$ be the final transition table (graph). Clearly, if a state z of L is in this final graph, say in s.c.c. B, then $C(z) = B$; if the state z is removed during this process, then $C(z)$ is undefined [there is no nontrivial strongly connected subset C that contains z and satisfies condition 1)].

Recall that $L = \tau_M \times \det(\tau_A)$ is a deterministic table over the alphabet X (the set of states of M) and if a state (v, Q) has a transition on letter x, then the transition goes to a state (x, Q') with first component x, i.e., the same as the letter. The table $\tilde{L}$ is a subtable of L. Thus, if we form the product of L and $\tilde{L}$, the only states that have incoming transitions are those obtained by combining states of L and $\tilde{L}$ that have the same state of M as their first component. Let Π be the restriction of $L \times \tilde{L}$ to this set of states. We can identify such a pair of states $(v, Q_1), (v, Q_2)$ of L and $\tilde{L}$ having the same first component v with the triple (v, Q_1, Q_2). Thus, in other words, Π has $X \times 2^S \times 2^S$ as its set of states and has a transition from state (v, Q_1, Q_2) to state (x, Q_3, Q_4) (on letter x) iff L has a transition from (v, Q_1) to (x, Q_3) and $\tilde{L}$ has a transition from (v, Q_2) to (x, Q_4).

We claim that a state (x, s) of $\tau_{M \times A}$ is controllably recurrent if and only if there is a set Q that contains s such that Π contains a nontrivial path (i.e., with at least one edge) from the state $(x, \{s\}, Q)$ to a state (w, Q_3, Q_4) where (w, Q_3) and (w, Q_4) are in the same closed subset of $\tilde{L}$. For the (only if) direction, suppose that (x, s) is controllably recurrent. Then there is a closed set C of L, a state $y = (x, Q)$ of C with $s \in Q$, and a (nonempty) word γ which takes L from $(x, \{s\})$ to some state of C, say (w, Q_3) (where w is the last letter of γ), and the path of L on γ from y stays in C and ends say in state (w, Q_4). Since the path of L on γ from y stays in C, the path is also present in $\tilde{L}$. Therefore, the product Π has a path on γ from state $(x, \{s\}, Q)$ to state (w, Q_3, Q_4). Conversely, if Π has a path as in the claim, we let in Definition 3.1 γ be the word of the path, $y = (x, Q)$, and C the closed subset of $\tilde{L}$ that contains (w, Q_3) and (w, Q_4). Then γ takes L from $(x, \{s\})$ to (w, Q_3), a state of C, and it takes $\tilde{L}$ from $y = (x, Q)$ to (w, Q_4), also a state of C. Since $\tilde{L}$ does not have any transitions exiting closed sets, state y and all the states along the path to (w, Q_4) must be in the same closed subset, which must be therefore also C. Thus, Definition 3.1 is satisfied and (x, s) is controllably recurrent.

Thus, once we form Π, we can find all the states (x, s) that are controllably recurrent by computing all the states of Π that can reach a state in the set $\{(w, Q_3, Q_4) : C((w, Q_3)) = C((w, Q_4))\}$. This can be done in linear time in the size of Π by standard graph searching algorithms [2]. If we let v_M and e_M be the number of states (vertices) and transitions (edges) of the controlled Markov chain (MDP) M, and similarly v_A and e_A are the number of states and transitions of the automaton A, then Π has $v_M \cdot 4^{v_A}$ states and $e_M \cdot 4^{v_A}$ transitions.

Once we compute the accepting controllably recurrent states, we can use Definition 3.2 in a straightforward way to compute the set of target states in linear time in the size of L. In the linear program LP1 the variables correspond to the states of the chain M', (i.e., the states of the table L), thus, the number of variables is $v_M \cdot 2^{v_A}$. The constraints correspond to the states and transitions of M', thus the number of constraints is $O(e_M \cdot 2^{v_A})$. The coefficients of the LP are transition probabilities of the given MDP M and 0, 1. The LP can be solved in time polynomial in its size, thus, in time polynomial in the size of the given MDP M and exponential in the size of the given automaton A.

We consider now the case of many automata (Section IV). The algorithm and the complexity analysis is similar. In this case L is the product of τ_M and the transition tables $\det(\tau_{A_i})$ of the determinizations of the given automata A_i, $i = 1, \cdots, m$. We compute the closed subsets of L as before, yielding the table $\tilde{L}$. We can form again the table Π in the same way by pairing the states of L and $\tilde{L}$ with the same state of M as their first component. Actually, we only need to pair states that agree in all the components except possibly for one component corresponding to one of the automata. If y, z are two such states of L and $\tilde{L}$, we will use (y, z) to denote the corresponding state of Π.

We can show along similar lines that a strongly connected subset C of L favors the set $\mathbf{A}_R$, $R \subseteq \{1, \cdots, m\}$ (Definition 4.1) if and only if C is closed, and for each $i \in R$, the table Π contains a nontrivial path from a state (y^i, z^i) to a state (u^i, v^i), where u^i and v^i are both in C, and y^i and z^i agree in all the components except for the one corresponding to $\det(\tau_{A_i})$ which in y^i is equal to $\{f^i\}$ for some accepting state f^i of the automaton A_i, and in z^i it is equal to some set that contains f^i. We can determine for each closed component C of L, the maximal set $\mathbf{A}_R$ that it favors by computing the set H_C of states of Π that can reach (in one or more steps) states of the form (u, v) with $u, v \in C$. Note that these sets H_C for different closed components C are disjoint because different closed components are completely disconnected in $\tilde{L}$. We can compute all the sets H_C in linear time in the size of Π by a standard graph searching algorithm. We examine all the states in each H_C to determine the set $\mathbf{A}_R$ that C favors. We can determine then in a straightforward way, for each state of L (and the chain M') the sets T_R that contain it, and thus the maximum reward k_R over all these sets. We can then construct and solve the linear program LP3. Note that we need to include only one constraint of the form $v(i) \geq k_R$ for each state i of M', namely the one with the maximum right-hand side, i.e., the maximum k_R with $i \in T_R$. The variables of the LP correspond again to the states of L, and the constraints correspond to the states and the transitions. We measure the size n of the automata by the sum of their number of states and transitions. We have thus the following theorem.

Theorem 5.1: The optimization problem for an MDP and a set of automata can be solved in time polynomial in the size of the MDP and exponential (c^n for some constant c) in the size of the automata. □

VI. Extensions

We can extend the definition of a controlled Markov chain in Section II to include the notion of fairness in how certain control transitions are chosen by the policy. In order to do so we need to define the set $N_F \subseteq N$ of *fair* states. We define a policy of a controlled Markov chain as being *fair* if the following condition holds: for any infinite trajectory of the controlled Markov chain, if a fair state $w \in N_F$ appears infinitely often, then all possible transitions out of w are taken infinitely often in the above sequence.

We can use the ideas developed in the previous sections to solve the similar problems under some fairness conditions. We must first extend the definition of a controllably recurrent state as follows.

