5-29-2009

Algorithms for Simple Stochastic Games

Elena Valkanova
University of South Florida

Algorithms for Simple Stochastic Games

by

Elena Valkanova

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science in Computer Science
Department of Computer Science and Engineering
College of Engineering
University of South Florida

Major Professor: Rahul Tripathi, Ph.D.
Nagarajan Ranganathan, Ph.D.
Sudeep Sarkar, Ph.D.

Date of Approval:
May 29, 2009

Keywords: game theory, optimal strategies, algorithms, computational complexity,
computational equilibrium

DEDICATION

To my parents

ACKNOWLEDGEMENTS

I would like to thank my advisor Prof. Rahul Tripathi for his guidance and collaboration during working on my thesis. I am extremely grateful for his encouragement, insightful comments, and ideas in my research work.

I am also thankful to my committee members Prof. Nagarajan Ranganathan and Prof. Sudeep Sarkar from Department of Computer Science and Engineering at USF for reviewing my thesis and providing helpful suggestions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

ALGORITHMS FOR SIMPLE STOCHASTIC GAMES

Elena Valkanova

ABSTRACT

A simple stochastic game (SSG) is a game defined on a directed multigraph and played between players MAX and MIN. Both players have control over disjoint subsets of vertices: player MAX controls a subset V_{MAX} and player MIN controls a subset V_{MIN} of vertices. The remaining vertices fall into either V_{AVE}, a subset of vertices that support stochastic transitions, or SINK, a subset of vertices that have zero outdegree and are associated with a payoff in the range $[0, 1]$. The game starts by placing a token on a designated start vertex. The token is moved from its current vertex position to a neighboring one according to certain rules. A fixed strategy σ of player MAX determines where to place the token when the token is at a vertex of V_{MAX}. Likewise, a fixed strategy τ of player MIN determines where to place the token when the token is at a vertex of V_{MIN}. When the token is at a vertex of V_{AVE}, the token is moved to a uniformly at random chosen neighbor. The game stops when the token arrives on a SINK vertex; at this point, player MAX gets the payoff associated with the SINK vertex.

A fundamental question related to SSGs is the SSG value problem: Given a SSG G, is there a strategy of player MAX that gives him an expected payoff at least $1/2$ regardless of the strategy of player MIN? This problem is among the rare natural combinatorial problems that belong to the class NP $\cap$ coNP but for which there is no known polynomial-time algorithm. In this thesis, we survey known algorithms for the SSG value problem and characterize them into four groups of algorithms: iterative approximation, strategy improvement, mathematical programming, and randomized algorithms. We obtain two new algorithmic results: Our first result is an improved worst-case, upper bound on the number of iterations required by the Hoffman-Karp strategy improvement algorithm. Our second result is a randomized Las Vegas strategy improvement algorithm whose expected running time is $O(2^{0.78n})$.

CHAPTER 1

INTRODUCTION

1.1 Motivation

Game theory is a branch of applied mathematics that is used in economics, biology, engineering, and computer science. Game theory captures behavior in strategic situations in which several players must make individual choices that potentially affect the interests of other players. There are different types of games, where the initials conditions or assumptions may vary based on the different final objectives. In many games, a central solution concept is that of computing equilibrium (commonly known as Nash equilibrium), where each player has adopted a strategy that is unlikely to yield a better payoff upon change. The outcomes (i.e, payoffs) in this case are stable in the sense that none of the players would want to deviate from the fixed strategy yielding the equilibrium. A payoff is a number, also called utility, that reflects the desirability of an outcome to a player and incorporates the player's attitude towards risk. There are two types of game representations known: standard (matrix form) and compact form. In the standard form all possible strategies and preferences of all players are explicitly listed. This form is very useful if there are only two players and the players have only a few strategies. In most of the games there are many players (e.g., many traffic streams, many ISPs controlling such streams), and so explicit representation is exponential-sized in the nature of the game. In routing games, the strategy space of each player consists of all possible paths from source to destination in the network, which is exponentially large in the natural size of the game.

The application of game theory in economics was first covered in a 1944 book titled "Theory of Games and Economic Behavior" by John von Neumann and Oskar Morgenstern. Game theory has been used to analyze a wide array of economic phenomena—auctions, bargaining, duopolies, fair division, oligopolies, social network formation, and voting systems. The

solution concepts are defined in norms of rationality. There are two types of games used in economics: cooperative and non-cooperative games. In non-cooperative games, each player uses a strategy that represents a best response to the other strategies. In cooperative games, a group of players coordinate their actions.

Game theory provides a model for interactive computations in multi-agent systems in computer science and logic. In particular, techniques of game theory are applicable to the problem of constructing reliable computer systems. Each game is played on a finite automaton and each state in the automaton is owned by one of the players. The player owning the state with the token can move the token along any of the outgoing edges to a next state, and the next turn starts. In general, plays are infinite and the number of players and their objectives may vary with the application. Autonomous agents with varied interests characterize many computer systems today. Game theory appears to be a natural tool for both designing and analyzing the interactions among such agents. Consequently, there has been much recent interest in applying game theory to systems problems (see [AKP$^+$02, SS95]). One system problem of recent interest is improving the routing paths used by Internet Service Providers (ISPs) by designing mechanisms that enable ISP coordination. The solution to this problem involves interaction between autonomous entities and application of game theoretic approaches.

Yao's [Yao77] principle is a game-theoretic technique for proving lower bounds on the computational complexity of randomized algorithms, and especially of online algorithms. This principle states that to obtain a lower bound on the performance of randomized algorithms, it suffices to determine an appropriate distribution of difficult inputs and to prove that no deterministic algorithm can perform well against that distribution. The theoretical basis of this principle relies on the min-max theorem for two-person zero-sum games, which is a fundamental result in game theory.

Many problems in artificial intelligence, networking, cryptography, computational complexity theory, and computer-aided verification can be reduced to a two-player game with specific winning conditions. The two-player stochastic game model was introduced first by Shapley [Sha53], and a simple stochastic game (SSG) is a restriction of the general stochastic game. SSGs have applications in reactive systems and in synthesizing controllers. An SSG is a game defined on a directed multigraph and has two players—MAX and MIN. In a com-

puter system, choices of MIN player for his strategy correspond to the actions available to the software driver, and the choices of MAX player for his strategy correspond to the non-deterministic behavior of the environment. In this context, the optimization problem is to find an optimal strategy for MIN player in the SSG that minimizes the probability of reaching an error state. In the simple stochastic games, rather than looking for a winning strategy, the goal is to find an *optimal strategy*, that is, a strategy which guarantees the best expected payoff for a player. The decision problem for SSGs is to determine if the MAX player will win with probability greater than $1/2$, when both players use their optimal strategies.

In this thesis, we focus on the algorithmic part of game theory. The field of algorithmic game theory combines computer science concepts of complexity and algorithm design in game theory. The emergence of the internet has motivated the development of algorithms for finding equilibria in games, markets, computational auctions, peer-to-peer systems, and security and information markets. This thesis studies algorithms for SSGs.

1.2 Our Contribution

We present known algorithms for solving SSGs and for finding their optimal strategies. We survey known algorithms and categorized them into four groups as: iterative approximation algorithms, strategy improvement algorithms, mathematical programming algorithms, and randomized algorithms. We introduce basic definitions and concepts required for the analysis of these algorithms. We formalize the notion of optimal strategies of players in SSGs, and characterize the running time of the algorithms by their iteration complexity (i.e., the number of iterations required to perform some fixed algorithm-specific polynomial-time computation). We obtain two new algorithmic results: Our first result is an improved worst-case, upper bound on the number of iterations required by the Hoffman-Karp strategy improvement algorithm. Our second result is a randomized Las Vegas strategy improvement algorithm whose expected running time is $O(2^{0.78n})$.

1.3 Organization

The thesis is organized as follows. In Chapter 1, we list some applications of game theory and present examples. In Chapter 2, we define the SSG value problem, describe some fundamental properties of SSGs, and briefly introduce related models such as parity games, mean payoff games, Markov decision processes, and stochastic games. We survey known algorithms for the SSG value problem in Chapter 3. In Chapter 4, we describe our new algorithmic results on this problem. The results of this chapter were obtained jointly with my advisor (R. Tripathi). Finally, we mention some future directions of work in Chapter 5.

CHAPTER 2

SIMPLE STOCHASTIC GAMES

2.1 Background

A simple stochastic game (SSG) $\mathcal{G}$ is a two-player game, defined on a directed multigraph $G(V, E)$. The vertex set V is partitioned into disjoint subsets V_{MAX}, V_{MIN}, V_{AVE}, and SINK. There are only two vertices in the subset SINK, that are labeled as 0-sink and 1-sink. All vertices of G, except those of SINK, have exactly two outgoing edges. The SINK vertices have only incoming edges but no outgoing edges. One vertex of G is designated as the *start* vertex, labeled start-vertex. The game $\mathcal{G}$ is played by two players MAX and MIN. Before the start of $\mathcal{G}$, the players are required to choose a strategy for playing the game. Both players adhere to their respective strategy throughout the game. A strategy σ for player MAX is a mapping from V_{MAX} to V such that for each $v \in V_{\text{MAX}}$, $(v, \sigma(v)) \in E(G)$. Similarly, a strategy τ for player MIN is a mapping from V_{MIN} to V such that for every $v \in V_{\text{MIN}}$, $(v, \tau(v)) \in E(G)$.

The game $\mathcal{G}$ is played as follows: A token is placed on the start vertex. At each step of the game, the token is moved from its current vertex position v to a neighboring one according to the following rule:

- If the current vertex v belongs to V_{MAX}, then the MAX player takes a turn. The player moves the token from v to $\sigma(v)$.
- If the current vertex v belongs to V_{MIN}, then the MIN player takes a turn. The player moves the token from v to $\tau(v)$.
- If the current vertex v belongs to V_{AVE}, then none of the players takes any turn. Instead, the token is moved from v to a neighbor chosen uniformly at random.

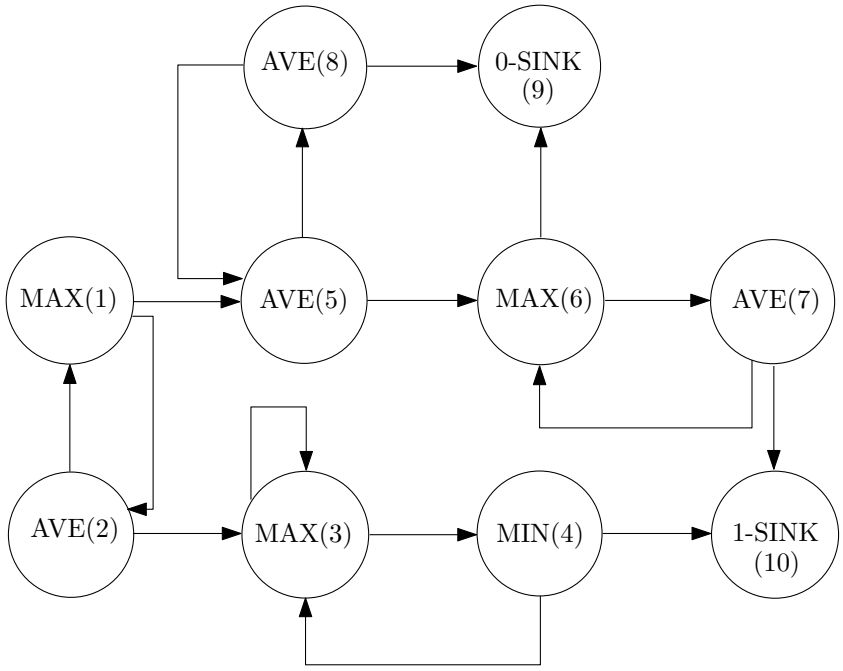

Figure 2.1 A simple stochastic game G with 10 vertices (source: Condon [Con92])

- Winning conditions: If the current vertex v belongs to SINK, then the game stops. If player MIN reaches 0-sink (i.e., v is 0-sink), then he wins the game. Otherwise, player MAX wins the game.

The objective of each player is to maximize his/her chances of winning the game. Thus, MAX would like to choose a strategy σ that gives the maximum chance of the token reaching 1-sink, no matter what strategy MIN chooses. On the other hand, MIN would like to choose a strategy τ that, irrespective of the strategy chosen by MAX, gives the maximum chance of the token reaching 0-sink.

