2025 IEEE 64th Conference on Decision and Control (CDC)
December 10-12, 2025. Rio de Janeiro, Brazil

Objective Improvement Algorithm for Controller Synthesis in Uncertain Environments

Daniele Dell'Erba, Sven Schewe, and Ashutosh Trivedi

2025 IEEE 64th Conference on Decision and Control (CDC) | 979-8-3315-2627-6/25/$31.00 ©2025 IEEE | DOI: 10.1109/CDC57313.2025.11312600

***Abstract*— *Stochastic games* provide a powerful framework for controller synthesis in multi-agent systems where cooperation between agents cannot be assumed. They also serve as a core model for *modular and decentralized control*, where interactions between components can be captured using assume-guarantee contracts. In this setting, synthesis for an individual module reduces to computing a policy robust to the behavior of the environment, modeled as a stochastic two-player game. Many control objectives reduce to *reachability objectives*, leading to the study of *simple stochastic games*, a well-known class whose exact computational complexity remains unresolved.**

A classic result of Condon reformulates the value and policy computation in such games as a *quadratic program*—a linear program with a quadratic objective. Motivated by their "almost linear" structure, we ask whether efficient linear programming solvers can be leveraged by iterating over a sequence of linear objectives. We introduce the *Objective Improvement Algorithm*, which iteratively solves linear programs to compute the optimal value and policy. Unlike strategy improvement, our method treats both players symmetrically, and unlike value iteration, it terminates with the optimal value in finitely many steps. We prove convergence and correctness and present experimental results demonstrating practical effectiveness.

I. Introduction

The correct-by-construction synthesis of controllers in uncertain environments is central to the promise of formal methods for safety-critical cyber-physical systems (CPS). Irrespective of the computability challenges associated with formal reasoning for CPS with even modest physical dynamics, a broad range of principled approaches—based on symbolic abstractions, functional certificates, and constraint satisfaction—have been developed to provide formal correctness guarantees for the synthesized controllers. However, the *curse of dimensionality* remains the central challenge in scaling these approaches to practical systems. In response, modular synthesis has emerged as a promising technique.

In modular synthesis, the core objective is to decompose the overall problem into smaller, interconnected components such that the synthesis of each component can be performed in isolation. This is achieved by carefully partitioning the global specification into local assume-guarantee contracts, which naturally give rise to stochastic two-player games as the canonical formulation. In these games, the objective of one player (the controller) is to find a strategy that satisfies its guarantee obligations, irrespective of the choices made by the environment (other modules), as long as those choices ensure that the module's assumptions are upheld. This separation of concerns allows control synthesis to be performed over a potentially much smaller component, albeit at the cost of game-theoretic reasoning. Thus, improved algorithmic results for solving stochastic two-player games have the potential to significantly advance controller synthesis for large-scale cyber-physical systems. This paper presents a novel algorithm to solve stochastic games.

Algorithms for Stochastic Games. Stochastic games [1] are a fundamental model for strategic interactions among agents operating in uncertain environments. Since the seminal work of Ramadge and Wonham [2], they have also served as a framework for controller synthesis. Many strategic objectives reduce to reachability objectives, leading to the class of *simple stochastic games* [3], whose computational complexity remains unresolved. Condon [4] proposed both a value iteration algorithm—based on iterating the Bellman optimality operator—and a strategy (or policy) improvement algorithm, in which one player myopically optimizes their strategy while the opponent computes a globally optimal counter-strategy (the stable response). The algorithmic complexity of solving stochastic games is known to lie in $\mathsf{NP} \cap \mathsf{CoNP}$ (and even in $\mathsf{UP} \cap \mathsf{CoUP}$ [5])[1], yet no polynomial-time algorithm is known for stochastic games.

While value iteration is often the preferred practical method for solving stochastic games, it only converges in the limit, and no general stopping criterion is known [7]. The strategy improvement algorithm, on the other hand, is guaranteed to converge to the optimal value, but its runtime can only be bounded by an exponential function in the size of the game. A further limitation of strategy improvement is its lack of symmetry in addressing what is inherently a symmetric problem: stochastic games are formulated with two players, Min and Max, whose objectives are directly opposed but structurally identical. Yet the algorithm improves the strategy of only one player at a time, while the opponent's strategy is recomputed as a best response, rather than being updated simultaneously. This asymmetry in treatment contrasts with the symmetric structure of the game itself.

A. Trivedi is a Royal Society Wolfson Visiting Fellow and gratefully acknowledges the support of the Wolfson Foundation and the Royal Society.

This research was supported in part by the National Science Foundation (NSF) under CAREER Award CCF-2146563 and by the Engineering and Physical Sciences Research Council (EPSRC) through grant EP/X03688X/1.