Definition 6.1: A state $(x,s), x \in X, s \in S$ is *controllably recurrent in a fair sense* if in L there exist a strongly connected subset C with a state $y \in C, (x,s) \in y$, such that:

1) there is no transition out of C corresponding to a randomizing transition in M;
2) there is a nonempty finite string $\gamma \in X^*$ with the following properties: the run of L over γ starting at y visits only states in C, and the run of L on γ starting at $(x,\{s\})$ ends inside C;
3) for any fair state $x \in N_F$ with transitions $(x,x'_1),\cdots,(x,x'_n)$ the following holds: if there exist a state z in C such that the first component of z is x ($z = \{(x,r) \mid r \in Q\}$ for some $Q \subseteq S$), then for each transition (x,x'_i), $i = 1,\cdots,n$, there must exist a transition (z_i,z'_i) in C ($z_i, z'_i \in C$), such that the first component of z_i is x and the first component of z'_i is x'_i. □

Corollary 6.1: There exists a fair policy u for which $P_u(\mathbf{A}) > 0$ if and only if there exists a $(x,f), x \in X, f \in F$, such that (x,f) is controllably recurrent in the fair sense and reachable in $\tau_{M\times A}$ from (x_0,s_0). Furthermore, if we restrict the space of policies to the set of fair policies, all the results in Section IV hold with the sets T_i being defined in terms of the states (x,f) being controllably recurrent in the fair sense, as defined in Definition 6.1. □

The proof of the above corollary is similar to the proofs in Section IV and is omitted.

VII. Discussion and Open Problems

An interesting observation is that the difficulty of the optimization problem is largely due to the fact that the ω-regular sets are represented by nondeterministic automata. One can easily see that if we are restricted to sets represented by deterministic Buchi automata, the optimization problem simplifies to a large extent and the complexity becomes polynomial in the size of the automata. Unfortunately, not all ω-regular properties can be represented by such deterministic automata. On the other hand, one could determinize such automata (see [26]) and obtain equivalent deterministic automata with acceptance conditions of the type introduced by Rabin (see [21]). The shortcoming of the above method is that the above acceptance conditions are not as simple as the ones we use in this paper and our approach will not work in this case. Similarly, the semideterminization approach followed in [7] could be proved useful.

An interesting extension of our model is to allow for a more general definition of a valuation mapping (see Section II). We could have that $V : X \to \Sigma \cup \{\varepsilon\}$, where ε is the empty string. This allows us to model invisible transitions by the system. One can easily see that the remark in the end of Section II-B still holds, i.e., we can construct an automaton A' for which we can use the identity mapping $\Sigma = X$. If we want to exclude histories which eventually become unobservable, we can modify slightly A' so that infinite runs which eventually perform only transitions labeled with ε are not accepted.

Some other interesting extensions do not seem to be as immediate. One problem is the corresponding minimization problem for regular events, or more generally, we may have a mix of properties some of which are desirable and some are undesirable. Of course, since regular sets are closed under complementation, we could use our techniques to solve this problem by complementing the undesirable properties. However, we do not know whether we can avoid an increase in complexity, i.e., whether we can solve this problem with the same efficiency as that of Theorem 5.1.

A more general problem is to develop a theory that combines sample path properties, such as the ones we consider here, with state dependent rewards as in the traditional cost structures of MDP's. For example, we would like to solve the classic MDP optimization problem conditioned on the event that the trajectories generated by the policies must satisfy certain sample path properties similar to the ones considered in this paper.

Appendix

We give a number of definitions and lemmata which are used in the proofs in the paper.

For a pair of states $x_{\text{rep}} \in X$, $s_{\text{rep}} \in S$, we define the event Ψ as the event that an infinite trajectory of the Markov chain M starting at x_{rep} has a corresponding run of $\tau_{M\times A}$ that starts in $(x_{\text{rep}}, s_{\text{rep}})$ and repeats $(x_{\text{rep}}, s_{\text{rep}})$ infinitely often. For a pair of initial states $x_{\text{init}} \in X$, $s_{\text{init}} \in S$, and a pair of repetition states $x_{\text{rep}} \in X$, $s_{\text{rep}} \in S$, we define the event Ω as the event that an infinite trajectory of the Markov chain M starting at x_{init} has a corresponding run of $\tau_{M\times A}$ that starts in $(x_{\text{init}}, s_{\text{init}})$ and repeats $(x_{\text{rep}}, s_{\text{rep}})$ infinitely often. Given a transition table $\tau_A = (X, S, \rho)$ and a set of states $y \in 2^S$ and a word $w \in X^*$ we use the notation $\rho(y,w)$ to denote the set $\cup\{\rho(s,w) \mid s \in y\}$.

The next lemma describes a structural property which holds for all Buchi automata.

Lemma A.1: For an automaton $A = (\tau_A, \{s_0\}, F)$ and an infinite word w, $w \in A$ iff w can be written as $w = w_0w_1w_2\cdots$, where $w_n, n = 0,1,2,\cdots$, are finite words with the following property: there exist a state $f \in F$ and a set $Q \in 2^S$ containing f such that

1) $f \in \rho(\{s_0\}, w_0)$;
2) $\rho(\{f\}, w_i) = \rho(Q, w_i) = Q$, $i = 1,2,\cdots$.

Proof: Clearly if w satisfies the above conditions it is accepted by A. We prove now that if $w \in A$ then 1) and 2) must hold.

Since w is accepted there is a corresponding run of A repeating some state $f \in F$ infinitely often. Let $t_1, t_2, \cdots$, be the times that the run repeats state f and let $w(i)$ denote the ith letter in the word w. At each t_i, $i = 1, 2, \cdots$, we start a new copy of $\langle \det(\tau_A), \{f\} \rangle$ to produce a run over $w(t_i+1)w(t_i+2)\cdots$ starting in state $\{f\}$. Let $d_{t_i}(t)$ be the state of such a run at time $t \geq t_i$ (if $\bar{w} = w(t_i+1)\cdots w(t)$ then $d_{t_i}(t) = \rho(\{f\}, \bar{w})$.) It is easy to observe that if $i < j$, then for all $t > t_i, t_j$, we get that $d_{t_j}(t) \subseteq d_{t_i}(t)$. Also, if $d_{t_j}(k) = d_{t_i}(k)$ for some k, i, j, then $d_{t_j}(t) = d_{t_i}(t)$ for all $t > k$.

We define now the set of *strictly ordered* runs $d_{\hat{t}_1}(t), d_{\hat{t}_2}(t), \cdots$, where $\hat{t}_i, i = 1, 2, \cdots$, is a subsequence of $t_i, i = 1, 2, \cdots$, as follows. Let $\hat{t}_1 = t_1$ and define $\hat{t}_{i+1} = \min\{t_j \mid t_j > \hat{t}_i, d_{t_j}(t) \subset d_{\hat{t}_i}(t) \ \forall t > t_j\}$ if the above set is nonempty, else $\hat{t}_{i+1}$ is not defined. Since $0 < \hat{t}_1 < \cdots < \hat{t}_k < t$ implies that $d_{\hat{t}_k}(t) \subset \cdots \subset d_{\hat{t}_1}(t)$, it follows that there can be at most $|S|$ strictly ordered runs, and hence there are finitely many $\hat{t}_i$s. Let $\hat{t}_m$ be the largest element in this set. This $\hat{t}_m$ has the property that $\hat{t}_m \in \{t_1, t_2, \cdots\}$ and for all $t_i > \hat{t}_m$ there is some sufficiently large $k > t_i$ for which $d_{t_i}(t) = d_{\hat{t}_m}(t)$, $\forall t > k$. We call $d_{\hat{t}_m}(t)$ the *minimal run* of $\det(\tau_{M \times A}, (x, s))$ over w.