There is also a more general version of SSGs studied in the literature [Fil81, Som05]. In this generalized version, the game is defined on a directed multigraph G that has a set SINK of sink vertices. A payoff $p(s) \in [0, 1]$ is associated with each sink vertex s of G. The payoff $p(s)$ of a sink vertex is a rational number of size polynomial in the number of vertices of G. The remaining vertices of G are partitioned into V_{MAX}, V_{MIN}, and V_{AVE}, as in the original version. If a play reaches a sink s of G, then the play stops and player MAX wins a payoff $p(s)$ from player MIN. The rules of playing the game at positions other than the sink vertices

are the same as before. The objective of player MAX is to maximize his expected payoff and that of player MIN is to minimize this amount.

Condon [Con92] showed that this more general version of SSGs can be transformed into the originally defined SSGs. Henceforth, this thesis will consider only the more general SSGs.

2.2 Notations

The notations used in this thesis are summarized in Table 2.1.

Table 2.1 Notations for Simple Stochastic Games

Notation	Meaning
G	graph defining a simple stochastic game
V	set of vertices of graph $G = (V, E)$
E	set of edges of graph $G = (V, E)$
$\|A\|$	cardinality of set A
$[n]$	set $\{1, 2, \ldots, n\}$
MAX	player 1
MIN	player 2
SINK	vertices that have zero outdegree
start-vertex	the start vertex of G
V_{MAX}	set of all MAX vertices
V_{MIN}	set of all MIN vertices
V_{AVE}	set of all AVE vertices
$p(x)$	payoff associated with a SINK vertex x
σ	strategy of MAX player
τ	strategy of MIN player
$v_{\sigma,\tau}$	expected payoff corresponding to strategies σ and τ
v_{opt}	optimal value vector
$F_{\sigma,\tau}$	operator with respect to strategies σ and τ
F_G	operator with respect to the game G

2.3 Definitions and Preliminaries

Definition 2.1 [strategies] *A strategy σ for player* MAX *is a mapping from V_{MAX} to V such that for each $v \in V_{\text{MAX}}$, $(v, \sigma(v)) \in E(G)$. Similarly, a strategy τ for player* MIN *is a mapping from V_{MIN} to V such that for every $v \in V_{\text{MIN}}$, $(v, \tau(v)) \in E(G)$.*

Definition 2.2 [stopping games] *A simple stochastic game G is a* stopping game *if for any position x and for all strategies σ and τ of the two players, any play of $G(x)$ using strategies σ and τ ends at a sink node with probability one.*

Definition 2.3 [expected payoffs] *Let G be a simple stochastic game with players* MAX *and* MIN*. Let σ and τ denote their respective strategies. Let $q_{\sigma,\tau}(x,s)$ denote the probability that a play of $G(x)$, using strategies σ and τ, ends in a node $s \in$ SINK. The expected payoff vector $v_{\sigma,\tau}$ of G, corresponding to σ and τ, is a vector of values $v_{\sigma,\tau}(x) \in [0,1]$ for each game position x of G such that:*

$$v_{\sigma,\tau}(x) = \sum_{s \in \text{SINK}} q_{\sigma,\tau}(x,s) \cdot p(s).$$

Definition 2.4 *Let $G(V,E)$ be a simple stochastic game with players* MAX *and* MIN*. Let σ and τ denote their respective strategies. Corresponding to σ and τ, the operator $F_{\sigma,\tau} : [0,1]^{|V|} \to [0,1]^{|V|}$ is defined as follows: For every $v \in [0,1]^{|V|}$, $F_{\sigma,\tau}(v) = w$ such that for every $x \in V$,*

$$w(x) = \begin{cases} v(\sigma(x)) & \text{if } x \in V_{\text{MAX}}, \\ v(\tau(x)) & \text{if } x \in V_{\text{MIN}}, \\ \frac{1}{2} \cdot v(y) + \frac{1}{2} \cdot v(z) & \text{if } x \in V_{\text{AVE}} \text{ and } (x,y), (x,z) \in E, \\ p(x) & \text{if } x \in \text{SINK}. \end{cases}$$

Proposition 2.5 *The vector $v_{\sigma,\tau}$ of expected payoffs is the* unique *fixed point of the operator $F_{\sigma,\tau}$. That is, $F_{\sigma,\tau}(v_{\sigma,\tau}) = v_{\sigma,\tau}$.*

Definition 2.6 *Let $G(V,E)$ be a simple stochastic game with players* MAX *and* MIN*. The operator $F_G : [0,1]^{|V|} \to [0,1]^{|V|}$ is defined as follows: For every $v \in [0,1]^{|V|}$, $F_G(v) = w$ such that for every $x \in V$ with (x,y), $(x,z) \in E$,*

$$w(x) = \begin{cases} \max\{v(y), v(z)\} & \text{if } x \in V_{\text{MAX}}, \\ \min\{v(y), v(z)\} & \text{if } x \in V_{\text{MIN}}, \\ \frac{1}{2} \cdot v(y) + \frac{1}{2} \cdot v(z) & \text{if } x \in V_{\text{AVE}}, \\ p(x) & \text{if } x \in \text{SINK}. \end{cases}$$

Definition 2.7 [optimal strategies] *Let $G(V, E)$ be a simple stochastic game with players* MAX *and* MIN*.*

- *The strategies $\sigma_\star$ and $\tau_\star$ are* optimal at a position x *if for any strategy σ of* MAX *and for any strategy τ of* MIN*, it holds that $v_{\sigma,\tau_\star}(x) \leq v_{\sigma_\star,\tau_\star}(x) \leq v_{\sigma_\star,\tau}(x)$.*
- *If $\sigma_\star$ and $\tau_\star$ exist, then $v_{\mathrm{opt}}(x) =_{df} v_{\sigma_\star,\tau_\star}(x)$ is called an optimal value of the game $G(x)$. The vector v_{opt}, whose value at any position x is $v_{\mathrm{opt}}(x)$, is said to be an* optimal value vector *of G.*
- *The strategies $\sigma_\star$ and $\tau_\star$ are called* optimal *for G if they are optimal at every position x of G.*

Shapley [Sha53] showed that there always exists a pair of optimal strategies $\sigma_\star$ and $\tau_\star$ for a *stopping* simple stochastic game. Moreover, any pair of optimal strategies for this game yields the same optimal value vector. Henceforth, we refer to an optimal value vector of a stopping simple stochastic game as the optimal value vector of the game.

Theorem 2.8 (see [Som05]) *Let G be a* stopping *simple stochastic game. Then, there is a* unique *fixed point $v_\star$ of the operator F_G (i.e., $F_G(v_\star) = v_\star$). Moreover, $v_\star(x)$ is the optimal value of $G(x)$ for all $x \in V$.*

Definition 2.9 [greedy strategies] *Let $G = (V, E)$ be a simple stochastic game with players* MAX *and* MIN*. Let $v : V \to \mathbb{R}$ be a value vector for G. A strategy σ of* MAX *player is said to be v-greedy at $x \in$ MAX if $v(\sigma(x)) = \max\{v(y), v(z)\}$, where (x, y), $(x, z) \in E$. Similarly, a strategy τ of* MIN *player is said to be v-greedy at $x \in$ MIN if $v(\tau(x)) = \min\{v(y), v(z)\}$, where (x, y), $(x, z) \in E$. (In both the cases, if there is a tie, i.e., $v(y) = v(z)$, then it is required that for $x \in V_{\mathrm{MAX}}$, $v(\sigma(x))$ equals $\max\{v(y), v(z)\}$, and for $x \in V_{\mathrm{MIN}}$, $v(\tau(x))$ equals $\min\{v(y), v(z)\}$.) For any player $P \in \{\mathrm{MAX}, \mathrm{MIN}\}$, a strategy for P is said to be v-greedy if it is v-greedy at every $x \in V_P$.*

Proposition 2.10 (see [Som05]) *Let G be a* stopping *simple stochastic game. Let v_{opt} be the optimal value vector of G. Then the following statements are equivalent: (a) strategies σ*

and τ are optimal, (b) $v_{\sigma,\tau} = v_{\text{opt}}$, (c) strategies σ and τ are $v_{\sigma,\tau}$-greedy, and (d) strategies σ and τ are v_{opt}-greedy.

Howard [How60] showed that in any stopping SSG G, for every strategy σ of player MAX, there is a strategy $\tau(\sigma)$ of player MIN that is *optimal* w.r.t. σ in the sense that, for every $x \in V_{\text{MIN}}$ with neighbors y and z, $v_{\sigma,\tau(\sigma)}(x)$ is equal to the minimum of $v_{\sigma,\tau(\sigma)}(y)$ and $v_{\sigma,\tau(\sigma)}(z)$. Henceforward, we call $\tau(\sigma)$ an optimal strategy of player MIN with respect to a strategy σ of player MAX. In the same way, we can define $\sigma(\tau)$ as an optimal strategy of player MAX with respect to a strategy τ of player MIN. The strategies $\sigma(\tau)$ and $\tau(\sigma)$ are the best response strategies of players MAX and MIN, respectively. The formal definition is as follows:

Definition 2.11 [best response strategies] *Let G be a simple stochastic game with players* MAX *and* MIN. *A strategy σ of* MAX *is said to be* optimal with respect *to a strategy τ of* MIN *if for all $x \in V_{\text{MAX}}$ with child y, $v_{\sigma,\tau}(x) \geq v_{\sigma,\tau}(y)$. Similarly, a strategy τ of* MIN *is said to be* optimal with respect *to a strategy σ of* MAX *if for all $x \in V_{\text{MIN}}$ with child y, $v_{\sigma,\tau}(x) \leq v_{\sigma,\tau}(y)$.*

Definition 2.12 [switchable nodes] *Let $G = (V, E)$ be a simple stochastic game with players* MAX *and* MIN. *Let $v : V \to \mathbb{R}$ be a value vector for G. Let $x \in V_{\text{MAX}} \cup V_{\text{MIN}}$ has children y and z. The node x is said to be* v-switchable *if $x \in V_{\text{MAX}}$ and $v(x) < \max\{v(y), v(z)\}$, or if $x \in V_{\text{MIN}}$ and $v(x) > \min\{v(y), v(z)\}$.*

Definition 2.13 [stable nodes] *Let $G = (V, E)$ be a simple stochastic game with players* MAX *and* MIN. *Let $v : V \to \mathbb{R}$ be a value vector for G. Let $x \in V_{\text{MAX}} \cup V_{\text{MIN}} \cup V_{\text{AVE}}$ has children y and z. The node x is said to be v-stable if the following holds: If $x \in V_{\text{MAX}}$ then $v(x) = \max\{v(y), v(z)\}$, if $x \in V_{\text{MIN}}$ then $v(x) = \min\{v(y), v(z)\}$, and if $x \in V_{\text{AVE}}$ then $v(x) = \frac{1}{2} \cdot v(y) + \frac{1}{2} \cdot v(z)$. The vector v is said to be stable if for all $x \in V$, x is v-stable. Otherwise, we say that v is not stable.*

SSGs were studied by Condon [Con92, Con93], motivated by complexity-theoretic analysis of randomized space-bounded alternating Turing machines. Condon [Con92] showed that the SSG value problem, defined below, is in NP $\cap$ coNP. This problem is a rare combinatorial problem that belongs to NP $\cap$ coNP, but is not known to be in P.

Definition 2.14 [The SSG Value Problem]

- *The* value *of a SSG G is defined to be* $\max_\sigma \min_\tau v_{\sigma,\tau}$(start-vertex).
- *The* SSG value problem *is defined as follows:*
  SSG-VAL $\equiv$ *Given a SSG G, is the value of $G > \frac{1}{2}$.*

The next lemma states that there is a polynomial-time procedure that transforms a SSG G to a stopping SSG G' such that the value of G' is greater than $1/2$ if and only if the value of G is greater than $1/2$. Thus, a SSG G belongs to the problem SSG-VAL if and only if G' belongs to SSG-VAL, where G' is the output of the procedure mentioned in Lemma 2.15. Henceforth, whenever we say that G is a SSG, we implicitly assume that G is a stopping SSG.

Lemma 2.15 [Con92] *There is a polynomial-time procedure that transforms a SSG G to a stopping SSG G' such that G' has the same number of* MAX *and* MIN *vertices as G and the value of G' is greater than $1/2$ if and only if the value of G is greater than $1/2$.*

Lemma 2.16 states that a pair of optimal strategies $\langle \sigma_\star, \tau_\star \rangle$ of a stopping SSG G is sufficient to solve the SSG value problem on G. It also implies that all pairs of optimal strategies $\langle \sigma, \tau \rangle$ yield the same value vector $v_{\sigma,\tau}$.