D. Dell'Erba and S. Schewe are with the Department of Computer Science at the University of Liverpool, while A. Trivedi is affiliated with the Department of Computer Science at the University of Colorado Boulder. Emails: dde@liverpool.ac.uk, sven.schewe@liverpool.ac.uk, Ashutosh.Trivedi@colorado.edu

[1] NP is the class of problems whose membership can be verified in polynomial time given a short certificate, and CoNP consists of their complements. UP (unambiguous polynomial time) is the subclass of NP in which each instance has at most one certificate verifiable in polynomial time (Papadimitriou [6, pp. 283–284]), and CoUP denotes their complements.

Objective Improvement Algorithm. The problem of determining the value and an optimal policy pair in stochastic games can also be formulated as a *quadratic program*—a linear program with a quadratic objective—via a reduction introduced by Condon [4]. Klingler [8] conducted an empirical evaluation of algorithms for solving stochastic games and observed that, in one-player games (MDPs in the stable-response step), linear-programming variations generally outperform value iteration, except in certain corner cases identified in that work. Moreover, Klingler showed that the quadratic-programming formulation can become computationally infeasible, sometimes requiring hours to solve games with as few as ten vertices [8].

Building on this empirical observation, we ask whether linear programming can be used to solve the quadratic-programming formulation of stochastic games by iteratively improving the linear objective functions. We refer to this approach as the *Objective Improvement Algorithm*. Since our algorithm repeatedly invokes linear programming, and practical solvers often rely on the popular and performant *Simplex algorithm* [9], we must carefully address the issue of degeneracy [10] in our LP formulations. To that end, we provide a novel characterization of the problem as a discounted reachability problem, specifically designed to mitigate degeneracy in the solutions. We analyze the resulting randomized algorithm and prove its almost-sure convergence. Finally, we present preliminary experimental results demonstrating the viability of the proposed approach.

II. Problem Definition

A *probability distribution* over a finite set S is a function $d\colon S\to[0,1]$ such that $\sum_{s\in S} d(s) = 1$. Let $\mathcal{D}(S)$ denote the set of all discrete distributions over S. We say a distribution $d \in \mathcal{D}(S)$ is a *point distribution* if $d(s)=1$ for some $s \in S$. For $d \in \mathcal{D}(S)$ we write $supp(d)$ for $\{s \in S\colon d(s) > 0\}$.

Definition 2.1 (Stochastic Game Arena (SGA)): An SGA $\mathcal{G}$ is a tuple $(S, A, T, S_{\rm Min}, S_{\rm Max})$, where S is a finite set of states, A is a finite set of *actions*, $T\colon S\times A\to\mathcal{D}(S)$ is a *transition function*, and $\{S_{\rm Min}, S_{\rm Max}\}$ is a partition of S.

By abusing notation, we call an SGA a Markov decision process (MDP) if $S_{\rm Min} = \emptyset$. For $s \in S$, the set $A(s)$ denotes the set of actions that can be selected in the state s. For states $s, s' \in S$ and $a \in A(s)$ we write $p(s'|s, a)$ for $T(s, a)(s')$.

A *run* of $\mathcal{G}$ is an ω-word $\langle s_0, a_1, s_1, \ldots\rangle \in S \times (A \times S)^\omega$ such that $p(s_{i+1}|s_i, a_{i+1}) > 0$ for all $i \geq 0$. A finite run is a finite such sequence, that is, a word in $S \times (A \times S)^*$. We write $Runs^{\mathcal{G}}(FRuns^{\mathcal{G}})$ for the set of runs (finite runs) of the SGA $\mathcal{G}$ and $Runs^{\mathcal{G}}(s)(FRuns^{\mathcal{G}}(s))$ for the set of runs (finite runs) of the SGA $\mathcal{G}$ starting from state s. We denote by $last(r)$ the last state of a finite run r.

Semantics. A game on an SGA $\mathcal{G}$ starts with a token in an *initial state* $s \in S$; players $\rm Min$ and $\rm Max$ construct an infinite run by choosing enabled actions, when the run arrives in a state controlled by them, and then moving the token to a successor state sampled from the selected distribution.

A strategy of player Min in $\mathcal{G}$ is a partial function $\mu\colon FRuns \to \mathcal{D}(A)$, defined for $r \in FRuns$ if and only if $last(r) \in S_{\rm Min}$, such that $supp(\sigma(r)) \subseteq A(last(r))$. A strategy ν of player Max is defined analogously. A strategy σ is *pure* if $\sigma(r)$ is a point distribution wherever it is defined; otherwise, σ is *mixed*. We say that σ is *stationary* if $last(r) = last(r')$ implies $\sigma(r) = \sigma(r')$ wherever σ is defined. A strategy is *positional* if it is both pure and stationary. Let $\Sigma_{\rm Min}$ and $\Sigma_{\rm Max}$ be the sets of all strategies of player Min and player Max, respectively. Similarly, $\Pi_{\rm Min}$ and $\Pi_{\rm Max}$ denote the sets of all *positional* strategies of player Min and player Max, respectively.