Consider the infinite sequence $d_{\hat{t}_m}(t_i), t_i \geq \hat{t}_m$. Since its elements are from a finite set it follows that there is a state $Q \in 2^S$ that appears infinitely often in the sequence $d_{\hat{t}_m}(t_i)$. Also since by definition f appears in the run of A at the times $t_i, i = 1, 2, \cdots$, it follows that $f \in d_{\hat{t}_m}(t_i)$, $\forall t_i > \hat{t}_m$, and hence $f \in Q$. Define the sequence $\zeta_0 = \min\{t_i \mid t_i > \hat{t}_m, d_{\hat{t}_m}(t_i) = Q\}$, $\zeta_{i+1} = \min\{t_i \mid t_i > \zeta_i, d_{\hat{t}_m}(t_i) = d_{\zeta_i}(t_i) = Q\}$, $i = 0, 1, \cdots$. Since Q appears infinitely often in $d_{\hat{t}_m}(t_i)$, $i = 1, 2, \cdots$, and all runs $d_{t_i}(t), t_i > \hat{t}_m$ will eventually coincide with $d_{\hat{t}_m}(t)$, it follows that ζ_i, $i = 0, 1, \cdots$, are all finite. By defining $w_0 = w(1)\cdots w(\zeta_0)$, and $w_{i+1} = w(\zeta_i + 1)\cdots w(\zeta_{i+1})$, $i = 0, 1, \cdots$, the proof of the lemma follows. □

A consequence of the above lemma is that to any infinite word $w = w_0 w_1 \cdots$ accepted by the automaton $A = (\tau_A, \{s_0\}, F)$ there correspond an accepting state $f \in F$ and a strongly connected subset C of states in $\det(\tau_A)$ such that the runs of $\det(\tau_A)$ starting from $\{f\}$ over $w_i w_{i+1} \cdots$, $i = 1, 2, \cdots$, repeat infinitely often exactly the states in C. Clearly C has the property that for the state Q as defined in Lemma A.1, $Q \in C$. Furthermore, for all i sufficiently large, the run of $\det(\tau_A)$ over w_i starting at Q stays entirely inside C.

The next lemma states that if there are randomizing transitions out of a component, then no policy which repeats all states in the component infinitely often can produce trajectories which will not eventually leave the component. Let $L = \tau_M \times \det(\tau_A)$, as defined in Section III.

Lemma A.2: Let C be a strongly connected subset of L with transitions out of C corresponding to randomizing transitions of M. Let $\mathbf{E}$ be the set of trajectories $\mathbf{X}$ for which there is a random time $\sigma < \infty$ such that the run of L over $X_\sigma, X_{\sigma+1}, \cdots$ starting from $(x, \{s\})$ repeats infinitely often exactly the set of states in C. Then for any policy u, $P(\mathbf{E}) = 0$.

Proof: Let $z \in C$ be such that the transition $z \to z'$, z' not in C, corresponds to the randomized transition $u \to v$ of the chain M having probability p_{uv}. Define t_n to be the random time such that $X_n = x$ for the nth time and define $\mathbf{H}_n$ to be the set of trajectories for which the run of L over $X_{t_n}, X_{t_n+1}, \cdots$ starting from $(x, \{s\})$ repeats infinitely often exactly the set of states in C. For any trajectory of the chain $\mathbf{X}$ define the sequence $\mathbf{Y}$ such that $Y_i = 0, 0 < i < t_n$, and $Y_{t_n}, Y_{t_n+1}, \cdots$ is the run of L over $X_{t_n}, X_{t_n+1}, \cdots$ starting from $(x, \{s\})$. Since u is causal, the distribution of the process $\mathbf{Y}$ is such that $P(Y_{k+1} = z' \mid Y_k = z)$ is equal to $p_{uv} > 0$, independent of the past. Since each $\mathbf{X} \in \mathbf{H}_n$ corresponds to a $\mathbf{Y}$ for which $Y_k = z$ occurs infinitely often but the transition $z \to z'$ occurs finitely many times, it follows that the measure of $\mathbf{H}_n$ is zero. Since $\mathbf{E} \subseteq \cup_1^\infty \mathbf{H}_n$ it follows that the measure of $\mathbf{E}$ is zero as well.

Note that if the policy is not causal, the previous result is not true. Assume that besides $z \to z'$ there is an additional transition $y \to y'$, $y, y' \in C$ also corresponding to the randomized transition p_{uv}. If the policy knew the future, it could predict when the transition $u \to v$ will occur and control the process so that every time $u \to v$ occurs, $\mathbf{Y}$ finds itself in state y instead of state z and hence prohibit $\mathbf{Y}$ from exiting C. In this case the "dangerous" transition of the chain occurs infinitely often but the policy is prepared for it when it occurs. If no knowledge of the future is available, the only way to avoid with certainty the transition $z \to z'$ is to prohibit $\mathbf{Y}$ from going to state z and hence taking z out of C. □

The next two lemmata provide the properties of the controllably recurrent states we need in the rest of the proofs.

Lemma A.3: Assume that the concurrent Markov chain starts at x and the automaton starts at s, and consider the event Ψ with $x_{\text{rep}} = x$, $s_{\text{rep}} = s$. Then the following statements are equivalent.

i) (x, s) is controllably recurrent.

ii) (x, s) satisfies Definition 3.1 with 2) replaced by 2′): There is a nonempty finite string $\gamma \in X^*$ with the following properties: the run of L over γ starting at y ends in y by visiting only states in C, and the run of L over γ starting at $(x, \{s\})$ ends in state y.

iii) There exist a policy u for which $P(\Psi) > 0$.

Proof: 1): ii) implies i). This follows trivially since ii) is a stronger condition than i).

2): i) implies iii). Assume that (x, s) is controllably recurrent and assume that the Markov chain M starts in state x and the automaton A is in state s. Let C, y, γ be as in the Definition 3.1. We show first a slightly stronger result: i) implies that there is a policy u for which the probability of Ψ conditioned on the event that M performs first the transitions in γ, is equal to one. In order to show this we will provide a strategy for which with probability one every sample path of the chain with prefix $x \cdot \gamma$ has a corresponding run of $\tau_{M \times A}$ that starts in (x, s) and repeats (x, s) infinitely often. We call this strategy $\theta(x, s)$. Since we already discussed that if a policy uses $\theta(\gamma)$

at time 0, γ will appear with probability p_γ, it follows that the above construction guarantees the existence of a policy for which $P(\Psi) \geq p_\gamma$.