Lemma 2.16 [Con92] *For a stopping SSG $G = (V, E)$, let $\langle \sigma_\star, \tau_\star \rangle$ be a pair of optimal strategies of players* MAX *and* MIN*. Then, for all $x \in V$,*

$$v_{\sigma_\star,\tau_\star}(x) = \max_\sigma \min_\tau v_{\sigma,\tau}(x).$$

Lemma 2.17 implies that a pair of strategies $\langle \sigma, \tau \rangle$ that achieve the value of a SSG G on a start vertex x is, in fact, an optimal pair of strategies at position x of G.

Lemma 2.17 [Con92] *Let $G = (V, E)$ be a stopping SSG and let $x \in V$. Then, the following holds for any vertex $x \in V$,*

$$\min_\tau \max_\sigma v_{\sigma,\tau}(x) = \max_\sigma \min_\tau v_{\sigma,\tau}(x).$$

2.4 Related Models

Many variants of SSGs, such as parity games, mean-payoff games, and discounted payoff games, have been extensively studied in the literature [HK66, Der70, FV97]. SSGs are a restriction of stochastic games. Stochastic games are a generalization of Markov decision processes. We next briefly introduce all these models.

2.4.1 Parity Games, Mean Payoff Games, and Discounted Payoff Games

Parity games (PGs), mean payoff games (MPGs) and discounted payoff games (DPGs) are non-cooperative two-person games, played on a directed graph in which each vertex has at least one outgoing edge. In a parity game, the directed graph has two types of vertices, V_{MAX} and V_{MIN}. Each vertex v has a positive integer color $p(v) \in \mathbb{N}$ and has at least one outgoing edge. Similarly to a SSG, the game is between two players MAX and MIN, and the game begins when a token is placed on the start vertex. Depending on the type of the vertex where the token currently lies, players alternately move the token along one of its outgoing edge and construct an infinite path v_0, v_1, v_2, ... called a play. Player MAX wins if the largest vertex color $p(v_i)$ among all vertices v_i occurring infinitely often in a play is odd, and player MIN wins if the color $p(v_i)$ is even, where v_0, v_1, ... is the infinite path formed by the players. Mean payoff games (MPGs) [EM79, GKK88] are similar to PGs, but instead of colored vertices have integer-weighted edges. In MPGs, the first player tries to maximize the average edge weight in the limit whereas the second player tries to minimize this value. MPGs can be used to design and analyze algorithms for job scheduling, finite-window online string matching, and selection with limited storage. In discounted payoff games, we are given rational discount factors. It is known that PGs can be reduced to MPGs (see [GW02]) and that MPGs and DPGs can be reduced to SSGs [ZP96]. The decision problems corresponding to PGS, MPGs, and DPGs are also known to be in $\text{NP} \cap \text{coNP}$.

2.4.2 Markov Decision Processes (MDPS)

A Markov decision process is a single agent controlled stochastic system, which is observed at discrete time points and is described by: a set of states $\mathcal{S}$, a set of actions $\mathcal{A}$, a set of

observations $\mathcal{O}$, a reward function r, a state transition function p, an observation function σ, and an initial state s_0. At every state s, the controller or the agent of the process makes an observation $o(s)$ and then chooses an action $a \in \mathcal{A}$ depending on the observation made. The choice of $a \in \mathcal{A}$ in a state s results in an immediate *reward* $r(s, a)$,and is accompanied by a probabilistic transition to a new state $s' \in \mathcal{S}$. Depending on the type of observation made in every state s, a Markov decision process is called a fully-observable Markov decision process (MDP), an unobservable Markov decision process (UMDP), or a partially-observable Markov decision process (POMDP).

Definition 2.18 (See [MGLA00, FV97]) *A partially-observable Markov decision process is a tuple* $M = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{O}, p, o, r)$*, where*

- $\mathcal{S}$*,* $\mathcal{A}$*, and* $\mathcal{O}$ *are finite sets of* states*,* actions*, and* observations*, respectively,*
- $s_o \in \mathcal{S}$ *is the* initial state*,*
- $p : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ *is the state transition function , i.e.,* $p(s, a, s')$ *is the probability of moving to state* $s' \in \mathcal{S}$ *upon taking action* $a \in \mathcal{A}$ *in state* $s \in \mathcal{S}$*,*
- $o : \mathcal{S} \to \mathcal{O}$ *is the* observation function*, i.e.,* $o(s)$ *is the observation made in state* $s \in \mathcal{S}$*, and*
- $r : \mathcal{S} \times \mathcal{A} \to \mathbb{Z}$ *is the* reward function*, i.e.,* $r(s, a)$ *is the reward gained upon taking an action* $a \in \mathcal{A}$ *in state* $s \in \mathcal{S}$*.*

  If a POMDP *is defined in such a way that the states and the observations coincide, i.e.,* $\mathcal{S} = \mathcal{O}$*, and* o *is the identity function, then the* POMDP *is called* fully-observable *and is denoted by* MDP*. In another case, when the set of observations is a singleton, then the* POMDP *is called* unobservable *and is denoted by* UMDP*.*

Finding an optimal policy in (MDPs) is a problem of optimization theory and is solvable in polynomial time using linear programming [d'E63, Kha79, Kar84]. Markov decision processes (MDPs) are widely used for modeling sequential decision-making problems that arise in engineering and social sciences.

2.4.3 Stochastic Games (also called "Competitive Markov Decision Processes")

Stochastic games, also called competitive Markov decision processes, are multiagent generalizations of Markov decision processes. In stochastic games, the state transition function depends jointly on the actions of all players; the rewards are also determined by the joint actions of the players. A formal definition of two-person stochastic games is as follows.

Definition 2.19 (See [FV97]) *A two-person stochastic game is a tuple* $G = (\mathcal{S},\ s_0,\ \mathcal{A}_1,\ \mathcal{A}_2,\ p,\ r_1,\ r_2)$, *where*

- $\mathcal{S}$, $\mathcal{A}_1$, *and* $\mathcal{A}_2$ *are finite sets of* states, actions of player 1, *and* actions of player 2, *respectively.*
- $s_o \in \mathcal{S}$ *is the* initial state.
- $p : \mathcal{S} \times \mathcal{A}_1 \times \mathcal{A}_2 \times \mathcal{S} \to [0, 1]$ *is the* state transition function, *i.e.,* $p(s, a_1, a_2, s')$ *is the probability of moving to state* s' *upon action* $a_1 \in \mathcal{A}_1$ *by player 1 and action* $a_2 \in \mathcal{A}_2$ *by player 2 in state* $s \in \mathcal{S}$.
- $r_1 : \mathcal{S} \times \mathcal{A}_1 \times \mathcal{A}_2 \to \mathbb{Z}$ *is the reward function of player 1, i.e.,* $r_1(s, a_1, a_2)$ *is the reward gained by player 1 upon taking action* $a_1 \in \mathcal{A}_1$ *by player 1 and action* $a_2 \in \mathcal{A}_2$ *by player 2 in state* $s \in \mathcal{S}$,
- $r_2 : \mathcal{S} \times \mathcal{A}_1 \times \mathcal{A}_2 \to \mathbb{Z}$ *is the reward function of player 2, i.e.,* $r_2(s, a_1, a_2)$ *is the reward gained by player 2 upon taking action* $a_1 \in \mathcal{A}_1$ *by player 1 and action* $a_2 \in \mathcal{A}_2$ *by player 2 in state* s.

CHAPTER 3

ALGORITHMS FOR SIMPLE STOCHASTIC GAMES

Many algorithms have been proposed for solving simple stochastic games. In this chapter, we introduce four main methods (approaches) used in the design of these algorithms. These methods are: the iteration approximation method, the strategy improvement method, the mathematical programming method and randomized algorithms. There is no strict differentiation between these four types and some algorithms involve more than one method to obtain an optimal pair of strategies.

We will restrict our attention to stopping SSGs as discussed in Chapter 2, Section 2.3 (see Lemma 2.15).

3.1 Iterative Approximation Algorithms

In an iterative approximation algorithm for a *stopping* simple stochastic game G, the algorithm begins with an initial value vector $v_1 \in [0,1]^{|V|}$, which is updated from one iteration to another. At the end of all iterations, the algorithm returns the optimal value vector v_{opt} and the optimal strategies $\sigma_\star$, $\tau_\star$ for G.

3.1.1 An Algorithm by Somla

Somla [Som05] proposed two iterative approximation algorithms for simple stochastic games; we describe below one of them.

The algorithm by Somla begins with $v_1 = F_G(0^{|V|})$. At the start of the i'th iteration, there is a current value vector $v_i \in [0,1]^{|V|}$. This value vector is updated to a new one $\tilde{v} \in [0,1]^{|V|}$ by solving a system of linear constraints. The vector $\tilde{v}$ is then transformed into a vector $v_{i+1} = F_G(\tilde{v})$ for the next iteration. The algorithm terminates when the current value vector v_i leads to v_i-greedy strategies $\langle \sigma_i, \tau_i \rangle$, which are optimal for G. The requirement for

optimality of $\langle \sigma_i, \tau_i \rangle$ is given by Proposition 2.10: Strategies $\langle \sigma_i, \tau_i \rangle$ are optimal for G if and only if $\langle \sigma_i, \tau_i \rangle$ are v_{σ_i,τ_i}-greedy, i.e., $F_G(v_{\sigma_i,\tau_i}) = v_{\sigma_i,\tau_i}$. At termination, the algorithm outputs the optimal strategies $\langle \sigma_i, \tau_i \rangle$ and the optimal value vector v_{σ_i,τ_i}.

Algorithm 1: An Algorithm by Somla [Som05]

Input : Simple stochastic game G
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

begin
  Start with $v_1 \leftarrow F_G(0^{|V|})$ and $i \leftarrow 1$
  Find v_i-greedy strategies $\langle \sigma_i, \tau_i \rangle$
  Stop if $\langle \sigma_i, \tau_i \rangle$ are optimal. Return the optimal strategies $\langle \sigma_i, \tau_i \rangle$ and the optimal value vector v_i, which also equals v_{σ_i,τ_i}
  Find a valuation v that maximizes $\sum_x v(x)$ and satisfies the linear constraints:
    (a) $v_i \sqsubseteq v$
    (b) strategies $\langle \sigma_i, \tau_i \rangle$ are v-greedy
    (c) $v \sqsubseteq F_{\sigma_i,\tau_i}(v)$
  Take $v_{i+1} \leftarrow F_G(v)$ and REPEAT steps 3-9
end

Define a partial order on $V \to [0,1]$ by $v \sqsubseteq v'$ if and only if $v(x) \leq v'(x)$ for all $x \in V$. Let $W \subseteq [0,1]^{|V|}$ be a region defined by

$$W = \{v \in [0,1]^{|V|} \mid v \sqsubseteq F_G(v)\}.$$

It can be shown that the optimal expected payoff v_{opt} is the maximal point of W under the partial order defined by $\sqsubseteq$. However, W is described by local optimality equations, which are nonlinear. The algorithm uses the crucial idea of partitioning W into subregions $W_{\sigma,\tau}$ in which the equations are linear. The subregions $W_{\sigma,\tau}$ are defined for each given strategies σ and τ of respective players as follows:

$$W_{\sigma,\tau} = \{v \in W \mid \langle \sigma, \tau \rangle \text{ are } v\text{-greedy and } v \sqsubseteq F_{\sigma,\tau}(v)\}.$$

The algorithm iterates through one subregion to another. In each iteration, a new subregion is visited and a maximal element $\tilde{v}$ of the current subregion is determined by using linear programming. The subregions are visited in a monotonically increasing order, defined by $\sqsubseteq$,

of their maximal elements. Eventually, the subregion containing the optimal expected payoff v_{opt} is visited in some iteration. At this iteration, the algorithm terminates because of the maximality of v_{opt}.

The algorithm finds an optimal pair of strategies after at most an exponential number of iterations, since the number of different subregions is the same as the number of different strategies. One drawback of this algorithm is that it is possible that the same subregion from the graph is traversed several times during the search for an optimal pair of strategies.