Probability Space. For $(\mu, \nu) \in \Sigma_{\rm Min} \times \Sigma_{\rm Max}$, let $Runs^{\mathcal{G}}_{\mu,\nu}(s)$ denote the set of infinite runs of $\mathcal{G}$ starting at s that are consistent with (μ, ν). We write $\mathcal{F}_{Runs^{\mathcal{G}}_{\mu,\nu}(s)}$ for the sigma-algebra generated by cylinder sets, where a cylinder set is the collection of all infinite runs sharing a given finite prefix. The behavior of $\mathcal{G}$ under (μ, ν) is then described by the probability space $(Runs^{\mathcal{G}}_{\mu,\nu}(s), \mathcal{F}_{Runs^{\mathcal{G}}_{\mu,\nu}(s)}, \Pr^{\mathcal{G}}_{\mu,\nu}(s))$, where the measure $\Pr^{\mathcal{G}}_{\mu,\nu}(s)$ is uniquely determined by the Ionescu–Tulcea extension theorem, which extends the measure from cylinder sets to all infinite runs. Given a random variable $f\colon Runs^{\mathcal{G}} \to \mathbb{R}$ over the infinite runs of $\mathcal{G}$, we denote by $\mathbb{E}^{\mathcal{G}}_{\mu,\nu}(s)\{f\}$ the expectation of f over the runs in the probability space $(Runs^{\mathcal{G}}_{\mu,\nu}(s), \mathcal{F}_{Runs^{\mathcal{G}}_{\mu,\nu}(s)}, \Pr^{\mathcal{G}}_{\mu,\nu}(s))$. We denote by X_n $(n \geq 0)$ the random variable corresponding to the n-th state, and by Y_n $(n \geq 1)$ the random variable corresponding to the n-th action.

Payoff Functions. We use the following payoff functions over stochastic game arenas (SGA) $\mathcal{G}$ in this paper.

- A *stochastic reachability payoff* is characterized by two distinguished sink[2] states: the *accepting sink* $s_a \in S$ and the *rejecting sink* $s_r \in S$. The stochastic reachability payoff $Reach^{\mathcal{G}}_{\mu,\nu}(s)$ for a strategy pair (μ, ν) from state $s \in S$ is the probability of reaching s_a, i.e.,
$$Reach^{\mathcal{G}}_{\mu,\nu}(s) = \Pr{}^{\mathcal{G}}_{\mu,\nu}(s)\big\{r{\in}Runs^{\mathcal{G}}_{\mu,\nu}(s){:}r \text{ visits } s_a\big\}.$$
- A *discounted payoff* is characterized by a discount factor $\lambda \in [0, 1)$. The discounted payoff $Disct^{\mathcal{G}}_{\mu,\nu}(s)$ for (μ, ν) from state $s \in S$ is defined as
$$Disct^{\mathcal{G}}_{\mu,\nu}(s) = \mathbb{E}^{\mathcal{G}}_{\mu,\nu}(s)\Big\{\sum_{n=0}^{\infty} \lambda^n R(X_n, Y_{n+1})\Big\}.$$

In a stochastic game $\mathcal{G}$, the objective of player Max is to maximize the payoff $\mathcal{P} \in \{Disct, Reach\}$, while the objective of player Min is the opposite. For every state $s \in S$, we define its *upper value* $\mathsf{UVal}(\mathcal{G}, s)$ as the minimum discounted payoff player Min can ensure irrespective of player Max's strategy. Similarly, the *lower value* $\mathsf{LVal}(\mathcal{G}, s)$ of a state $s \in S$ is the maximum discounted payoff player Max can ensure irrespective of player Min's strategy, i.e.,
$$\mathsf{UVal}(\mathcal{G}, s) = \inf_{\mu\in\Sigma_{\rm Min}} \sup_{\nu\in\Sigma_{\rm Max}} \mathcal{P}^{\mathcal{G}}_{\mu,\nu}(s) \text{ and}$$
$$\mathsf{LVal}(\mathcal{G}, s) = \sup_{\nu\in\Sigma_{\rm Max}} \inf_{\mu\in\Sigma_{\rm Min}} \mathcal{P}^{\mathcal{G}}_{\mu,\nu}(s).$$

[2]Recall that a sink state s satisfies $p(s|s, a) = 1$ for all $a \in A$.