Consider the Markov chains M and M' constructed on the same probability space with initial states $x_{\text{init}} = x, s_{\text{init}} = s$. Let $\mathbf{X} = X_0, X_1, \cdots$ be an infinite trajectory of the Markov chain M such that $X_0 = x, X_1 \cdots X_m = \gamma$ and let $\mathbf{Y}$ be the corresponding trajectory of M'. By the definition of γ we have that $Y_m \in C$. We start now using the strategy $\theta(x,s)$ which is defined as follows. Let the stopping time n_1 be the first time at which both of the following conditions are met: 1) $Y_{n_1} = y$ and 2) $X_{n_1+1} \cdots X_{n_1+m} = \gamma$. Our strategy makes n_1 almost surely finite by using at any time k the information of the graph of L and the histories $X_0, \cdots, X_k$, $Y_0, \cdots, Y_k$ as follows: while $Y_k \neq y$ it uses the alternating strategy in selecting transitions out of the control states by excluding transitions for which Y_{k+1} exits C; when $Y_k = y$ it uses the strategy $\theta(\gamma)$. If $\theta(\gamma)$ fails it switches back to the alternating strategy (until y appears again). If $\theta(\gamma)$ succeeds, the stopping condition for n_1 is met and the policy terminates after having produced the trajectory $X_0, \cdots, X_{n_1+m}$. One can easily see that n_1 is almost surely finite by the following argument. By construction $Y_m \in C$ and C is a strongly connected subset of M' with no randomizing transitions out of C. If the policy excludes the remaining transitions out of C and uses the alternating strategy while in C it follows by ergodicity and by the definition of the above policy that $\mathbf{Y}$ will visit infinitely often all states in C with probability one, and hence the particular state $y \in C, (x,s) \in y$ will with probability one always eventually appear in $\mathbf{Y}$. Assume that $Y_k = y$ for some $k > 0$. Then $X_k = x$ and since $\theta(\gamma)$ is used at time k it follows that the probability $P(X_{k+1} = x_1, \cdots, X_{k+m} = x_m \mid Y_k = y)$ is equal to $p_\gamma > 0$. The above observations imply that the complement of the event $\{Y_k = y, X_{k+1} = x_1, \cdots, X_{k+m} = x_m\}$ has zero probability and hence n_1 is finite with probability one. We can construct now the first piece $z_0, \cdots, z_{n_1}$ of the run of $\tau_{M\times A}$ on $\mathbf{X}$ such that $z_0 = z_{n_k} = (x,s)$. Since $X_{n_1} = x, X_{n_1+1}, \cdots$ has the prefix γ and the future of the randomized transitions after time $n_1 + m$ is independent of the past, we can repeat the previous step and use strategy $\theta(x,s)$ to construct a second piece of the trajectory of the chain with similar properties (has a run which starts and ends in (x,s)). By repeating this procedure forever we construct an infinite trajectory with a corresponding run that repeats (x,s) infinitely often.

3): iii) implies ii). Assume that there is a policy for which $P(\Psi) > 0$. Consider Lemma A.1 and the subsequent remarks for the automaton $(\tau_{M\times A}, \{(x,s)\}, \{(x,s)\})$ and the infinite word $\mathbf{X} \in \Psi$. Let y and C, $(x,s) \in y$, $y \in C$, be the state and the strongly connected component of states of L corresponding to $\mathbf{X}$ by the above lemma (y corresponds to the set Q in the lemma). Since $P(\Psi) > 0$ and there are finitely many such sets y and C it follows that we can always choose a specific pair so that the corresponding set of trajectories $\mathbf{X}$ forms an event $\Psi' \subseteq \Psi$ of positive probability. In what follows, we use y and C to denote the above sets. Since each $\mathbf{X} \in \Psi'$ can be written in the form $w_0, w_1, \cdots$, we can define for each such $\mathbf{X}$ the random time σ corresponding to the beginning of the word w_1. Then Lemma A.1 implies that the run of L starting from $(x, \{s\})$ over $X_\sigma X_{\sigma+1} \cdots, \mathbf{X} \in \Psi'$, will repeat infinitely often exactly the set of states in C. Since $P(\Psi') > 0$ it follows by using Lemma A.2 that there is no randomizing transition out of C. Consider now any $\mathbf{X} \in \Psi'$. As we mentioned in the remark following Lemma A.1, for sufficiently large i, the run of L starting from $(x, \{s\})$ over w_i stays inside C. Let γ be any such w_i. Then property 2′) is satisfied because of Lemma A.1. □

Lemma A.4: Assume that (x,s) is not controllably recurrent. Then for any pair of initial states $x_{\text{init}} \in X, s_{\text{init}} \in S$ and for any policy u the probability of Ω with $x_{\text{rep}} = x$, $s_{\text{rep}} = s$, is equal to zero.

Proof: Fix a policy u. If $(x_{\text{init}}, s_{\text{init}}) = (x,s)$ then Lemma A.3 implies that $P_u(\Omega) = P_u(\Psi) = 0$. We prove now that $P_u(\Omega) = 0$ for arbitrary $(x_{\text{init}}, s_{\text{init}})$. Consider the sequence of events $\mathbf{H}_n, n = 1, 2, \cdots$, defined as follows: $\mathbf{H}_n = \{X_n = x$ and $X_n X_{n+1} \cdots$ has an accepting run in the automaton $(\tau_{M\times A}, \{(x,s)\}, \{(x,s)\})\}$. Let $P_u(\mathbf{H}_n \mid X_0 = x_0, X_n = x, X_1, \cdots, X_{n-1})$ be the conditional probability of $\mathbf{H}_n$ conditioned on the past of the chain prior to time n. Then $P_u(\mathbf{H}_n \mid X_0 = x_0, X_n = x, X_1, \cdots, X_{n-1}) = 0$ almost surely by observing that for any value of $X_1, \cdots, X_{n-1}$ this probability is equal to $P'_u(\Psi)$ where u' is a new policy corresponding to u given the additional past information $X_0, \cdots, X_n$. Since u' is also causal, $P'_u(\Psi) = 0$. The above observation implies that $P_u(\mathbf{H}_n) = 0$ and since $\Omega \subseteq \cup_1^\infty \mathbf{H}_n$ it follows that $P_u(\Omega) = 0$. □

Proof of Proposition 3.1: Assume first that such a controllably recurrent state (x,f) exists and is reachable with the string $\delta \in X^*$ in the table $\tau_{M\times A}$ starting in state (x_0, s_0). Then if we use the strategy favoring δ, there is a positive probability p_δ that the chain will generate δ during its first $|\delta|$ steps. Now we can start the chain in state x and the automaton in state f. Lemma A.3 guaranties that there exists a strategy under which the probability of producing a trajectory with a corresponding run of $\tau_{M\times A}$ which repeats (x,f) infinitely often is positive and equal to $p(\Psi)$, with $x_{\text{rep}} = x$ and $s_{\text{rep}} = f$. By combining the above strategies it follows that $p(\mathbf{B}) \geq p_\delta p(\Psi) > 0$.