3.1.2 An Algorithm by Shapley

Algorithm 2: An Algorithm by Shapley [Sha53]

Input : Simple stochastic game G
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

begin
  Start with a value vector v initialized as follow: For every $x \in V$

$$v(x) = \begin{cases} 1 & \text{if } x \in V_{\text{MAX}} \\ 0 & \text{if } x \in V_{\text{MIN}} \\ \frac{1}{2} \cdot v(y) + \frac{1}{2} \cdot v(z) & \text{if } x \in V_{\text{AVE}} \text{ and } (x,y),\ (x,z) \in E \\ p(x) & \text{if } x \in \text{SINK}. \end{cases}$$

  while $(F_G(v) \neq v)$ **do**
    Let v' be defined as follows:

$$v'(x) = \begin{cases} \max\{v(y), v(z)\} & \text{if } x \in V_{\text{MAX}} \text{ and } (x,y),\ (x,z) \in E \\ \min\{v(y), v(z)\} & \text{if } x \in V_{\text{MIN}} \text{ and } (x,y),\ (x,z) \in E \\ \frac{1}{2} \cdot v(y) + \frac{1}{2} \cdot v(z) & \text{if } x \in V_{\text{AVE}} \text{ and } (x,y),\ (x,z) \in E \\ v(x) & \text{if } x \in \text{SINK}. \end{cases}$$

    Set $v \leftarrow v'$
  Output v and v-greedy strategies $\langle \sigma, \tau \rangle$
end

The algorithm uses iterative approximation method. It starts with some initial value vector v, where all MIN vertices have value 0, all MAX vertices have value 1, and all AVE vertices are stable. It iteratively updates v such that after each iteration, the vector v gets closer to the optimal value vector v_{opt}. In each iteration step, the value $v(x)$ of each node x is updated based on the value of its children using the operator F_G until $F_G(v) = v$, where F_G is defined in Definition 2.6. Condon [Con93] showed that in the worst case this algorithm requires $\Omega(2^n)$ iterations, where n is the number of nodes in the graph.

3.1.3 The "Converge from Below" Algorithm by Condon

Algorithm 3: The "Converge From Below" Algorithm by Condon [Con93]

Input : Simple stochastic game G
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

begin
  Start with a value vector v in which all nodes $x \in V_{\text{MIN}}$ have value $v(x) = 0$ and all nodes $x \in V_{\text{MAX}} \cup V_{\text{AVE}}$ are stable. To find this v, use linear programming to solve
    minimize $\sum_{x \in V} v'(x)$, subject to
    $v'(x) \geq v'(y)$ if $x \in V_{\text{MAX}}$ and $(x, y) \in E$
    $v'(x) = 0$ if $x \in V_{\text{MIN}}$
    $v'(x) = \frac{1}{2} \cdot v'(y) + \frac{1}{2} \cdot v'(z)$ if $x \in V_{\text{AVE}}$ and (x, y), $(x, z) \in E$
    $v'(x) = p(x)$ if $x \in \text{SINK}$
  while $(F_G(v) \neq v)$ **do**
    Let v' be the solution to the following linear program, LP(v):
      maximize $\sum_{x \in V} v'(x)$, subject to
      $v'(x) \leq v'(y)$ if $x \in V_{\text{MIN}}$ and $(x, y) \in E$
      $v'(x) = \frac{1}{2} \cdot v'(y) + \frac{1}{2} \cdot v'(z)$ if $x \in V_{\text{AVE}}$ and (x, y), $(x, z) \in E$
      $v'(x) = v'(y)$ if $x \in V_{\text{MAX}}$ & (x, y), $(x, z) \in E$, and $v(y) > v(z)$
      $v'(x) = \frac{1}{2} \cdot v'(y) + \frac{1}{2} \cdot v'(z)$ if $x \in V_{\text{MAX}}$ & (x, y), $(x, z) \in E$, and $v(y) = v(z)$
      $v'(x) = p(x)$ if $x \in \text{SINK}$
    Let v be the value vector in which all nodes $x \in V_{\text{MIN}}$ have value $v(x) = v'(x)$ and all nodes $x \in V_{\text{MAX}} \cup V_{\text{AVE}}$ are stable. This v can be found using linear programming as in Step 1
  Output v and v-greedy strategies $\langle \sigma, \tau \rangle$
end

The algorithm uses linear programming to output the optimal value vector v_{opt} for game G. It starts with an initial value vector v where all MIN vertices have value 0 and all vertices in MAX $\cup$ AVE are v-stable. It iteratively invokes Steps 6-9 until all vertices are stabilized. At this point, the algorithm reaches the optimal value vector.

3.2 Strategy Improvement Algorithms

The strategy improvement method for solving a SSG was first introduced by Hoffman and Karp [HK66]. Algorithms, based on this method, start with an initial pair of strategy(ies) for players MAX and MIN. The algorithm iteratively computes an optimal pair of strategies. In

each iteration, the strategy of one of the players is improved by switching the nodes at which the optimal choice is not achieved. The main idea is that local optimization on a strategy will eventually lead to a global optimization over the game.

3.2.1 An Algorithm by Hoffman and Karp

Algorithm 4: An Algorithm by Hoffman and Karp [HK66]

Input : Simple stochastic game G
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

```
1 begin
2     Let σ and τ be arbitrary strategies for players MAX and MIN, respectively
3     while (F_G(v_{σ,τ}) ≠ v_{σ,τ}) do
4         Let σ' be obtained from σ by switching all v_{σ,τ}-switchable MAX vertices
5         Let τ' be an optimal strategy of player MIN w.r.t. σ'
6         Set σ ← σ', τ ← τ'
7     Output ⟨σ, τ⟩ and the optimal value vector v_{σ,τ}
8 end
```

Melekopoglou and Condon [MC90] showed that many variants of the Hoffman-Karp algorithm, where, instead of switching all switchable MAX vertices, only one MAX vertex is switched, require $\Omega(2^n)$ iterations in the worst case. In Chapter 4, we show that the Hoffman-Karp algorithm requires at most $O(2^n/n)$ iterations in the worst case. This is the best known worst-case, upper bound on the iteration complexity of this algorithm.

3.3 Mathematical Programming Algorithms

The SSG value problem can be reduced to mathematical programming problems (e.g., to a quadratic programming problem with non-convex objective function), which are generally NP-hard. Some such algorithms are included in the survey by Filar and Schultz [FS86].

3.3.1 A Quadratic Programming Algorithm by Condon

In this algorithm, a (non-convex) quadratic program with linear constraints is formulated given a SSG G. The following theorem by Condon [Con93] ensures that a locally optimal solution to this quadratic program yields the optimal value vector of G.

Algorithm 5: A Quadratic Programming Algorithm by Condon [Con93]

Input : Simple stochastic game G
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

1 **begin**
2 Define the following quadratic program:

$$\text{Minimize} \sum_{\substack{x \in V_{max} \cup V_{min} \\ \text{with children } y \text{ and } z}} (v(x) - v(y))(v(x) - v(z)),$$

subject to the constraints

$$\begin{array}{ll} v(x) \geq v(y) & \text{if } x \in V_{\text{MAX}} \text{ with child } y \\ v(x) \leq v(y) & \text{if } x \in V_{\text{MIN}} \text{ with child } y \\ v(x) = \frac{1}{2} \cdot v(y) + \frac{1}{2} \cdot v(z) & \text{if } x \in V_{\text{AVE}} \text{ with children } y \text{ and } z \\ v(x) = p(x) & \text{if } x \in \text{SINK} \end{array}$$

3 Find a locally optimal solution v to this quadratic program. Output v and v-greedy strategies $\langle \sigma, \tau \rangle$.
4 **end**

Theorem 3.1 [Con93] *The objective function value is zero if and only if the value vector v is the optimal value vector of the game. Moreover, zero is the only locally optimal solution of the objective function in the feasible region.*

Algorithm 5 finds a locally optimal solution v and outputs v-greedy strategies $\langle \sigma, \tau \rangle$. It is open whether a locally optimal solution to the quadratic program used in this algorithm can be computed in polynomial time.

3.3.2 Linear Programming Algorithms

There are special cases when the SSG value problem can be solved in polynomial time. For instance, for the case when the graph $G = (V, E)$ defining a SSG consists of only two types of vertices, such as (1) AVE and MAX vertices, (2) AVE and MIN vertices, or (3) MAX and MIN, there is a polynomial-time algorithm.

Theorem 3.2 *The SSG value problem restricted to SSGs with only (1)* AVE *and* MAX *vertices, (2)* AVE *and* MIN *vertices, and (3)* MAX *and* MIN *can be solved in polynomial time.*

Algorithm 6: An LP Algorithm for SSGs with Only AVE and MAX Vertices [Der70]

Input : A simple stochastic game $G = (V, E)$ with only AVE and MAX vertices
Output: The optimal value vector v_{opt}

begin

Minimize $\sum_{x \in V} v(x)$,

subject to the constraints:

$$\begin{array}{ll} v(x) \geq v(y) & \text{if } x \in V_{\text{MAX}} \text{ and } (x, y) \in E \\ v(x) \geq \frac{1}{2} \cdot v(y) + \frac{1}{2} \cdot v(z) & \text{if } x \in V_{\text{AVE}} \text{ and } (x, y), (x, z) \in E \\ v(x) = p(x) & x \in \text{SINK} \\ v(x) \geq 0 & x \in V \end{array}$$

Solve this linear program to obtain an optimal value vector v. Output v.

end

Algorithm 7: An LP Algorithm for SSGs with Only AVE and MIN Vertices [Con92]

Input : A simple stochastic game $G = (V, E)$ with AVE and MIN vertices
Output: The optimal value vector v_{opt}

begin

Maximize $\sum_{x \in V} v(x)$,

subject to the constraints:

$$\begin{array}{ll} v(x) \leq v(y) & \text{if } x \in V_{\text{MIN}} \text{ and } (x, y) \in E \\ v(x) \leq \frac{1}{2} \cdot v(j) + \frac{1}{2} \cdot v(k) & \text{if } x \in V_{\text{AVE}} \text{ and } (x, y), (x, z) \in E \\ v(x) = p(x) & x \in \text{SINK} \\ v(x) \geq 0 & x \in V \end{array}$$

Solve this linear program to obtain an optimal value vector v. Output v.

end

Algorithm 8: An LP Algorithm for SSGs with Only MAX and MIN Vertices [Con92]

Input : A simple stochastic game with only MAX and MIN vertices
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

begin
  $D \leftarrow \emptyset$
  $U \leftarrow V - D$
  forall vertices $x \in$ SINK **do**
    $v(x) \leftarrow p(x)$
    $D \leftarrow D \cup \{x\}$
  repeat
    forall vertices $x \in U$ **do**
      if $x \in V_{\text{MAX}}$ with a 1-valued child in D **then**
        move x to D
        $v(x) \leftarrow 1$
      if $x \in V_{\text{MAX}}$ with two 0-valued children in D **then**
        move x to D
        $v(x) \leftarrow 0$
      if $x \in V_{\text{MIN}}$ with a 0-valued child in D **then**
        move x to D
        $v(x) \leftarrow 0$
      if $x \in V_{\text{MIN}}$ with two 1-valued children in D **then**
        move x to D
        $v(x) \leftarrow 1$
  until no vertices are moved from U to D in the loop
  forall vertices $x \in U$ **do**
    $v(x) \leftarrow 0$
  Output v and v-greedy strategies $\langle \sigma, \tau \rangle$.
end

Proof: The proof of case (1) is due to Derman [Der70]. Consider a SSG $G = (V, E)$, which is an input to Algorithm 6. Since G is a stopping SSG, by Lemma 2.16 there is a unique optimal value vector of G. We claim that if v_{LP} is an optimal solution to the LP defined in Algorithm 6, then v_{LP} must be the unique optimal value vector of G.

Assume to the contrary that v_{LP}, which is an optimal solution to the LP, is not an optimal value vector of G. Then, at least one of the following two cases arises: (1) there is a vertex $x \in V_{\mathrm{MAX}}$ with neighbors y, z in G such that $v_{\mathrm{LP}}(x) > \max\{v_{\mathrm{LP}}(y), v_{\mathrm{LP}}(z)\}$, and (2) there is a vertex $x \in V_{\mathrm{AVE}}$ with neighbors y, z in G such that $v_{\mathrm{LP}}(x) > \frac{1}{2} \cdot (v_{\mathrm{LP}}(y) + v_{\mathrm{LP}}(z))$. In the first case, it is easy to see that the value vector v', defined for every $u \in V$ by

$$v'(u) = \begin{cases} v_{\mathrm{LP}}(u) & \text{if } x \neq u, \text{ and} \\ \max\{v_{\mathrm{LP}}(y), v_{\mathrm{LP}}(z)\} & \text{if } x = u \text{ and } (x, y), (x, z) \in E, \end{cases}$$

is a feasible solution of the LP with an improved objective function value: $\sum_{x \in V} v'(x) < \sum_{x \in V} v_{\mathrm{LP}}(x)$. This leads to a contradiction. In the second case, it is easy to see that the value vector v', defined for every $u \in V$ by

$$v'(u) = \begin{cases} v_{\mathrm{LP}}(u) & \text{if } x \neq u, \text{ and} \\ \frac{1}{2} \cdot (v_{\mathrm{LP}}(y) + v_{\mathrm{LP}}(z)) & \text{if } x = u \text{ and } (x, y), (x, z) \in E, \end{cases}$$

is a feasible solution of the LP with an improved objective function value: $\sum_{x \in V} v'(x) < \sum_{x \in V} v_{\mathrm{LP}}(x)$. This also leads to a contradiction. Hence, v_{LP} must be the optimal value vector of G.