The inequality $\mathsf{LVal}(\mathcal{G}, s) \leq \mathsf{UVal}(\mathcal{G}, s)$ holds for all two-player zero-sum games. A game is called *determined* when, for every state $s \in S$, its lower value and its upper value are equal; we then say that the value of the game Val exists and $\mathsf{Val}(\mathcal{G}, s) = \mathsf{LVal}(\mathcal{G}, s) = \mathsf{UVal}(\mathcal{G}, s)$ for every $s \in S$. For a strategy $\mu \in \Sigma_{\mathrm{Min}}$ and $\nu \in \Sigma_{\mathrm{Max}}$, we define their values as:

$$\mathsf{Val}^{\mu} : s \mapsto \sup_{\nu \in \Sigma_{\mathrm{Max}}} \mathcal{P}^{\mathcal{G}}_{\mu,\nu}(s) \text{ and}$$
$$\mathsf{Val}^{\nu} : s \mapsto \inf_{\mu \in \Sigma_{\mathrm{Min}}} \mathcal{P}^{\mathcal{G}}_{\mu,\nu}(s).$$

We say that a strategy $\mu_* \in \Pi_{\mathrm{Min}}$ of player Min is optimal if $\mathsf{Val}^{\mu_*} = \mathsf{Val}$. Similarly, a positional strategy $\nu_* \in \Pi_{\mathrm{Max}}$ of player Max is optimal if $\mathsf{Val}^{\nu_*} = \mathsf{Val}$. We say that a game is *positionally determined* if both players have positional optimal strategies. In his seminal work [1], Shapley showed that stochastic games with discounted objectives on finite arenas are positionally determined. Condon [4] studied the complexity of a class of stochastic games (simple stochastic games) with reachability objectives and showed that it lies in NP and in co-NP. In this work, we study stochastic games with reachability objectives and study the following problem.

Definition 2.2 (Stochastic Reachability Games): Given a stochastic game arena $\mathcal{G} = (S, A, T, S_{\mathrm{Min}}, S_{\mathrm{Max}})$ with a reachability objective specified by an accepting sink $s_a \in S$ and a rejecting sink $s_r \in S$, determine the optimal value and optimal positional strategies for both players.

Our approach to solving stochastic games with reachability objectives consists in reducing them to stochastic games with discounted objectives. Following a construction similar to that of Condon [4], we then solve these discounted objectives using the objective improvement algorithm.

III. Solving Stochastic Reachability Games

We begin with a small running example that, while simple, illustrates several of the key challenges that must be addressed. The example has two states: a player Min state s_{min} with a single outgoing transition that always leads to the player Max state s_{max}. At s_{max}, player Max can either end the game—resulting in a 50/50 chance of winning or losing—or return to s_{min}. This creates a loop in which player Max may remain indefinitely or eventually choose to stop the game. Since we are considering reachability objectives, player Max loses if he stays in the loop forever. Thus, his optimal strategy is to (eventually) take the 50/50 chance.

Bellman Optimality Equations. We characterize the optimal value of stochastic reachability games on SGA $\mathcal{G} = (S, A, T, S_{\mathrm{Min}}, S_{\mathrm{Max}})$ with accepting and rejecting sinks $s_a, s_r \in S$, using Bellman optimality equations as follows:

$$\mathcal{R}(s) = \begin{cases} 0 & \text{if } s = s_r \\ 1 & \text{if } s = s_a \\ \max\limits_{a \in A(s)} \lambda \sum\limits_{s' \in S} p(s'|s,a)\mathcal{R}(s') & \text{if } s \in S_{\mathrm{Max}} \\ \min\limits_{a \in A(s)} \lambda \sum\limits_{s' \in S} p(s'|s,a)\mathcal{R}(s') & \text{if } s \in S_{\mathrm{Min}} \end{cases}$$

When the discount factor λ lies in $[0, 1)$, the Bellman equations admit a unique, well-defined solution. However, when $\lambda = 1$, the solutions to the Bellman optimality equations are no longer unique. For instance, in the case of our running example, the reader may verify that any assignment to $\mathcal{R}(s_{\mathrm{min}})$ and $\mathcal{R}(s_{\mathrm{max}})$ that is greater than or equal to 0.5 is a solution of these equations. This challenge can be tackled by ensuring that all values are non-negative and then by looking for the smallest valuation that satisfies the Bellman equations, or by adding contraction λ close to 1. It is well known that, when λ is close to 1, optimal solutions for the resulting discounted game are also optimal for the original game, and optimal values for the discounted and non-discounted games are close.

Preprocessing (Qualitative Solution). A first step in solving a simple stochastic game is to identify the almost sure winning regions of both players. For this, we can translate the stochastic resolution to choice: the player Min can surely avoid reaching the accepting sink iff she can avoid reaching it when the player Max can choose the successor from the support of $T(s, a)$. The player Max can win almost surely in the maximal set $M \subseteq S \cup S \times A$ such that 1) he can surely stay in this set (even if the player Min resolves the random choices) from every state and every state action pair in M, and (2) can reach the accepting sink if the player Max itself resolves the random choices. The former can be checked by solving a reachability game, the latter by solving a Büchi game [11, 12]. After determining the almost sure winning regions of both players, we remove all states of their winning region from T, re-routing them to accepting sink with fixed value 1 for almost sure winning states of the player Max, and making the matrix sub-stochastic where it had successors in the sure winning region of the player Min.