Assume now that there is some u for which $P_u(\mathbf{B}) > 0$. From this it follows that under u there must be states of the form (x,f), $x \in X, f \in F$, for which $P_u(\Omega) > 0$, where Ω is defined with $(x_{\text{init}}, s_{\text{init}}) = (x_0, s_0)$, and $(x_{\text{rep}}, s_{\text{rep}}) = (x,f)$. Clearly such a state must also be reachable in $\tau_{M\times A}$ from (x_0, s_0). If no such state is controllably recurrent we get a contradiction with Lemma A.4. □

Proof of Lemma 3.1: By the definition of the target set T, the run of L over w starting from $(x_0, \{s_0\})$ ends in some state $z \in T$ with the following property: for some (x,f) controllably recurrent and some corresponding subset C from Definition 3.1, there exists a state $z' \in C$ such that $z' \subseteq z$. We use from time n onwards the strategy $\theta(x,f)$ as defined in part 2) of the proof of Lemma A.3, by considering the state Y_m in the above proof to be the state z'. Clearly we can do so since $z' \subseteq z$, and the proof in Lemma A.3 implies that the trajectory we construct will have a corresponding run of $\tau_{M\times A}$ which repeats (x,f) (equivalently a run of τ_A that repeats f) infinitely often with probability one. □

Proof of Lemma 3.2: Fix a policy u and assume that there exist a set Φ of trajectories of M starting from x_0, such that $\Phi \subseteq A$, $P_u(\Phi) > 0$, and for all $\mathbf{X} \in \Phi$ the corresponding $\mathbf{Y}$ does not hit T. We will get a contradiction as follows.

Since $P_u(\Phi) > 0$ and $\Phi \subseteq A$, it follows from Lemma A.3 that there must exist some controllably recurrent state $(x, f), f \in F$, and a subset Φ' of Φ with $P_u(\Phi') > 0$, such that for any $\mathbf{X} \in \Phi'$ there is a run of $\tau_{M\times A}$ starting at (x_0, s_0) which repeats (x, f) infinitely often. For $\mathbf{X} \in \Phi'$ let $r(\mathbf{X})$ denote the above corresponding run of $\tau_{M\times A}$. Let $\mathbf{H}_n$ be the subset of Φ' consisting of all $\mathbf{X}$ for which $r(\mathbf{X})$ visits for the first time state (x, f) at time n. Clearly, there must be some n for which $P_u(\mathbf{H}_n) > 0$. Assume now that $P_u(\mathbf{H}_n) > 0$. By a similar argument as in the proof of part 3) of Lemma A.3, there must exist a subset $\mathbf{H}'_n \subseteq \mathbf{H}_n$, $P_u(\mathbf{H}'_n) > 0$, a strongly connected subset C of states of L which satisfies the conditions 1), 2) of Definition 3.1, and a random time $\sigma > n$, for which if $\mathbf{X} \in \mathbf{H}'_n$ then the run of L starting from $(x, \{f\})$ over $X_\sigma, X_{\sigma+1}, \cdots$ will repeat infinitely often the states in C. Let $y \in C$. Then the run of L starting from $(x_0, \{s_0\})$ over $\mathbf{X}$, $\mathbf{X} \in \mathbf{H}'_n$ will repeat infinitely often some state y', $y \subseteq y'$. Clearly $y' \in T$, hence we get a contradiction since $P_u(\mathbf{H}'_n) > 0$. □

Proof of Theorem 3.1: It follows from the fact that for any policy π, $P_\pi(A) \leq P_\pi(\zeta(T) < \infty) = P_{u(\pi)}(A)$. □

Proof of Proposition 3.2: We transform the original chain M' as follows. We remove all transitions out of the states in T, we add an absorbing state 0, and with each state in T we associate a unique transition producing a unit of reward, to state 0. With each control state we associate an action for each controlled transition out that state, and with each randomizing state we associate a trivial action; all the above actions produce zero reward. Note that the only way a reward can be produced is by reaching some state in T. The above system fits into the positive dynamic programming formulation in [23], and LP1 follows from the linear program found in Section IV-C2 of the above reference. □

Proof of Lemma 4.1: Define the event $\mathbf{H}_i$ as the event on which Z_t eventually hits y_i^*, $i = 1, \cdots, m$. Note that these events are disjoint. One can easily see that $R_\pi^2 = \sum_i k_i P_\pi(\mathbf{H}_i)$. By the definition of $u(\pi)$, conditioned on the event $\mathbf{H}_i$, the policy will generate with probability one a trajectory (the projection of $\mathbf{Y}$ on the component corresponding to M) in A_i, hence $P_{u(\pi)}(A_i) \geq P_\pi(\mathbf{H}_i)$, which implies that $R^1_{u(\pi)} = \sum_i k_i P_{u(\pi)}(A_i) \geq \sum_i k_i P_\pi(\mathbf{H}_i) = R_\pi^2$. □

Proof of Lemma 4.2: To simplify notation, define the events $\mathbf{B}_i$ and $\mathbf{C}_i$ so that the inequalities of the lemma can be written as $\sum_i k_i P_u(\mathbf{A}_i) \leq \sum_i k_i P_u(\mathbf{B}_i) \leq \sum_i k_i P_u(\mathbf{C}_i)$. Observe that Lemma 3.2 implies that the union of the $\mathbf{B}_i$'s covers P_u, almost surely the $\mathbf{A}_i$'s, and also by construction, the $\mathbf{B}_i$'s are disjoint and satisfy the condition $P_u(\mathbf{A}_i \cap \mathbf{B}_j) = 0$, for $j < i$. This, together with the fact that $k_j \leq k_i$, $j < i$, and that the $\mathbf{A}_i$s are disjoint, imply the first inequality. The same argument hold for the second inequality. The last statement of the lemma follows from the properties of the conditional expectation. □

Proof of Lemma 4.3: Let $\{\mathbf{Y}_s, i\}$ denote the event that $\mathbf{Y}_s$ is a stopping history with $Y_s \in T_i$. Note that there are countably many such events and that Lemma 4.2 together with the definition of $g_u(\mathbf{Y}_s)$ imply that $\sum_j k_j P_u(A_j \mid \mathbf{Y}_s, i) \leq g_u(\mathbf{Y}_s)$. Observe also that R_u^1 can be written as $E_u[r]$ where r is the random variable $\sum_i k_i 1_{\{A_i\}}$, and where $1_{\{\mathbf{A}\}}$ is the indicator function of the event $\mathbf{A}$. By using the above remarks it follows that

$$E_u[r \mid \mathbf{Y}_s, i] = \sum_j k_j P_u(A_j \mid \mathbf{Y}_s, i) \leq g_u(\mathbf{Y}_s) < k_i + \varepsilon.$$

Let $\mathbf{B}$ be the event on which $\psi = \infty$. From 4.4 it follows that

$$\begin{aligned} E_u[r] &= \sum_{\mathbf{Y}_s, i} E_u[r \mid \mathbf{Y}_s, i] P_u(\mathbf{Y}_s, i) + E_u[r \mid \mathbf{B}] P_u(\mathbf{B}) \\ &< \sum_i k_i P_{\pi(u,\varepsilon)}(Z_t \text{ eventually hits } y_i^*) \\ &\quad + \varepsilon P_{\pi(u,\varepsilon)}(Z_t \text{ eventually hits any } y_i^*) \\ &\quad + E_u[r \mid \mathbf{B}] P_u(\mathbf{B}) \\ &\leq R^2_{\pi(u,\varepsilon)} + \varepsilon + E_u[r \mid \mathbf{B}] P_u(\mathbf{B}) \end{aligned}$$

where in the first inequality we used the fact that $P_{\pi(u,\varepsilon)}(Z_t \text{ eventually hits } y_i^*)$ is equal to the probability $P_u(Y_\psi \in T_i, \psi < \infty)$.