In a similar way, we can prove case (2), i.e., that the solution output from Algorithm 7 is the unique optimal value vector of a game G with only AVE and MIN vertices.

For the proof of case (3), consider a stopping SSG G that has only MAX and MIN vertices. In this case, we claim that Algorithm 8 outputs an optimal pair of strategies $\langle \sigma, \tau \rangle$ and the unique optimal value vector v of G in polynomial time. The algorithm maintains two sets of vertices D and U. D is the set of vertices whose values have already been determined and U is the set of vertices whose values are still undetermined. The algorithm runs in polynomial time since on each iteration except the last one at least one vertex is moved from U to D.

The correctness of the algorithm requires some analysis whose details we omit in this thesis. For complete details of the proof, we refer the reader to the paper [Con92]. ∎

3.4 Randomized Algorithms

3.4.1 A Randomized Variant of the Hoffman-Karp Algorithm by Condon

Algorithm 9: A Randomized Algorithm by Condon [Con93]

Input : A stopping simple stochastic game G
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

begin
  Let σ and τ be arbitrary strategies for players MAX and MIN, respectively
  while $(F_G(v_{\sigma,\tau}) \neq v_{\sigma,\tau})$ **do**
    Choose $2n$ non-empty subsets of V_{MAX} randomly and uniformly that are $\vec{v}_{\sigma,\tau}$-switchable
    Let the strategies obtained by switching these subsets be $\sigma_1, \ldots, \sigma_{2n}$
    Let the optimal strategies of player MIN with respect to $\sigma_1, \ldots, \sigma_{2n}$ be $\tau_1, \ldots, \tau_{2n}$, respectively
    Let $1 \leq k \leq 2n$ be an index such that $\sum_{x \in V} v_{\sigma_k,\tau_k}(x) \geq \sum_{x \in V} v_{\sigma_l,\tau_l}(x)$ for all $1 \leq l \leq 2n$
    Let $\sigma \leftarrow \sigma_k$ and $\tau \leftarrow \tau_k$
  Output $\langle \sigma, \tau \rangle$ and the optimal value vector $v_{\sigma,\tau}$
end

The output strategies $\langle \sigma, \tau \rangle$ are optimal since v_{opt} is $v_{\sigma,\tau}$-greedy. The algorithm halts since there are only a finite number of strategies, and no pair can be repeated at the start of a new iteration. Condon [Con93] proved the correctness of the algorithm. She also showed that the expected number of iterations of this algorithm is $2^{n-f(n)} + 2^{o(n)}$, for any function $f(n) = o(n)$, where n is the number of MAX vertices.

3.4.2 A Subexponential Randomized Algorithm by Ludwig

The randomized algorithm for SSGs, proposed by Ludwig [Lud95], is subexponential in the number of vertices of the game when the outdegree of the vertices in the game is at most two. The algorithm will be exponential if a reduction is applied from a game with arbitrary outdegree to a game with outdegree at most two.

Algorithm 10: A Subexponential Randomized Algorithm by Ludwig [Lud95]

Input : A stopping SSG $G = (V, E)$ and a strategy σ of player MAX
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

begin
  repeat
    Choose uniformly at random a vertex $s \in V_{\text{MAX}}$
    Construct a new game $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ as follows:
    $\widetilde{V} \leftarrow V - \{s\}$
    $\widetilde{E} \leftarrow (E - \{(x, y) \in E \mid x = s \text{ or } y = s\}) \cup \{(x, y) \mid (x, s) \in E \text{ and } \sigma(s) = y\}$
    Let the set of MAX vertices of $\widetilde{G}$ be $\widetilde{V}_{\text{MAX}}$. Recursively apply the algorithm to the game $\widetilde{G}$ and the strategy $\widetilde{\sigma} : \widetilde{V}_{\text{MAX}} \to \widetilde{V}$ of player MAX to find an optimal strategy $\widetilde{\sigma'}$ of player MAX for the game $\widetilde{G}$. Here we define $\widetilde{\sigma}$ for every $x \in \widetilde{V}_{\text{MAX}}$ as follows:
    $\widetilde{\sigma}(x) = \sigma(x)$ if $\sigma(x) \neq s$, and $\widetilde{\sigma}(x) = \sigma(s)$ otherwise.
    Extend $\widetilde{\sigma'}$ to a strategy σ' for G by setting $\sigma'(s) = \sigma(s)$.
    Find an optimal strategy τ' for player MIN w.r.t. the strategy σ' in G.
    if the pair $\langle \sigma', \tau' \rangle$ is optimal **then**
      return $\langle \sigma', \tau' \rangle$ and the optimal value vector $v_{\sigma', \tau'}$.
    else
      Let σ be obtained from σ' by switching vertex s
  until an optimal pair of strategies $\langle \sigma, \tau \rangle$ in G is found
  Output $\langle \sigma, \tau \rangle$ and the optimal value vector $v_{\sigma, \tau}$.
end

The algorithm starts with a given strategy σ of player MAX and a random choice of a vertex s. In each iteration, a new strategy σ' is output, which is the best strategy for player MAX with respect to σ at vertex s. The algorithm requires solving a linear program (LP) of size polynomial in n. It is important to note that this LP is solved for each "switch" operation performed by the algorithm, where "switch" is defined as a change in the strategy of player MAX at a single vertex. Therefore, the running time per switch operation is polynomial in n. It can be shown that the expected number of switch operations performed is $2^{O(\sqrt{d})}$, where d is the number of MAX vertices. Hence, the expected running time of this algorithm is $2^{O(\sqrt{d})} \times \text{poly}(n)$, which is sub-exponential in the input size n.

Theorem 3.3 [Lud95] *The expected running time of Algorithm 10 is* $2^{O(\sqrt{\min\{|V_{\text{MAX}}|,|V_{\text{MIN}}|\}})} \times \text{poly}(n)$.

CHAPTER 4

NEW RESULTS

In this chapter, we obtain two new algorithmic results. Our first result is an improved worst-case, upper bound on the number of iterations required by the Hoffman-Karp strategy improvement algorithm. This result is described in Section 4.2. Our second result is a randomized Las Vegas strategy improvement algorithm whose expected running time is $O(2^{0.78n})$. This result is described in Section 4.3.

All the results of this chapter were obtained jointly with my advisor (R. Tripathi). These results appeared in preliminary form in a technical report by V. Kumar and R. Tripathi [KT04].

4.1 Preliminaries

Notation 4.1 *Let G be a* stopping *simple stochastic game with players* MAX *and* MIN*. For every strategy σ of player* MAX*, we use $\tau(\sigma)$ to denote the unique optimal strategy of player* MIN *w.r.t. strategy σ. Likewise, for every strategy τ of player* MIN*, we use $\sigma(\tau)$ to denote the unique optimal strategy of player* MAX *w.r.t. strategy τ.*

Notation 4.2 *Let G be a* stopping *simple stochastic game with players* MAX *and* MIN*. For every strategy σ of player* MAX*, we use S_σ to denote the set of all $v_{\sigma,\tau(\sigma)}$-switchable vertices of G. Likewise, for every strategy τ of player* MIN*, we use T_τ to denote the set of all $v_{\sigma(\tau),\tau}$-switchable vertices of G.*

Definition 4.3 *Let G be a* stopping *simple stochastic game with players* MAX *and* MIN*. For every strategy σ of player* MAX *and subset $S \subseteq S_\sigma$, let* $\text{switch}(\sigma, S) : V_{\text{MAX}} \rightarrow V$ *be a strategy of player* MAX *obtained from σ by switching all vertices of S only. The strategy*

$\mathrm{switch}(\sigma, S)$ *is defined as follows: For every* $x \in V_{\mathrm{MAX}}$ *with neighbors* y, z *such that* $y = \sigma(x)$,

$$\mathrm{switch}(\sigma, S) = \begin{cases} y & \text{if } x \notin S, \text{ and} \\ z & \text{if } x \in S. \end{cases}$$

Likewise, for every strategy τ *of player* MIN *and subset* $T \subseteq T_\tau$, $\mathrm{switch}(\tau, T) : V_{\mathrm{MIN}} \to V$ *is defined as a strategy of player* MIN *obtained from* τ *by switching all vertices of* T *only. Its formal definition is as follows: For every* $x \in V_{\mathrm{MIN}}$ *with neighbors* y, z *such that* $y = \tau(x)$,

$$\mathrm{switch}(\tau, T) = \begin{cases} y & \text{if } x \notin T, \text{ and} \\ z & \text{if } x \in T. \end{cases}$$

Definition 4.4 *Let* v_1, v_2 *be value vectors in* $[0,1]^n$, *for some* $n \in \mathbb{N}^+$. *We say that*

- $v_1 \succeq v_2$ *if for each position* $x \in [n]$, *it holds that* $v_1(x) \geq v_2(x)$.
- $v_1 \succ v_2$ *if* $v_1 \succeq v_2$ *and there is some position* $x \in [n]$ *such that* $v_1(x) > v_2(x)$.
- $v_1 = v_2$ *if for each position* $x \in [n]$, *it holds that* $v_1(x) = v_2(x)$.
- v_1 *and* v_2 *are incomparable if there are positions* $x, y \in [n]$ *such that* $v_1(x) > v_2(x)$ *and* $v_1(y) < v_2(y)$.
- $v_1 \not\succeq v_2$ *if either* $v_2 \succ v_1$, *or* v_1 *and* v_2 *are incomparable.*
- $v_1 \not\succ v_2$ *if either* $v_2 \succeq v_1$, *or* v_1 *and* v_2 *are incomparable.*

Fact 4.5 (see [Juk01, Chapter 1]) *Let* $H(x) = -x \log_2 x - (1-x)\log_2(1-x)$, *where* $0 \leq x \leq 1$ *and* $H(0) = H(1) = 0$, *be the binary entropy function. Then, for every integer* $0 \leq s \leq n/2$, $\sum_{k=0}^{s} \binom{n}{k} \leq 2^{n \cdot H(s/n)}$.

4.2 An Improved Analysis of the Hofffman-Karp Algorithm

In this section, we obtain the first non-trivial, worst-case, upper bound on the number of iterations required by the Hoffman-Karp algorithm, which is described as Algorithm 4 in Section 3.2.1. Our proof technique extends the technique of Mansour and Singh [MS99] for

bounding the number of iterations required by policy iteration algorithms to solve Markov decison processes (MDPs). Simple stochastic games are two-person games over a finite horizon in which the expected payoff depends on the strategies of both players. In contrast, Markov decision processes consist of a single agent whose actions determine the cumulative award over an infinite horizon. Because of these contrasting features, it is not obvious whether techniques developed for analyzing Markov decision processes can be generalized in a straightforward manner to analyze simple stochastic games. One of our contributions is to demonstrate that simple stochastic games indeed carry some structural properties similar to Markov decision processes, which can be harnessed to analyze these games.

The main result of this section is Theorem 4.12, which relies on several lemmas on the properties of simple stochastic games and the Hoffman-Karp algorithm. We first present these lemmas and their proofs, and then use these results in proving the theorem.