Polynomial Programming. A solution to the Bellman optimality equations can be characterized as a polynomial program (or a *quadratic* program when each transition has at most two successors—as in "simple" stochastic games of Condon [4]) by expressing the equations as linear constraints together with a "polynomial" objective function. The linear program uses the inequations:

$$\begin{aligned} R(s_r) &= 0 \text{ and } R(s_a) = 1 \\ R(s) &\geq \sum_{s' \in S} p_\lambda(s'|s,a) \cdot R(s'), \text{ for } s \in S_{\mathrm{Max}}, a \in A(s) \\ R(s) &\leq \sum_{s' \in S} p_\lambda(s'|s,a) \cdot R(s'), \text{ for } s \in S_{\mathrm{Min}}, a \in A(s) \end{aligned}$$

where $p_\lambda(s'|s,a)$ equals:

$$\begin{cases} p(s'|s,a) \cdot \lambda & \text{if } s' \neq s_a \text{ and } s' \neq s_r \\ p(s'|s,a) \cdot \lambda + \alpha(s,a) & \text{if } s' = s_a. \\ 1 - \sum\limits_{t \neq s_r} p_\lambda(t|s,a) & \text{if } s' = s_r. \end{cases}$$

Here $\alpha(s, a)$ is a hyperparameter chosen in the range $[0, 1-\lambda]$. The values $\alpha(s, a)$ need not be the same for every transition; in fact, we draw them uniformly at random. These values govern the termination of a run: setting $\alpha(s, a) = 0$ corresponds to the standard contraction case, where the maximizer loses immediately with probability $1-\lambda$ and proceeds

with the normal transition with probability λ. Conversely, setting $\alpha(s,a) = 1-\lambda$ corresponds to the maximizer winning immediately with probability $1-\lambda$ and proceeding with the normal transition with probability λ. These two extremes slightly under- and overestimate the likelihood of winning, respectively. By sampling intermediate values, we obtain a better approximation without sacrificing precision.

We transform these inequalities into normal form by introducing a slack variable $\mathbf{S}(s,a)$ with constraint

$$\mathbf{S}(s,a) \geq 0$$

for each state–action pair $s \in S$ and $a \in A(s)$. The inequalities then become sharp:

$$R(s) - \sum_{s' \in S} p_\lambda(s'|s,a) \cdot R(s') \quad = \quad \mathbf{S}(s,a), \quad \text{if } s \in S_{\text{Max}},$$
$$R(s) - \sum_{s' \in S} p_\lambda(s'|s,a) \cdot R(s') \quad = \quad -\mathbf{S}(s,a), \quad \text{if } s \in S_{\text{Min}}.$$

We refer to this system of inequalities as $\mathsf{Inequations}(\mathcal{G})$.

Now satisfying the Bellman equations then reduces to satisfying the following constraint $\sum_{s \in S} \prod_{a \in A(s)} \mathbf{S}(s,a) = 0$, in addition to the linear equations, as all slack variables have non-negative value and the Bellman equations are satisfied if, for every state, (at least) one slack variable has value 0. This can also be phrased as the minimization problem:

$$\text{minimize} \sum_{s \in S} \prod_{a \in A(s)} \mathbf{S}(s,a) \ .$$

When there are only two actions, this is quadratic programming; as the restriction to two actions is a minor technicality (more actions can be modeled as a sequence of binary choices), this problem was initially phrased as quadratic programming (instead of higher order polynomial programming) by Condon [4].

Objective Improvement Algorithm. As solving polynomial programs is hard, we instead solve a sequence of linear programs. The idea behind this reduction is to home in on the Min-Max joint optimal policy σ^* that maps every state s to its optimal action $\sigma^*(s)$. We call such policy *co-optimal*. Of course, we initially do not know σ^*, but we can start with any function $\sigma_0 \colon S \to A$ (the function $\mathsf{ChooseInitialStrategies}(\mathcal{G})$) and then create a sequence of increasingly better policies $\sigma_1, \sigma_2, \ldots$ until we have found the co-optimal policy σ^*. For a given policy σ_i we use the objective (the function $\mathsf{ObjectiveFunction}(\mathcal{G}, \sigma)$) as:

$$\text{minimize} \sum_{s \in S} \beta(s, \sigma_i(s)) \cdot \mathbf{S}(s, \sigma_i(s)),$$

where the $\beta(s, \sigma_i(s)) > 0$ is another hyperparameter randomly drawn from a continuous domain. The pseudocode of our algorithm is shown in Algorithm 1.

Avoiding Degeneracy. To solve the individual linear programs, we use the Gurobi library [13], which implements a fast Simplex-based algorithm. While Simplex algorithms perform well in practice, they can sometimes stall on hard instances, particularly in the presence of degeneracy [10].