To complete the proof it remains to show that $E_u[r \mid \mathbf{B}] P_u(\mathbf{B}) = 0$. We write $\mathbf{B}$ as the union of the following disjoint events $\mathbf{B}_i$, $i = 0, 1, \cdots, m$ (on the probability space of M'). $\mathbf{B}_i$, $i = 1, \cdots, m$, is the event that $\mathbf{Y} \in \mathbf{B}$ and Y_t hits infinitely often some state in the set T_i and does not hit infinitely often any state in $T_{>i}$; $\mathbf{B}_0$ is the event that $\mathbf{Y} \in \mathbf{B}$ and Y_t does not hit infinitely often any state in $\cup_i T_i$. Since these events are disjoint it follows that $E_u[r \mid \mathbf{B}] P_u(\mathbf{B}) = \sum_{i=0}^m E_u[r \mid \mathbf{B}_i] P_u(\mathbf{B}_i)$. Observe first that $E_u[r \mid \mathbf{B}_0] = 0$ by Lemma 3.2. Consider now some $\mathbf{B}_i$, $i \neq 0$. We will show that this set must have measure zero. This is done as follows.

Let σ_n denote the nth time $\mathbf{Y}$ hits the set T_i, and let Γ_j denote the event $\{\sigma_n < \infty$, after time σ_n, $\mathbf{Y}$ hits eventually T_j and never again $T_{>j}\}$. By the definition of $g_u(\mathbf{Y}_s)$, if at all such times σ_n, $n = 1, \cdots$, the policy $\pi(u, \varepsilon)$ did not decide to jump to y_i^*, we must have that $\sum_j k_j P_u(\Gamma_j \mid \mathbf{Y}_{\sigma_n}) \geq k_i + \varepsilon$, which implies

$$k_m \sum_{j>i} P_u\big(\Gamma_j \,\big|\, \mathbf{Y}_{\sigma_n}\big) \geq k_i - \sum_{j \leq i} k_j P_u\big(\Gamma_j \,\big|\, \mathbf{Y}_{\sigma_n}\big) + \varepsilon \geq \varepsilon.$$

This implies that at each time $\sigma_n < \infty$, the probability that $\mathbf{Y}$ in the future will eventually hit some state in $T_{>i}$ is bounded below by some constant δ, uniformly in n. The above analysis implies the following properties for the set $\mathbf{B}_i$.

1) For every $\mathbf{Y}$ in $\mathbf{B}_i$, $\mathbf{Y}$ hits T_i infinitely often and does not hit infinitely often $T_{>i}$.
2) For every $\mathbf{Y}$ in $\mathbf{B}_i$, every time $\mathbf{Y}$ hits T_i, the probability conditioned on the past that $\mathbf{Y}$ will eventually hit in the future some state of $T_{>i}$ is larger than $\delta > 0$.

To complete the proof we show the following lemma.

Lemma A.5: The event $\mathbf{B}_i$ has measure zero.

Proof: To simplify notation, let Q_1 denote the set T_i, Q_2 denote the set $T_{>i}$, and $q(\mathbf{Y}_s) = P_u(Y_t$ will eventually hit Q_2 after time $s \mid \mathbf{Y}_s)$. Let $\mathbf{L}^{(j)}$ be the event that 1) Y_t hits Q_1 i.o., 2) Y_t does not hit Q_2 after time j, and 3) if σ_k denotes the

kth time Y_t hits Q_1, then $q(\mathbf{Y}_{\sigma_k}) > \delta$, $k = 1, 2, \cdots$. We will show that $P_u(\mathbf{L}^{(j)}) = 0$ which will imply that $P_u(\mathbf{B}_i) = 0$ since $\mathbf{B}_i = \cup_{j>0}\mathbf{L}^{(j)}$.

We define ζ_1 to be the first time Y_t hits Q_2 after time j. Let $\mathbf{F}_1$ be the event $\{\mathbf{Y} \mid \zeta_1 < \infty\}$, and $\mathbf{G}_1$ be the event $\{\mathbf{Y} \mid \mathbf{Y} \in \mathbf{F}_1, q(\mathbf{Y}_{\zeta_1}) > \delta\}$. Let $\mathbf{Y}_{\zeta_1}$ be a prefix in $\mathbf{G}_1$. For each such prefix we define the random time $m_1 = m_1(\mathbf{Y}_{\zeta_1})$ to be the smallest time with the property that $P_u(Y_t$ does not hit Q_2 in the interval $\zeta_1, \zeta_1 + m_1 \mid \mathbf{Y}_{\zeta_1}) < 1 - \delta/2$. Note that such a finite m_1 exists for all prefixes in $\mathbf{G}_1$ since all these prefixes satisfy the condition $q(\mathbf{Y}\zeta_1) > \delta$. We define now the event $\mathbf{H}_1 = \{\mathbf{Y} \mid \mathbf{Y} \in \mathbf{G}_1, Y_t$ does not hit Q_2 in the interval $\zeta_1, \zeta_1 + m_1\}$. By the above construction, $P_u(\mathbf{H}_1 \mid \mathbf{G}_1) < 1 - \delta/2$, hence $P_u(\mathbf{H}_1) < 1 - \delta/2$. Let $\rho_1 = \zeta_1 + m_1$.