Lemma 4.6 *Let G be a* stopping *simple stochastic game with n vertices and with players* MAX *and* MIN*. Let σ and τ be strategies of players* MAX *and* MIN*, respectively. There is an $n \times n$ matrix Q with entries in $\{0, \frac{1}{2}, 1\}$ and an n-vector b with entries in $\{0\} \cup \{p(x) \mid x \in$ SINK$\}$ such that $v_{\sigma,\tau}$ is the unique optimal solution to the equation $v_{\sigma,\tau} = Qv_{\sigma,\tau}+b$. Moreover, $I-Q$ is invertible, all entries of $(I-Q)^{-1}$ are non-negative, and the entries along the diagonal are strictly positive.*

Proof Sketch. The proof of this lemma is a straightforward extension of Lemma 1 in Condon's paper [Con92] to the case of simple stochastic games with payoffs. So, we refer the reader to her paper [Con92] for this proof. ∎

Lemma 4.7 *Let $G = (V, E)$ be a* stopping *simple stochastic game with players* MAX *and* MIN*. Let σ be a strategy of player* MAX *such that S_σ is nonempty and let S be any nonempty subset of S_σ. Let σ' denote* switch(σ, S)*, which a strategy of player* MAX*. Then, it holds that $v_{\sigma',\tau(\sigma')} \succ v_{\sigma,\tau(\sigma)}$.*

Proof. As stated in Notation 4.1, let $\tau(\sigma)$ ($\tau(\sigma')$) denote the unique optimal strategy of player MIN w.r.t. strategy σ (respectively, σ') of player MAX. Throughout this proof, we write τ for $\tau(\sigma)$ and τ' for $\tau(\sigma')$ for notational convenience.

The construction of a pair of strategies $\langle \sigma', \tau' \rangle$ from $\langle \sigma, \tau \rangle$ proceeds in two steps: construction of $\langle \sigma', \tau \rangle$ from $\langle \sigma, \tau \rangle$ in the first step, and construction of $\langle \sigma', \tau' \rangle$ from $\langle \sigma', \tau \rangle$ in the next one. In the first step, σ' is obtained from σ as a result of switching all vertices of S. In the second step, τ' is obtained from τ as a result of switching all vertices of some subset $T \subseteq V_{\text{MIN}}$. Notice that except for vertices in $S \cup T$, all other vertices $x \in V_{\text{MAX}} - S$ and $u \in V_{\text{MIN}} - T$ have $\sigma(x) = \sigma'(x)$ and $\tau(u) = \tau'(u)$.

From Lemma 4.6, we know that $v_{\sigma,\tau}$ and $v_{\sigma',\tau'}$ are the unique solutions to the equations

$$v_{\sigma,\tau} = Q_\sigma v_{\sigma,\tau} + b_\sigma \tag{4.1}$$

$$v_{\sigma',\tau'} = Q_{\sigma'} v_{\sigma',\tau'} + b_{\sigma'}, \tag{4.2}$$

for some Q_σ, $Q_{\sigma'}$, b_σ, and $b_{\sigma'}$. Let $\triangle = v_{\sigma',\tau'} - v_{\sigma,\tau}$. We show that $\triangle \geq 0$ and for some entry $\triangle$ is actually > 0. Clearly, by Definition 4.4 this would suffice to prove Lemma 4.7.

Subtracting Eq. (4.2) from Eq. (4.1) , we get

$$\triangle = (Q_{\sigma'} v_{\sigma',\tau'} + b_{\sigma'}) - (Q_\sigma v_{\sigma,\tau} + b_\sigma). \tag{4.3}$$

Adding and subtracting $Q_{\sigma'} v_{\sigma,\tau} + b_{\sigma'}$ to $\triangle$, we get

$$\triangle = (Q_{\sigma'} v_{\sigma',\tau'} + b_{\sigma'}) - (Q_{\sigma'} v_{\sigma,\tau} + b_{\sigma'}) + (Q_{\sigma'} v_{\sigma,\tau} + b_{\sigma'}) - (Q_\sigma v_{\sigma,\tau} + b_\sigma), \text{ or}$$

$$\triangle = Q_{\sigma'} \triangle + \delta, \tag{4.4}$$

where $\delta = (Q_{\sigma'} v_{\sigma,\tau} + b_{\sigma'}) - (Q_\sigma v_{\sigma,\tau} + b_\sigma)$. From Lemma 4.6, we know that $I - Q_{\sigma'}$ is invertible. Hence, there is a unique solution to $\triangle$ given by $\triangle = (I - Q_{\sigma'})^{-1}\delta$. Lemma 4.6 also implies that $(I - Q_{\sigma'})^{-1}$ has all entries non-negative and only positive entries along the diagonal. So, it only suffices to show that $\delta \geq 0$ and that some entry of δ is > 0.

The vector δ is the difference of two vectors A and B, where $A = Q_{\sigma'} v_{\sigma,\tau} + b_{\sigma'}$ and $B = Q_\sigma v_{\sigma,\tau} + b_\sigma$. Notice that the vectors A and B are equivalent to the vectors obtained by applying $F_{\sigma',\tau'}$ and $F_{\sigma,\tau}$, respectively, on $v_{\sigma,\tau}$, where the operator $F_{\sigma',\tau'}$ (or, $F_{\sigma,\tau}$) is defined

in Definition 2.4. In other words, $A = F_{\sigma',\tau'}(v_{\sigma,\tau})$ and $B = F_{\sigma,\tau}(v_{\sigma,\tau})$. Hence, we have

$$\delta = F_{\sigma',\tau'}(v_{\sigma,\tau}) - F_{\sigma,\tau}(v_{\sigma,\tau}). \tag{4.5}$$

Consider an arbitrary vertex $x \in \text{SINK}$ of G. By definition 2.4, it follows that both $A(x)$ and $B(x)$ equal $p(x)$, the payoff associated with x. Hence, in this case $\delta(x) = 0$ from Eq. (4.5). Next, suppose that $x \in S$ is an arbitrary vertex with (x, y), $(x, z) \in E$ such that $y = \sigma(x)$ and $z = \sigma'(x)$. In this case, $A(x)$ equals $v_{\sigma,\tau}(\sigma'(x))$ and $B(x)$ equals $v_{\sigma,\tau}(\sigma(x))$ by Definition 2.4. Therefore, $\delta(x)$ equals $v_{\sigma,\tau}(z) - v_{\sigma,\tau}(y)$ from Eq. (4.5). Since $x \in S$, x is a $v_{\sigma\tau}$-switchable vertex of V_{MAX}, and so by Definition 2.12, it must be the case that $v_{\sigma,\tau}(z) > v_{\sigma,\tau}(y)$. Hence, we get that $\delta(x) > 0$.

Next, suppose that $x \in T$ is an arbitrary vertex with (x, y), $(x, z) \in E$ such that $y = \tau(x)$ and $z = \tau'(x)$. Then, we have $A(x) = v_{\sigma,\tau}(\tau'(x)) = v_{\sigma,\tau}(z)$ and $B(x) = v_{\sigma,\tau}(\tau(x)) = v_{\sigma,\tau}(y)$. Since τ is an optimal strategy w.r.t. σ and since $x \in T \subseteq V_{\text{MIN}}$, $\tau(x)$ must point to the neighbor of x that has the smaller value in $v_{\sigma,\tau}$. That is, it must be the case that $v_{\sigma,\tau}(y) \leq v_{\sigma,\tau}(z)$. Hence, we have $\delta(x) \geq 0$ for every vertex $x \in T$.

Finally, consider any arbitrary vertex $x \in V - (S \cup T \cup \text{SINK})$. In this case, it is easy to see that when restricted to position x, the actions of $F_{\sigma',\tau'}$ and $F_{\sigma,\tau}$ on any value vector v are the same. That is, $A(x)$ equals $B(x)$. It follows from Eq. (4.5) that $\delta(x) = 0$ for every such vertex x.

Thus, we have shown that for every $x \in V$, $v_{\sigma',\tau'}(x) \geq v_{\sigma,\tau}(x)$, and for every $x \in S$, where S is a nonempty subset of S_σ, $v_{\sigma',\tau'}(x) > v_{\sigma,\tau}(x)$. This proves that $v_{\sigma',\tau'} \succ v_{\sigma,\tau}$. ∎

Lemma 4.8 *Let G be a* stopping *simple stochastic game with players* MAX *and* MIN. *Let σ and σ' be strategies of player* MAX *such that σ' is obtained from σ by switching a single vertex $x \in V_{\text{MAX}}$, i.e., $\sigma' = \text{switch}(\sigma, \{x\})$. Then, either $v_{\sigma,\tau(\sigma)} \succ v_{\sigma',\tau(\sigma')}$ or $v_{\sigma',\tau(\sigma')} \succeq v_{\sigma,\tau(\sigma)}$. In other words, $v_{\sigma,\tau(\sigma)}$ and $v_{\sigma',\tau(\sigma')}$ are not incomparable.*

Proof Sketch. The proof of this lemma is almost the same as that of Lemma 4.7. So, we omit this proof. ∎

Lemma 4.9 *Let G be a* stopping *simple stochastic game with players* MAX *and* MIN. *If σ and σ' are two distinct strategies of player* MAX *such that they both agree on vertices in S_σ (i.e., $\forall x \in S_\sigma$, $\sigma(x) = \sigma'(x)$), then $v_{\sigma,\tau(\sigma)} \succeq v_{\sigma',\tau(\sigma')}$.*

Proof. Consider a new game $G' = (V', E')$ obtained from $G = (V, E)$ as follows: V' equals V and E' is obtained from E by deleting all edges $(x, z) \in E$ such that $x \in S_\sigma$ and $z \neq \sigma(x)$, and duplicating all edges $(x, y) \in E$ such that $x \in S_\sigma$ and $y = \sigma(x)$. In other words, we have

$$V' = V$$
$$E' = E - \{(x, z) \mid x \in S_\sigma \text{ and } z \neq \sigma(x)\} + \{(x, y) \mid x \in S_\sigma \text{ and } y = \sigma(x)\}.$$

Here, '+' stands for multiset union and '-' stands for multiset difference operation. Notice that every MAX strategy in G' is also a MAX strategy in G. Also, notice that E' and E have same edges, except the edges that have tail vertex from S_σ. From these observations, it follows that G' is a stopping simple stochastic game. Also, since σ and σ' agree on vertices in S_σ and E' includes all edges (x, y) for which $x \in S_\sigma$ and $y = \sigma(x)$, both $\langle \sigma, \tau(\sigma) \rangle$ and $\langle \sigma', \tau(\sigma') \rangle$ are player strategies for G'. Let $v_{\sigma,\tau(\sigma)}[G]$ be the value vector in G and let $v_{\sigma,\tau(\sigma)}[G']$ be the value vector in G' corresponding to the same pair of strategies $\langle \sigma, \tau \rangle$. In a similar way, we define $v_{\sigma',\tau(\sigma')}[G]$ and $v_{\sigma',\tau(\sigma')}[G']$ corresponding to the pair of strategies $\langle \sigma', \tau' \rangle$. We note that $v_{\sigma,\tau(\sigma)}[G] = v_{\sigma,\tau(\sigma)}[G']$ and that $v_{\sigma',\tau(\sigma')}[G] = v_{\sigma',\tau(\sigma')}[G']$ based on the aforementioned observations.