Algorithm 1: Objective Improvement Algorithm

input : An SGA $(S, A, T, S_{\text{Min}}, S_{\text{Max}})$
output: The optimal policy pair for $\mathcal{G}$

```
H ← Inequations(G)
σ ← ChooseInitialStrategies(G)
while true do
    f_σ ← ObjectiveFunction(G, σ)
    V ← LinearProgramming(H, f_σ)
    if f_σ(V) = 0 then
        return V
    σ ← ChooseBetterStrategies(G, V, σ)
```

A common source of degeneracy arises when a corner of the simplex has more slack variables at zero (i.e., tight inequalities) than actual variables. A standard remedy is to add small perturbations (noise) to the constant vector b in the constraints $Ax \leq b$. Since we already apply contraction with a discount factor $\lambda<1$ (close to 1) to ensure uniqueness of the solution, we draw perturbations independently and uniformly at random from the interval $[0, 1-\lambda]$ and add them to each transition probability $T(s,a)(s_a)$. This introduces a small amount of stochasticity, effectively transforming the game into a stochastic discounted-payoff game. We adapt the recent method of [14], originally developed for non-stochastic discounted-payoff games, to this stochastic variant. The introduction of hyperparameters α and β is motivated by the need to incorporate this perturbation and thereby avoid degeneracy in our Simplex-based formulation. We now establish the almost-sure uniqueness of a basis for a solution.

Lemma 1: If the hyperparameters $\alpha(s,a)$ are drawn independently and uniformly at random from the interval $[0, 1-\lambda]$, then, with probability one, no solution to the system of inequalities has more than $|S|+1$ constraints that are simultaneously tight.

Proof: Assume, for contradiction, that there exist $|S|$ inequalities (which we refer to as *defining inequalities*) and an additional, distinct inequality (the *testing inequality*) such that the following occurs with positive probability:

1) The defining inequalities, when treated as equalities, form a system of full rank.
2) The testing inequality is tight (i.e., holds with equality).

If condition 1) holds, then the solution to the system defined by the defining inequalities is uniquely determined. Consequently, there exists at most one value that can be drawn and added to $T(s,a)(s_a)$ in the testing inequality to make it tight as well. Since the draw is made from a continuous distribution (uniform over $[0, 1-\lambda]$), the probability of hitting exactly this value is zero. As this holds for all combinations of defining and testing inequalities, the lemma follows. ■

The random drawing of the hyperparameters $\beta(s,a)$ ensures almost surely that no two corners of the polytope have the same value of the objective function.

Lemma 2: Two adjacent corners of the polytope almost

surely have different values of the objective function:

$$\text{minimize} \quad \sum_{s \in S} \beta(s, \sigma_i(s)) \cdot \mathbf{S}(s, \sigma_i(s)).$$

Proof: For two adjacent corners of the polytope, at least one of the variables $\mathbf{S}(s, \sigma_i(s))$ must differ; let $s' \in S$ be one such state. The coefficients $\beta(s, \sigma_i(s))$ are drawn independently at random from a continuous domain. Without loss of generality, assume that $\beta(s', \sigma_i(s'))$ is drawn last. Then, conditioned on the other values, there exists at most one specific value of $\beta(s', \sigma_i(s'))$ for which the objective function remains unchanged across the two corners. Since this value is drawn from a continuous distribution, the probability of selecting it is zero. Hence, with probability one, the objective function assigns different values to the corners. The proof is now complete. ∎

Taken together, these observations almost surely eliminate all sources of degeneracy in our simplex algorithms.

Implementing ChooseBetterStrategies$(\mathcal{G}, V, \sigma)$**.** We begin with an arbitrary strategy for both players to define the initial joint policy σ_0. The approach described here generalizes the non-stochastic setting of [14]. Our call to the linear programming solver yields the optimal value for a fixed joint policy σ_i. If the objective function evaluates to 0, then σ_i is co-optimal, and we have found optimal strategies for both players. Otherwise, we improve σ_i to obtain a new joint policy σ_{i+1}. The goal in moving from σ_i to σ_{i+1} is to strictly decrease the objective value:

$$\sum_{s \in S} \beta(s, \sigma_{i+1}(s)) \cdot \mathbf{S}(s, \sigma_{i+1}(s));$$

compared to the previous one $\sum_{s \in S} \beta(s, \sigma_i(s)) \cdot \mathbf{S}(s, \sigma_i(s))$.

To achieve this, we update the policy in such a way that the new objective value is smaller than the previous one at the corner of the simplex that minimizes the original objective. This is not only the typical case but also easy to verify in practice: for each state $s \in S$, we can simply choose an action $a \in A(s)$ that minimizes the product $\beta(s, a) \cdot \mathbf{S}(s, a)$. If σ_i is locally optimal, i.e., if $\mathbf{S}(s, \sigma_i(s)) = 0$ holds for all states $s \in S$, then it is co-optimal. Otherwise, we identify a strictly better policy corresponding to a neighboring vertex.

Lemma 3: If σ_i cannot be locally improved yet is not optimal, then—under the random choices for the hyperparameters α and β—there is almost surely a local improvement that yields a better objective value at a corner of the simplex neighboring the optimal solution for σ_i.