We continue the construction of the sets $\mathbf{F}_i, \mathbf{G}_i, \mathbf{H}_i$, $i = 2, \cdots$ as follows. Let ζ_i be the first time Y_t hits the set Q_1 after time ρ_{i-1}, $\mathbf{F}_i = \{\mathbf{Y} \mid \mathbf{Y} \in \mathbf{H}_{i-1}, \zeta_i < \infty\}$, $\mathbf{G}_i = \{\mathbf{Y} \mid \mathbf{Y} \in \mathbf{F}_i, q(\mathbf{Y}_{\zeta_i}) > \delta\}$, $\mathbf{H}_i = \{\mathbf{Y} \mid \mathbf{Y} \in \mathbf{G}_i, Y_t$ does not hit Q_2 in the interval $\zeta_i, \zeta_i + m_i\}$, where m_i is defined in a similar way as m_1. Then $P_u(\mathbf{H}_i \mid \mathbf{G}_i) < 1 - \delta/2$, and since $\mathbf{G}_i \subseteq \mathbf{H}_i - 1$ for $i = 1, 2, \cdots$, by induction one can easily show that $P_u(\mathbf{H}_i) < (1 - \delta/2)^i$. Consider now the set $\cap_{i>0}\mathbf{H}_i$. This event is the set of trajectories satisfying the following properties: 1) Y_t hits Q_1 i.o., 2) Y_t does not hit Q_2 in the random intervals $\zeta_i, \zeta_i + m_i$, $i = 1, \cdots$, and 3) every time l at which Y_t hits Q_1, we have that $q(\mathbf{Y}_l) > \delta$. This implies that $\mathbf{L}^{(j)} \subseteq \cap_{i>0}\mathbf{H}_i$, and since by the above construction $P_u(\cap_{i>0}\mathbf{H}_i) = 0$ it follows that $P_u(\mathbf{L}^{(j)}) = 0$ as well. This completes the proof of the lemma. □

Since $\mathbf{B}_i$ must have measure zero, the proof of Lemma 4.3 is complete. □

Proof of Theorem 4.1: Observe first that the cost structure in O_2 implies that an optimal policy always exists, and that this policy can be selected to be deterministic and memoryless; see [9]. Then, Lemma 4.3 implies that for all u and for any $\varepsilon > 0$, $R^1_u - \varepsilon < R^2_{\pi(u,\varepsilon)} \leq R^2_{\pi^*}$. This implies that $\sup_u\{R^1_u\} \leq R^2_{\pi^*}$. But Lemma 4.1 implies that $R^1_{u(\pi^*)} \geq R^2_{\pi^*}$ and the proof is complete. Note that if $T_i = \emptyset$, $i = 1, \cdots, m$, then under any policy $R^1_u = 0$. □

Proof of Proposition 4.1: The proof is similar to the proof of Proposition 3.2 and uses the construction of O_2. □

Proof of Lemma 4.4: Assume first that the run of L over γ ends in a state $z \in C$, for some C satisfying the properties in Definition 4.1. Assume also without loss of generality that C favors the set $R = \{i, j\}$. The following strategy will almost surely produce a trajectory in A_R. The Markov chain M' is in state z at time n. Consider first the index $i \in R$. We use the alternating policy while staying in C until M' hits y^i (see Definition 4.1), and then we switch to the strategy $\theta(w^i)$ (see proof of part (ii) of Lemma A.3); we keep switching between the two strategies until the suffix of the history is w^i, and just prior to that, the state of M' was y^i. This defines the time n^i_1. Note that at this time M' is still in C. We turn now to the index $j \in R$. We perform the similar strategy as we did for the case of i until w^j appears immediately following the state y^j. This defines the time n^j_1. Now we turn back to i and we repeat this procedure forever. One can easily see that with probability one for the resulting trajectory, there is a run of A_i which repeats the state f^i infinitely often at the times $n^i_1, n^i_2, \cdots$. The similar property holds for A_j and the sequence of times $n^j_1, n^j_2, \cdots$. A straightforward extension of the above strategy can be used in the case of general $R \subseteq \{1, \cdots, m\}$, and clearly the same construction will work if after the initial prefix γ the state of L is z', $z \subseteq z'$. □

Proof of Lemma 4.5: We first prove the following lemma.

Lemma A.6: Suppose a word w is in $\mathbf{A}_R$. Then there exist a strongly connected subset C that satisfies part 2) of the conditions in Definition 4.1, and a state $z \in C$ such that: 1) at some time n the run of L over w visits some state z', $z \subseteq z'$, and 2), if we start L from state z at time n, then the run of L over $w(n+1)w(n+2)\cdots$ will stay in C and will repeat infinitely often all states in C. Also if $P_u(\mathbf{A}_R) > 0$, then there exists such a component C which satisfies condition 1) in Definition 4.1.

Proof: Consider each $i \in R$. By Lemma A.1, since $w \in \mathbf{A}_i$, let the times n^i_k, $k \geq 0$ correspond to the times ζ_k (ζ_k is defined in the proof of the above lemma and it is equal to $|w_0| + \cdots + |w_k|$ in the definition of the lemma), the accepting state f^i of A_i correspond to f, and the set y_i of states of A_i, $f^i \in y_i$ correspond to the set Q in the above lemma. Let $\tilde{n} = \max_{i \in R} n^i_0$. For $i \in R$, let $\tilde{z}_i$ be the state of the run of $\det(\tau_{A_i})$ over $w(n^i_0)w(n^i_0+1)\cdots$ at time $\tilde{n}$, starting at time n^i_0 in state y_i; for $i \notin R$, let $\tilde{z}_i$ be the state of the run of $\det(\tau_{A_i})$ over $w(0)w(1)\cdots$ at time $\tilde{n}$, starting at time 0 in state s^0_i. Consider now the table L. Start L at time $\tilde{n}$ in state $\tilde{z} = (w(\tilde{n}), \tilde{z}_1, \cdots, \tilde{z}_m)$. Let n be a time greater than $\tilde{n}$ such that after time n the above run of L over $w(\tilde{n})w(\tilde{n}+1)\cdots$ goes only through states that repeat infinitely often. Let z be the state of the above run at this time n, and let C be the set of states that repeat infinitely often; clearly C is strongly connected. By construction, if we continue the run after time n, then the ith component of the state will repeat infinitely often the set y_i and this will at least occur at the times n^i_k such that $n^i_k > n$. Let $y^i = (x, z_1, \cdots, y_i, \cdots, z_m)$ be the state of the run at any of the above times. Then the set C satisfies part 2) of Definition 4.1 by using the above y^i and the state f^i defined earlier together with the word $w^i = w(n^i_k)\cdots w(n^i_{k+1} - 1)$. Clearly now C and z satisfy the conditions of the lemma since the run of L over w starting from $(x, s^1_0, \cdots, s^m_0)$ will be at time $\tilde{n}$ at a state containing $\tilde{z}$, and hence at time n it will contain z as well.

If $P_u(\mathbf{A}_R) > 0$, it follows by the same arguments we used in the proof of part iii) of Lemma A.3 that there must be such a component C which satisfies condition 1) in Definition 4.1. □

The previous lemma implies that the run of L over w starting from $(x, s^1_0, \cdots, s^m_0)$ will repeat infinitely often after time n states containing the states in C. The proof of Lemma 4.5 follows easily now from the definition of the sets T_R. □

Proof of Proposition 4.2: The proof is similar to the proof of Proposition 4.1. □

REFERENCES

[1] S. Aggarwal, C. Courcoubetis, and P. Wolper, "Adding liveness properties to coupled finite-state machines," *ACM TOPLAS*, 1990.