Next, notice that every vertex $x \notin S_\sigma$ is $v_{\sigma,\tau(\sigma)}[G]$-stable. The equality of $v_{\sigma,\tau(\sigma)}[G]$ and $v_{\sigma,\tau(\sigma)}[G']$ implies that every $x \notin S_\sigma$ is also $v_{\sigma,\tau(\sigma)}[G']$-stable. All edges $(x, y) \in E$ such that $x \in S_\sigma$ and $y = \sigma(x)$ are duplicated in G'. Hence, every $x \notin S_\sigma$ is also $v_{\sigma,\tau(\sigma)}[G']$-stable. Therefore, we see that every vertex of G' is $v_{\sigma,\tau(\sigma)}[G']$-stable, and so strategies σ, τ are $v_{\sigma,\tau(\sigma)}[G']$-greedy (by Definition 2.9). Since G' is a stopping SSG, Proposition 2.10 implies that σ and τ are optimal strategies for G' and $v_{\sigma,\tau(\sigma)}[G']$ is the unique optimal value vector of G'. Thus, we have $v_{\sigma,\tau(\sigma)}[G'] \succeq v_{\sigma',\tau(\sigma')}[G']$ by Lemma 2.16. Using the equality of the value vectors obtained above, we get that $v_{\sigma,\tau(\sigma)}[G] \succeq v_{\sigma',\tau(\sigma')}[G]$. ∎

Lemma 4.10 *Let G be a* stopping *simple stochastic game. Let $\langle \sigma_i, \tau(\sigma_i)\rangle$ and $\langle \sigma_j, \tau(\sigma_j)\rangle$ be pairs of player strategies during iterations i and j, where $i < j$, of the Hoffman-Karp algorithm. Then, it holds that $S_{\sigma_i} \not\subseteq S_{\sigma_j}$.*

Proof. Assume to the contrary that for some $i < j$, $S_{\sigma_i} \subseteq S_{\sigma_j}$. Let S be a subset of S_{σ_i} containing all vertices on which σ_i and σ_j disagree. (That is, $S = \{x \in S_{\sigma_i} \mid \sigma_i(x) \neq \sigma_j(x)\}$.) We define a new strategy σ' for player MAX as follows: $\sigma' = \text{switch}(\sigma_j, S)$. Then, σ_i and σ' agree on vertices in S_{σ_i}. Applying Lemma 4.9, we get that $v_{\sigma_i,\tau(\sigma_i)} \succeq v_{\sigma',\tau(\sigma')}$. On the other hand, using the facts that $\sigma' = \text{switch}(\sigma_j, S)$ and $S \subseteq S_{\sigma_i} \subseteq S_{\sigma_j}$, and using Lemma 4.7, it follows that $v_{\sigma',\tau(\sigma')} \succ v_{\sigma_j,\tau(\sigma_j)}$. By transitivity, we get that $v_{\sigma_i,\tau(\sigma_i)} \succ v_{\sigma_j,\tau(\sigma_j)}$. However, in the Hoffman-Karp algorithm, by Lemma 4.7, the value vectors monotonically increase with the number of iterations, i.e., if $i < j$ then it must be the case that $v_{\sigma_j,\tau(\sigma_j)} \succ v_{\sigma_i,\tau(\sigma_i)}$. This leads to a contradiction. ∎

Lemma 4.11 *In the Hoffman-Karp algorithm for solving simple stochastic games, let σ, $\tau(\sigma)$ be a pair of strategies at the start of an iteration, where S_σ is nonempty, and let $\sigma' = \text{switch}(\sigma, S_\sigma)$ and $\tau' = \tau(\sigma')$ be the pair of strategies at the end of the iteration. There there are at least $|S_\sigma|$ pairs of strategies σ_i, $\tau(\sigma_i)$ such that $v_{\sigma',\tau'} \succeq v_{\sigma_i,\tau_i} \succ v_{\sigma,\tau}$.*

Proof. Without loosing generality, let the elements of S_σ be denoted by 1, 2, …, $|S_\sigma|$. For every $S \subseteq S_\sigma$, we define $\sigma_S = \text{switch}(\sigma, S)$. The unique optimal strategy of player MIN w.r.t. σ_S is denoted as $\tau(\sigma_S)$ using Notation 4.1. However, for notational convenience, we write τ_S for $\tau(\sigma_S)$ throughout this proof.

Assume w.l.o.g. that $v_{\sigma_{\{1\}},\tau_{\{1\}}}$ is a minimal vector among the set of value vectors $v_{\sigma_{\{i\}},\tau_{\{i\}}}$, for every $1 \leq i \leq n$. That is, we assume that for every $1 < i \leq |S_\sigma|$, either $v_{\sigma_{\{i\}},\tau_{\{i\}}} \succeq v_{\sigma_{\{1\}},\tau_{\{1\}}}$ or $v_{\sigma_{\{i\}},\tau_{\{i\}}}$ is incomparable to $v_{\sigma_{\{1\}},\tau_{\{1\}}}$. From Lemma 4.7, we know that $v_{\sigma_{\{1\}},\tau_{\{1\}}} \succ v_{\sigma,\tau}$. We now prove that for every $1 < i \leq |S_\sigma|$, $v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}} \succeq v_{\sigma_{\{1\}},\tau_{\{1\}}}$. Once this is proven, we can pick a minimal vector, say $v_{\sigma_{\{1,j\}},\tau_{\{1,j\}}}$, among the set of value vectors $v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$, for every $1 < i \leq |S_\sigma|$, and repeat the above proof argument iteratively to get a monotonically decreasing sequence $v_{\sigma',\tau'} = v_{\sigma_{S_\sigma},\tau_{S_\sigma}} \succeq \cdots \succeq v_{\sigma_{\{1,j\}},\tau_{\{1,j\}}} \succeq v_{\sigma_{\{1\}},\tau_{\{1\}}} \succ v_{\sigma,\tau}$. Clearly, this would imply the statement of the lemma.

Assume to the contrary that for some $1 < i < |S_\sigma|$, $v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}} \not\succeq v_{\sigma_{\{1\}},\tau_{\{1\}}}$. Since $\sigma_{\{1,i\}} = \text{switch}(\sigma_{\{1\}}, \{i\})$, we know from Lemma 4.8 that for every $1 < i \leq |S_\sigma|$, either $v_{\sigma_{\{1\}},\tau_{\{1\}}} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$ or $v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}} \succeq v_{\sigma_{\{1\}},\tau_{\{1\}}}$. Our assumption $v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}} \not\succeq v_{\sigma_{\{1\}},\tau_{\{1\}}}$ implies that we must have $v_{\sigma_{\{1\}},\tau_{\{1\}}} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$. Next, notice that $\sigma_{\{1,i\}}$ also equals $\text{switch}(\sigma_{\{i\}}, \{1\})$. Therefore, by Lemma 4.8 either $v_{\sigma_{\{i\}},\tau_{\{i\}}} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$ or $v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}} \succeq v_{\sigma_{\{i\}},\tau_{\{i\}}}$. If the latter holds, then by transitivity, we get that $v_{\sigma_{\{1\}},\tau_{\{1\}}} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}} \succeq v_{\sigma_{\{i\}},\tau_{\{i\}}}$, which will contradict the minimality of $v_{\sigma_{\{1\}},\tau_{\{1\}}}$. Hence, it must be the case that $v_{\sigma_{\{i\}},\tau_{\{i\}}} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$. Since $v_{\sigma_{\{1\}},\tau_{\{1\}}} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$ and $\sigma_{\{1\}} = \text{switch}(\sigma_{\{1,i\}}, \{i\})$, we must have $i \in S_{\sigma_{\{1,i\}}}$, and since $v_{\sigma_{\{i\}},\tau_{\{i\}}} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$ and $\sigma_{\{i\}} = \text{switch}(\sigma_{\{1,i\}}, \{1\})$, we must have $1 \in S_{\sigma_{\{1,i\}}}$.

Thus, we have shown that $\{1, i\} \subseteq S_{\sigma_{\{1,i\}}}$. By Lemma 4.7, this implies that σ, which equals $\text{switch}(\sigma_{\{1,i\}}, \{1, i\})$, must satisfy: $v_{\sigma,\tau} \succ v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}}$. However, we know that $\{1, i\} \subseteq S_\sigma$ and $\sigma_{\{1,i\}} = \text{switch}(\sigma, \{1, i\})$, and so by Lemma 4.7 we must also have $v_{\sigma_{\{1,i\}},\tau_{\{1,i\}}} \succ v_{\sigma,\tau}$. This leads to a contradiction. ∎

Theorem 4.12 *The Hoffman-Karp algorithm requires at most $O(2^n/n)$ iterations in the worst case, where $n = \min\{|V_{\text{MAX}}, V_{\text{MIN}}\}$.*

Proof. Assume that $n = |V_{\text{MAX}}| \leq |V_{\text{MIN}}|$; if $|V_{\text{MAX}}| > |V_{\text{MIN}}|$, then we can repeat the same proof argument with MAX and MIN interchanged. In the Hoffman-Karp algorithm, since, by Lemma 4.7, the value vectors monotonically increase with the number of iterations, there can be at most 2^n iterations corresponding to 2^n distinct subsets of switchable MAX vertices. We partition the analysis of the number of iterations into two cases: (1) iterations in which $|S_\sigma| \leq n/3$ and (2) iterations in which $|S_\sigma| > n/3$. In the first case, using Fact 4.5, the number of such iterations is bounded by $\sum_{k=1}^{n/3} \binom{n}{k} \leq 2^{n \cdot H(1/3)}$ since no MAX strategy can repeat. In the second case, since $|S_\sigma| > n/3$ for each such iteration, by Lemma 4.11, the Hoffman-Karp algorithm discards at least $n/3$ strategies σ_i, better than the current strategy σ, in favor of the superior strategy σ' for the next iteration. Thus, the number of iterations in which $|S_\sigma| > n/3$ is bounded by $\frac{2^n}{n/3} = 3 \cdot \frac{2^n}{n}$. It follows that the Hoffman-Karp algorithm requires at most $2^{n \cdot H(1/3)} + 3 \cdot \frac{2^n}{n} \leq 4 \cdot \frac{2^n}{n}$ iterations in the worst case. ∎

4.3 A New Randomized Algorithm

We propose a Las Vegas (randomized) algorithm for solving simple stochastic games. Our algorithm can be seen as a variation of the Hoffman-Karp algorithm in that, instead of deterministically choosing a single switchable MAX vertex, our randomized algorithm chooses a uniformly random subset of switchable MAX vertices in each iteration. Similar to the results in the previous section, our results in this section are based on an extension of the proof technique of Mansour and Singh [MS99], which they applied to analyze a randomized policy iteration algorithm for Markov decison processes.

Our randomized algorithm is described as Algorithm 11.

Algorithm 11: Our Randomized Algorithm

Input : A stopping simple stochastic game G
Output: An optimal pair of strategies $\langle \sigma, \tau \rangle$ and the optimal value vector v_{opt}

```
1 begin
2     Let σ, τ be arbitrary strategies of players MAX and MIN, respectively
3     while (F_G(v_{σ,τ}) ≠ v_{σ,τ}) do
4         Choose a subset S of v_{σ,τ}-switchable MAX vertices uniformly at random
5         Let σ' be obtained from σ by switching all vertices of S
6         Let τ' be an optimal strategy of player MIN w.r.t. σ'
7         Set σ ← σ' and τ ← τ'
8     Output ⟨σ, τ⟩ and the optimal value vector v_{σ,τ}
9 end
```

Lemma 4.13 *For each iteration i in Algorithm 11, let $\langle \sigma_i, \tau(\sigma_i) \rangle$ be a pair of player strategies at the start of this iteration. Let $S_i \subseteq S_{\sigma_i}$ be a subset of $v_{\sigma_i,\tau(\sigma_i)}$-switchable* MAX *vertices and let $\sigma' = \text{switch}(\sigma_i, S_i)$ be a strategy of* MAX. *If $v_{\sigma',\tau(\sigma')} \not\succ v_{\sigma_{i+1},\tau(\sigma_{i+1})}$, then for any $j \geq i+2$, $\sigma' \neq \sigma_j$.*

Proof. By Lemma 4.7, for each iteration j, it holds that $v_{\sigma_j,\tau(\sigma_j)} \succ v_{\sigma_{j-1},\tau_{j-1}}$. Assume to the contrary that for some $j \geq i+2$, $\sigma' = \sigma_j$. Then, by transitivity, we have $v_{\sigma',\tau(\sigma')} = v_{\sigma_j,\tau(\sigma_j)}$ $\succ v_{\sigma_{j-1},\tau_{j-1}} \succeq v_{\sigma_{i+1},\tau(\sigma_{i+1})}$, which contradicts the given fact that $v_{\sigma',\tau(\sigma')} \not\succ v_{\sigma_{i+1},\tau(\sigma_{i+1})}$. ∎

Lemma 4.14 *In Algorithm 11, let $\langle \sigma, \tau(\sigma) \rangle$ be a pair of strategies at the start of an iteration, where S_σ is nonempty, and let $\langle \sigma', \tau(\sigma') \rangle$ be the pair of strategies at the end of this iteration.*

Then, the expected number of strategy pairs $\langle \sigma_i, \tau(\sigma_i) \rangle$ such that $v_{\sigma_i,\tau(\sigma_i)} \not\succ v_{\sigma',\tau'}$ is at least $2^{|S_\sigma|-1}$.