Proof: Let V be a minimal solution to the objective $\sum_{s \in S} \beta(s, \sigma_i(s)) \cdot \mathbf{S}(s, \sigma_i(s))$ at a corner of the simplex, and assume that this value is nonzero (i.e., σ_i is not yet a co-optimal strategy) and that no local improvements are possible. We now consider a simplified game that retains all transitions either selected by σ_i or corresponding to tight inequalities (i.e., transitions for which the slack variable satisfies $\mathbf{S}(s, a) = 0$). This simplified game clearly has an optimal solution, say V', attained by a joint policy σ'. For this solution, we have: $\sum_{s \in S} \beta(s, \sigma'(s)) \cdot \mathbf{S}(s, \sigma'(s))$. Moreover, it holds that:

$$\sum_{s \in S} \beta(s, \sigma_i(s)) \cdot \mathbf{S}(s, \sigma_i(s)) = \sum_{s \in S} \beta(s, \sigma'(s)) \cdot \mathbf{S}(s, \sigma'(s)).$$

Although V' may not satisfy all the original inequalities, the convex combination $\varepsilon \cdot V' + (1-\varepsilon) \cdot V$ does, for sufficiently small $\varepsilon > 0$. Therefore, we conclude the following:

- Minimizing $\sum_{s \in S} \beta(s, \sigma'(s)) \cdot \mathbf{S}(s, \sigma'(s))$ yields a strictly smaller value for the original game than minimizing $\sum_{s \in S} \beta(s, \sigma_i(s)) \cdot \mathbf{S}(s, \sigma_i(s))$, and
- By Lemmas 1 and 2, the first step along the simplex from V, guided by the objective to minimize $\sum_{s \in S} \beta(s, \sigma'(s)) \cdot \mathbf{S}(s, \sigma'(s))$, leads to a strictly smaller objective value.

This smaller value is achieved at a neighboring corner, and since a locally optimal strategy of this adjacent corner cannot perform worse, we can choose σ_{i+1} as a locally optimal strategy at this adjacent corner. ∎

Correctness. Putting the lemmas of this section together, we get termination guarantees.

Theorem 1 (Correctness and Convergence): If λ is sufficiently close to 1, and the hyperparameters α and β are drawn independently and uniformly at random, then the algorithm almost surely terminates with co-optimal strategies, and none of the linear programs invoked during its execution encounters degeneracy.

Proof: We consider a fixed $\lambda < 1$. For any linear program invoked during the algorithm, degeneracies of both types—(i) corners of the simplex that satisfy more inequalities than the polytope's dimensionality, and (ii) edges along which the objective function does not change—occur with probability zero by Lemmas 1 and 2, respectively. Hence, with probability one, no degeneracy is encountered when solving these linear programs using the Simplex algorithm.

If co-optimal policies have not yet been found, the algorithm proceeds by improving the objective function, typically through greedy local updates. In cases where no such updates are available, Lemma 3 ensures that a better policy exists at a neighboring vertex of the simplex. Since the number of joint policies is finite, this process is guaranteed to terminate with co-optimal strategies. We finally recall that, if the contraction factor $\lambda < 1$ is chosen sufficiently close to 1, then the optimal strategies for the discounted game are also optimal for the original (uncontracted) simple stochastic game. ∎

IV. Experimental Evaluation

We conducted preliminary experiments on variations of the classic Frozen Lake environment from OpenAI's Gym library. In this environment, the agent is placed on a 4×4 grid, where each cell is labeled as start (S), frozen (F), hole (H), or goal (G). The agent can move north, east, west, or south, unless the move would cross the grid boundary, in which case the agent remains in the current state. Cells labeled H and G are absorbing states: any action taken from these states results in no change of state.

In our modified setting, we randomly select between 0 and 10 holes and reinterpret these as states controlled by

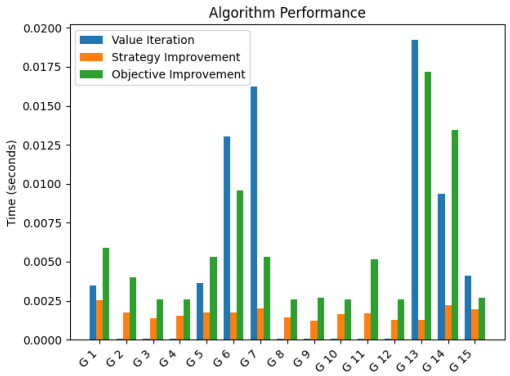

Fig. 1: Comparison of our algorithm, value iteration, and strategy improvement on modified *Frozen Lake* grid-world.

the opponent (player Min). The objective of player Max is to reach the goal state G, while player Min aims to prevent this. Figure 1 reports the performance of three algorithms. These instances are particularly favorable to the strategy improvement algorithm, which converges quickly to the optimal policy. Our algorithm performs comparably to value iteration on these examples, while offering additional advantages that make it a competitive alternative.