[2] A. Aho, J. Hopcroft, and J. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, MA: Addison-Wesley, 1974.
[3] M. Abadi, L. Lamport, and P. Wolper, "Realizable and unrealizable specifications of reactive systems," in *Proc. 16th ICALP*, 1989, vol. LNCS 372, pp. 1–17.
[4] F. Beutler and K. Ross, "Optimal policies for controlled Markov chains with a constraint," *J. Math. Analysis and Appl.*, vol. 112, pp. 236–252, 1985.
[5] ———, "Time-average optimal constrained semi-Markov decision processes," *Adv. Appl. Prob.*, vol. 18, no. 2, pp. 341–359, 1986.
[6] R. Cieslak, C. Desclaux, A. Fawaz, and P. Varaiya, "Supervisory control of discrete-event processes with partial information," *IEEE Trans. Automat. Contr.*, vol. 33, pp. 249–260, Mar. 1988.
[7] C. Courcoubetis and M. Yannakakis, "Verifying temporal properties of finite-state probabilistic programs," in *Proc. 29th FOCS*, 1988, pp. 338–345.
[8] ———, "The complexity of probabilistic verification," *J. ACM*, vol. 42, pp. 857–907, Apr. 1995.
[9] C. Derman, *Finite-State Markovian Decision Processes*. New York, Academic, 1970.
[10] S. Hart and M. Sharir, "Probabilistic temporal logic for finite and bounded models," in *Proc. 16th ACM Symp. Theory of Computing*, J. Thistle, Ed. 1984, pp. 1–13.
[11] L. C. M. Kallenberg, *Linear Programming and Finite Markovian Control Problems*. Amsterdam, The Netherlands: Mathematical Center Tracts, 1983.
[12] D. Lehman and S. Shelah, "Reasoning with time and chance," *Inform. and Contr.*, vol. 53, pp. 165–198, 1982.
[13] J. Lin and W. M. Wonham, "On observability of discrete-event systems," *Inform. Sci.*, vol. 44, no. 2, pp. 173–198, 1988.
[14] R. McNaughton, "Testing and generating infinite sequences by a finite automaton," *Inform. and Contr.*, vol. 9, pp. 521–530, 1966.
[15] A. Pnueli, "The temporal logic of concurrent programs," *Theoretical Computer Sci.*, vol. 13, pp. 45–60, 1981.
[16] A. Pnueli and R. Rosner, "On the synthesis of a reactive module," in *Proc. 16th ACM Symp. Principles of Programming Languages*, 1989, pp. 179–190.
[17] ———, "Distributed reactive systems are hard to synthesize," in *Proc. 31th IEEE Symp. Foundations of Computer Science*, 1990, pp. 746–757.
[18] A. Pnueli and L. D. Zuck, "Probabilistic verification," *Inform. and Computation*, vol. 103, pp. 1–29, 1993.
[19] M. L. Puterman, *Markov Decision Processes*. New York: Wiley, 1994.
[20] J. P. Queille and J. Sifakis, "Fairness and related properties in transition systems," IMAG, Grenoble, France, Res. Rep. 292, 1982.
[21] M. O. Rabin, "Automata on infinite objects and Church's problem," in *Proc. Regional AMS Conf. Series in Math.*, 1972, vol. 13, pp. 1–22.
[22] P. Ramadge, "Some tractable supervisory control problems for discrete event systems modeled by Buchi automata," *IEEE Trans. Automat. Contr.*, vol. 34, pp. 10–19, Jan. 1989.
[23] S. Ross, *Introduction to Stochastic Dynamic Programming*. New York: Academic, 1983.
[24] K. Ross and R. Varadarajan, "Markov decision processes with sample path constraints; the communicating case," *Operations Res.*, 1990.
[25] P. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete-event processes," *SIAM J. Contr. and Optimiz.*, vol. 25, no. 1, pp. 206–230, Jan. 1987.
[26] S. Safra, "On the complexity of ω-automata," in *Proc. 29th FOCS*, Oct. 1988, pp. 319–327.
[27] W. Thomas, "Automata on infinite objects," in *Handbook of Theoretical Computer Science*, vol. 2, J. van Leeuwen, Ed. Cambridge, MA: MIT Press, 1990, pp. 133–192.
[28] J. Thistle, "Control of infinite behavior of discrete event systems," Ph.D. dissertation, Univ. of Toronto, 1991.
[29] J. Thistle and W. Wonham, "On the synthesis of supervisors subject to ω-language specifications," *22nd Annual Conf. Information Sciences and Systems*, Princeton NJ, Mar. 1988, pp. 440–444.
[30] M. Vardi, "Automatic verification of probabilistic concurrent finite-state programs," in *Proc. 26th STOC*, 1985.
[31] M. Vardi and P. Wolper, "An automata-theoretic approach to automatic program verification," in *Proc. 1st Symp. Logic in Computer Science*, 1986.
[32] H. Wong-Toi and D. Dill, "Synthesizing processes and schedulers from temporal specifications," *Computer-Aided Verfication (Proc. CAV 90 Workshop)*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 3. American Mathematical Society, 1991.
[33] P. Wolper, "Temporal logic can be more expressive," *Inform. and Contr.*, vol. 56, pp. 72–99, 1983.
[34] W. M. Wonham and P. Ramadge, "On the supremal controllable sublanguage of a given language," *SIAM J. Contr. and Optim.*, vol. 25, no. 3, pp. 637–659, May 1987.
[35] P. Wolper, M. Y. Vardi, and A. P. Sistla, "Reasoning about infinite computation paths," in *Proc. 24th IEEE Symp. Foundations of Computer Science*, 1983, pp. 185–194.

Costas Courcoubetis was born in Athens, Greece. He received the Diploma (1977) from the National Technical University of Athens, Greece, in electrical and mechanical engineering and the M.S. and Ph.D degrees from the University of California, Berkeley, in electrical engineering and computer science, in 1980 and 1982, respectively.

From 1982 to 1991 he was Member of the Technical Staff (MTS) in the Mathematical Sciences Research Center, Bell Laboratories, Murray Hill, NJ, and since 1990 he has been with the Computer Science department, University of Crete, Heraklion, Greece, where he is currently a Professor. He also heads the Telecommunications and Networks Group of the Institute of Computer Science, FORTH. His interests include performance analysis and pricing in broad-band networks with emphasis on ATM, network management, formal methods for the specification, and verification of concurrent and real-time systems.

Mihalis Yannakakis received the Diploma in electrical engineering from the National Technical University of Athens, Greece, in 1975 and the Ph.D. degree in computer science from Princeton University, Princeton, NJ, in 1979.

He has been a Member of Technical Staff at Bell Laboratories, Murray Hill, NJ, since 1978, and he has been Head of the Computing Principles Research Department since 1991. His research interests include algorithms and complexity, combinatorial optimization, databases, testing and verification.

Dr. Yannakakis is the Editor-in-Chief of the *SIAM Journal on Computing* and serves on the editorial boards of the *Journal of the ACM*, *Journal of Computer and System Sciences*, and *Journal of Combinatorial Optimization*. He has served on the program committees and chaired several conferences, including the IEEE Symposium on Foundations of Computer Science and the ACM Symposium on Principles of Database Systems. He is a Fellow of Bell Labs and of the ACM. He is on the Executive Committee of DIMACS, the NSF STC Center for Discrete Mathematics and Theoretical Computer Science.