Proof. Consider an iteration in which $\langle \sigma, \tau(\sigma) \rangle$ is a pair of player strategies. Let U denote the set of all MAX strategies obtained by switching some subset of S_σ, the set of $v_{\sigma,\tau(\sigma)}$-switchable MAX vertices. Clearly, $|U| = 2^{|S_\sigma|}$. For each strategy $\alpha \in U$, we associate two sets: a set U_α^+ that contains all strategies $\beta \in U$ such that $v_{\beta,\tau(\beta)} \succ v_{\alpha,\tau(\alpha)}$, and a set U_α^- that contains all strategies $\beta \in U$ such that $v_{\alpha,\tau(\alpha)} \succ v_{\beta,\tau(\beta)}$. Note that for any pair α, $\beta \in U$, we have $\beta \in U_\alpha^+$ if and only if $\alpha \in U_\beta^-$. From this, it follows that

$$\sum_{\alpha \in U} |U_\alpha^+| = \sum_{\alpha \in U} |U_\alpha^-| \leq \frac{|U|^2}{2}.$$

Thus, for a strategy σ' chosen uniformly at random from U, the expected number of MAX strategies $\sigma_i \in U$ such that $v_{\sigma_i,\tau(\sigma_i)} \not\succ v_{\sigma',\tau'}$ is $|U| - \sum_{\alpha \in U} |U_\alpha^+|/|U| \geq |U|/2 = 2^{|S_\sigma|-1}$. ∎

Theorem 4.15 *With probability at least $1 - 2^{-2^{O(n)}}$, Algorithm 11 requires at most $O(2^{0.78n})$ iterations in the worst case, where $n = \min\{|V_{\text{MAX}}|, |V_{\text{MIN}}|\}$.*

Proof. Let $c \in (0, 1/2)$ that we will fix later in the proof. As in the proof of Theorem 4.12, we bound the number of iterations in which $\langle \sigma, \tau(\sigma) \rangle$ is a pair of player strategies and $|S_\sigma| \leq cn$ by $\sum_{k=0}^{cn} \binom{n}{k}$, which is at most $2^{n \cdot H(c)}$ by Fact 4.5.

We next bound the number of iterations in which $\langle \sigma, \tau(\sigma) \rangle$ is a pair of player strategies and $|S_\sigma| > cn$. Let $\langle \sigma, \tau(\sigma) \rangle$ ($\langle \sigma', \tau(\sigma') \rangle$) be a pair of player strategies at the start (respectively, end) of one such iteration. By Lemma 4.14, the expected number of strategy pairs $\langle \sigma_i, \tau(\sigma_i) \rangle$ such that $v_{\sigma_i,\tau(\sigma_i)} \not\succ v_{\sigma',\tau'}$ is at least $2^{|S_\sigma|-1}$. By Lemma 4.13, none of these strategy pairs $\langle \sigma_i, \tau(\sigma_i) \rangle$ is chosen in any future iteration. Therefore, the expected number of strategy pairs $\langle \sigma_i, \tau(\sigma_i) \rangle$ that Algorithm 11 discards in each such iteration is at least $2^{|S_\sigma|-1} \geq 2^{cn}$. It follows from Markov's inequality that with probability at least $1/2$, Algorithm 11 discards at least 2^{cn-1} strategy pairs in each such iteration.

We say that an iteration in which $|S_\sigma| > cn$ is *good* if Algorithm 11 discards at least 2^{cn-1} pairs of strategies at the end of it. We know from above that the probability that an iteration in which $|S_\sigma| > cn$ is good is at least $1/2$. By Chernoff bounds, for any $t > 0$, at least $1/4$

of the t iterations in which $|S_\sigma| > cn$ will be good with probability at least $1 - \exp(-t/16)$. Thus, the total number of iterations is at most $2^{n \cdot H(c)} + 2^{n \cdot (1-c)+3}$ with a high probability. For $c = 0.227$ and for sufficiently large n, this number of iterations is bounded by $2^{0.78n}$. Also, when the number of iterations (t) is $2^{0.78n}$, then the probability of success is $\geq 1 - 2^{-2^{O(n)}}$. ∎

CHAPTER 5

RELATED WORK, CONCLUSION, AND OPEN PROBLEMS

Parity games are defined in Section 2.4. The algorithmic problem of solving a parity game is: PARITY-GAME $\equiv$ Given a game $\mathcal{G} = (V_{\text{MAX}}, V_{\text{MIN}}, E, p)$ defined on a directed graph $G = (V, E)$ and a start vertex v_0, where $(V_{\text{MAX}}, V_{\text{MIN}})$ is a partition of $V(G)$ and $p : V(G) \to \mathbb{N}$ is a color function, determine whether player MAX has a winning strategy in the game if the token is initialy placed on vertex v_0. Recall that player MAX wins if the largest vertex color $p(v_i)$ of a vertex v_i occurring infinitely often is odd, i.e., if $\limsup_{i\to\infty} p(v_i)$ is odd, while player MIN wins if it is even. Here v_0, v_1, v_2, ... is the infinite path formed by the players. The best known deterministic algorithm for PARITY-GAME by Jurdziński, Paterson, and Zwick [JPZ08] runs in subexponential time. Some faster (randomized) strongly expected subexponential-time algorithms for this problem are by Björklund, Sandberg, and Vorobyov [BSV03] and by Halman [Hal07]. The randomized algorithm in [BSV03] is based on the randomized algorithm of Ludwig [Lud95] for simple stochastic games, which in turn is inspired by the subexponential randomized simplex algorithm for linear programming by Kalai [Kal92]. The randomized algorithm in [Hal07] is based on the known algorithm of Matoušek, Sharir, and Welzl [MSW96] for LP-type problems.

Mean payoff games and discounted payoff games are also defined in Section 2.4. The algorithmic problems for these games are defined similar to the problem PARITY-GAME. The best known (randomized) strongly expected subexponential-time algorithm for solving mean payoff games is by Halman [Hal07], which improves upon the randomized algorithm by Björklund, Sandberg, and Vorobyov [BSV07]. The best known algorithm for solving discounted payoff games is the (randomized) strongly expected subexponential-time algorithm by Halman [Hal07].

The best known algorithms for solving simple stochastic games are by Ludwig [Lud95] and by Halman [Hal07]. Gärtner and Rüst [GR05] showed that finding optimal strategies for players in a simple stochastic game polynomial-time reduces to the generalized linear complementarity problem (GLCP) with a P-matrix. The hardness of solving a simple stochastic game is addressed in [Jub05].

The algorithms surveyed in this thesis are summarized in Table 5.1.

Table 5.1 Summary of Algorithms for Simple Stochastic Games

Method	Algorithm	Running Time
Iterative Approximation	by Somla	$2^{O(n)}$
	by Shapley	$2^{O(n)}$
	"Converge from Below" by Condon	$2^{O(n)}$
Strategy Improvement	The Hoffman-Karp	$2^{O(n)}$
Mathematical Programming	A Quadratic Program	$2^{O(n)}$
	Linear Programs for Restricted SSGs	$n^{O(1)}$
Randomized	A Randomized Variant of the Hoffman-Karp	$2^{O(n)}$
	by Ludwig	$2^{O(\sqrt{n})}$
	Our Algorithm	$2^{O(n)}$

In this thesis, we obtained an improved worst-case, upper bound on the number of iterations required by the Hoffman-Karp strategy improvement algorithm. We also presented a randomized Las Vegas strategy improvement algorithm whose expected running time is $O(2^{0.78n})$.

One interesting direction for future work would be to find a super-polynomial lower bound on the number of iterations required by the Hoffman-Karp algorithm in the worst case. Another interesting direction would be to conduct experimental work related to comparing algorithms for their running time performance. The possibility of a polynomial-time algorithm, or even an improved deterministic subexponential-time algorithm, cannnot be ruled out for the simple stochastic game value problem. Obtaining an algorithm of this sort would be a challenging research direction.

REFERENCES

[AKP+02] A. Akella, R. Karp, C. Papadimitrou, S. Seshan, and S. Shenker. Selfish behavior and stability of the internet: A game-theoretic analysis of TCP. In *SIGCOMM '02*, 2002.

[BSV03] H. Björklund, S. Sandberg, and S. Vorobyov. A discrete subexponential algorithm for parity games. In *Proceedings of the 20th Annual Symposium on Theoretical Aspects of Computer Science*, pages 663–674. Springer Verlag *Lecture Notes in Computer Science #2607*, 2003.

[BSV07] H. Björklund, S. Sandberg, and S. Vorobyov. A combinatorial strongly subexponential strategy improvement algorithm for mean payoff games. *Discrete Applied Mathematics*, 155:210–229, 2007.

[Con92] A. Condon. The complexity of stochastic games. *Information and Computation*, 96:203–224, 1992.

[Con93] A. Condon. On algorithms for simple stochastic games. In J. Cai, editor, *Advances in Computational Complexity Theory*, volume 13, pages 51–73. DIMACS series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, 1993.

[d'E63] F. d'Epenoux. A probabilistic production and inventory problem. *Management Science*, 10(1):98–108, 1963.

[Der70] C. Derman. *Finite State Markovian Decision Processes*, volume 67 of *Mathematics in Science and Engineering*. Academic Press, New York, 1970.

[EM79] A. Ehrenfeucht and J. Mycielski. Positional strategies for mean payoff games. *International Journal of Game Theory*, 8:109–113, 1979.

[Fil81] J. Filar. Ordered field property for stochastic games when the players who controls transition changes from state to state. *Journal of Optimization Theory and Applications*, 34(4):503–515, 1981.

[FS86] J. Filar and T. Schultz. Nonlinear programming and stationary strategies in stochastic games. *Mathematical Programming*, 35:243–247, 1986.

[FV97] J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer-Verlag, 1997.

[GKK88] V. Gurvich, A. Karzanov, and L. Khachiyan. Cyclic games and an algorithm to find minimax cycle means in directed graphs. *USSR Computational Mathematics and Mathematical Physics*, 28:85–91, 1988.

[GR05] B. Gärtner and L. Rüst. Simple stochastic games and P-matrix generalized linear complementarity problems. In *Proceedings of the 15th Conference on Fundamentals of Computation Theory*, pages 209–220. Springer Verlag *Lecture Notes in Computer Science #3623*, 2005.

[GW02] E. Grädel and T. Wolfgang. Automata, logics, and infinite games. *Lecture Notes in Computer Science*, 2500, 2002.

[Hal07] N. Halman. Simple stochastic games, parity games, mean payoff games and discounted payoff games are all LP-type problems. *Algorithmica*, 49(1):37–50, 2007.

[HK66] A. Hoffman and R. Karp. On nonterminating stochastic games. *Management Science*, 12:359–370, 1966.

[How60] R. Howard. *Dynamic Programming and Markov Processes*. M.I.T. Press, Cambridge, MA, 1960.

[JPZ08] M. Jurdziński, M. Paterson, and U. Zwick. A deterministic subexponential algorithm for solving parity games. *SIAM Journal on Computing*, 38(4):1519–1532, 2008.

[Jub05] B. Juba. On the hardness of simple stochastic games. Master's thesis, Carnegie Mellon University, 2005.

[Juk01] S. Jukna. *Extremal Combinatorics*. Springer, 2001.

[Kal92] G. Kalai. A subexponential randomized simplex algorithm. In *Proceedings of the 24th ACM Symposium on Theory of Computing*, pages 475–482. ACM press, 1992.

[Kar84] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4:373–397, 1984.

[Kha79] L. Khachiyan. A polynomial algorithm in linear programming. *Soviet Math. Dokl.*, 20:191–194, 1979.

[KT04] V. Kumar and R. Tripathi. Algorithmic results in simple stochastic games. Technical Report TR855, Department of Computer Science, University of Rochester, November 2004.

[Lud95] W. Ludwig. A subexponential randomized algorithm for the simple stochastic game problem. *Information and Computation*, 117(1):151–155, 1995.

[MC90] M. Melekopoglou and A. Condon. On the complexity of the Policy Iteration algorithm for simple stochastic games. Technical Report TR941, Department of Computer Science, University of Wisconsin-Madison, June 1990.

[MGLA00] M. Mundhenk, J. Goldsmith, C. Lusena, and E. Allender. Complexity of finite-horizon markov decision process problems. *Journal of the ACM*, 47(4):681–720, 2000.

[MS99] Y. Mansour and S. Singh. On the complexity of Policy Iteration. In *Proceedings of the 15th Conference on Uncertainity in Artificial Intelligence, Stockholm, Sweden*, pages 401–408. Morgan Kaufmann, July 1999.

[MSW96] J. Matoušek, M. Sharir, and E. Welzl. A subexponential bound for linear programming. *Algorithmica*, 16(4/5):498–516, 1996.

[Sha53] L. Shapley. Stochastic games. In *Proceedings of National Academy of Sciences (U.S.A.)*, volume 39, pages 1095–1100, 1953.

[Som05] R. Somla. New algorithms for solving simple stochastic games. *Electronic Notes in Theoretical Computer Science*, 119(1):51–65, 2005.

[SS95] S. Shenker and J. Scott. Making greed work in networks: a game-theoretic analysis of switch service disciplines. *IEEE/ACM Trans. Netw.*, 3(6):819–831, 1995.

[Yao77] A. Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th IEEE Symposium on Foundations of Computer Science*, pages 222–227, 1977.

[ZP96] U. Zwick and M. Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158:343–359, 1996.