We also evaluated the algorithms on a small suite of challenging benchmark examples, shown in Fig. 2. These include: G1, Condon's counterexample to the naive strategy improvement algorithm [4]; G2, a four-state, two-action stochastic game with a low discount factor; G3, the same example with a high discount factor; G4, a smaller three-state reachability instance; G5, the example from Section III; and G6, a simplified room temperature control example inspired by [15]. On this suite, performance is mixed across all three algorithms. However, when the discount factor is high, value iteration generally requires more iterations to converge.

Overall, our experiments suggest that the proposed algorithm consistently outperforms value iteration, particularly in settings with high discount factors where value iteration converges slowly. Its performance relative to strategy improvement is more mixed: in several cases—especially on randomly generated examples—strategy improvement converges faster, while in others our algorithm is competitive. We view this as representative of larger problems, since—much like the Simplex method for linear programming—strategy improvement is known to perform well in practice despite its high theoretical complexity. Further work is needed to better characterize the classes of examples on which our algorithm outperforms strategy improvement.

V. Conclusion

We presented a new algorithm for solving stochastic games with reachability objectives. Inspired by Condon's quadratic programming formulation, our approach replaces the quadratic objective with a sequence of linear objectives, each solved using linear programming.

To address degeneracy issues commonly encountered with the Simplex algorithm, we construct an intermediate discounted optimization game and incorporate randomly chosen

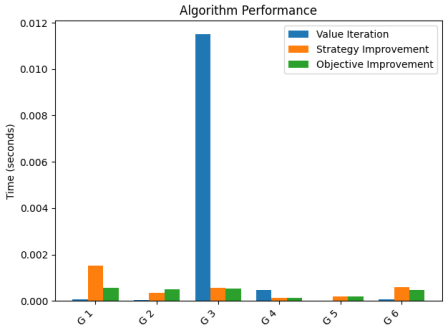

Fig. 2: Comparison of our algorithm, value iteration, and strategy improvement on *smaller specialized examples*.

hyperparameters (α, β). These hyperparameters mitigate two key sources of degeneracy: (i) corners of the simplex that satisfy more inequalities than the polytope's dimensionality, and (ii) edges along which the objective function remains constant. We prove that the algorithm converges almost surely, terminating in finitely many iterations and yielding both the optimal value and optimal strategies for the two players. Since stochastic games lie in $\mathsf{NP} \cap \mathsf{CoNP}$ and no polynomial-time algorithm is currently known, we believe that novel approaches such as the one proposed here provide promising directions toward a polynomial-time solution.

References

[1] S. Shapley, "Stochastic Games." *Proceedings of the national academy of sciences*, vol. 39, no. 10, pp. 1095–1100, 1953.

[2] P. J. Ramadge and W. M. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–98, 1989.

[3] A. Condon, "The complexity of stochastic games." *Information and Computation*, vol. 96, no. 2, pp. 203–224, 1992.

[4] ——, "On Algorithms for Simple Stochastic Games." in *Advances in Computational Complexity Theory*, ser. Discrete Mathematics and Theoretical Computer Science13. American Mathematical Society, 1993, pp. 51–73.

[5] A. J. Hoffman and R. M. Karp, "On nonterminating stochastic games," *Management Science*, vol. 12, no. 5, pp. 359–370, 1966.

[6] C. H. Papadimitriou, *Computational Complexity*. Addison-Wesley, 1994.

[7] E. Kelmendi, J. Krämer, J. Křetínskỳ, and M. Weininger, "Value iteration for simple stochastic games: Stopping criterion and learning algorithm," in *International conference on computer aided verification*. Springer, 2018, pp. 623–642.

[8] C. W. Klingler, "An empirical analysis of algorithms for simple stochastic games," Master's thesis, West Virginia University, 2023.

[9] G. B. Dantzig, "Origins of the simplex method," in *A history of scientific computing*, 1990, pp. 141–151.

[10] ——, "Making progress during a stall in the simplex algorithm," *Linear Algebra and its Applications*, vol. 114, pp. 251–259, 1989.

[11] E. M. Hahn, S. Schewe, A. Turrini, and L. Zhang, "A simple algorithm for solving qualitative probabilistic parity games," in *Computer Aided Verification (CAV)*, ser. Lecture Notes in Computer Science, vol. 9780. Springer, 2016, pp. 291–311.

[12] K. Chatterjee, M. Jurdzinski, and T. A. Henzinger, "Quantitative stochastic parity games," in *SODA*, vol. 4, 2004, pp. 121–130.

[13] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2024. [Online]. Available: https://www.gurobi.com

[14] D. Dell'Erba, A. Dumas, and S. Schewe, "An Objective Improvement Approach to Solving Discounted Payoff Games." in *Games, Automata, Logics, and Formal Verification'23*, 2023, pp. 203–219.

[15] A. Lavaei, M. Perez, M. Kazemi, F. Somenzi, S. Soudjani, A. Trivedi, and M. Zamani, "Compositional reinforcement learning for discrete-time stochastic control systems," *IEEE Open Journal of Control Systems*, vol. 2, pp. 425–438, 2023.