Algorithms for Linear Equations with Min and Max Operators Under (Absolutely) Halting Condition

Krishnendu Chatterjee
IST Austria
Klosterneuburg, Austria
kchatterjee@ist.ac.at

Ruichen Luo
IST Austria
Klosterneuburg, Austria
rluo@ist.ac.at

Raimundo Saona
London School of Economics
London, United Kingdom
raimundo.saona@gmail.com

Jakub Svoboda
Dartmouth College
Hanover, NH, USA
jakub.svoboda@dartmouth.edu

Abstract

We consider linear equations with min and max operators (LEMMs) that contain many subproblems ranging from optimization, learning, to games. Recently, Chatterjee et al. [2025] systematically studied the complexity of different subclasses and proved that three key subclasses (the halting branching process, absolutely halting LEMMs, and halting LEMMs) belong to UP ∩ coUP while generalizing stochastic games (SSGs). In this work, we study the classic algorithms of Policy Iteration (PI) and Value Iteration (VI) for these general subclasses.

First, we simplify the problem hierarchy by showing the equivalence between halting branching process and SSGs. Then, we show that while PI diverges for absolutely halting LEMMs due to the loss of monotonicity, VI remains convergent, a result we establish via diagonal rescaling. Finally, we show that neither PI nor VI converges for general halting LEMMs and, to this end, propose variants of simple policy iteration that ensure convergence across all subclasses.

1 Introduction

Optimization problem. Optimization is core to many problems in machine learning, games, and artificial intelligence [Sra et al., 2011, Rockafellar and Wets, 1998], such as sequential decision making, constraint satisfaction, verifying neural networks, and evolutionary games. In this work, we are interested in the optimization problem of solving for $\mathbf{x} = [x_1, \ldots, x_n]^{\mathrm{T}} \in \mathbb{R}^n$ in the system of *Linear Equations with Min and Max operators* (LEMM) given by $(S_{\min}, S_{\max}, S_{\text{aff}}, n, \mathcal{N}, \mathbf{q}, \mathbf{b})$:

$$\begin{cases} x_i = \min\limits_{l \in \mathcal{N}(i)} x_l, & i \in S_{\min}\,, \\ x_j = \max\limits_{l \in \mathcal{N}(j)} x_l, & j \in S_{\max}\,, \\ x_k = \mathbf{q}_k^{\mathrm{T}} \mathbf{x} + b_k, & k \in S_{\text{aff}}\,, \end{cases} \tag{1}$$

where the following conditions are satisfied: (a) $S_{\min}$, $S_{\max}$, and S_{aff} are disjoint sets such that $S_{\min} \cup S_{\max} \cup S_{\text{aff}} = [n]$,[1] (b) $\varnothing \subsetneq \mathcal{N}(i) \subseteq [n]$ for $i \in S_{\min} \cup S_{\max}$, (c) $\mathbf{q}_k \in \mathbb{R}^n$ for $k \in S_{\text{aff}}$, and (d) $\mathbf{b} = [b_1, \cdots, b_n]^{\mathrm{T}} \in \mathbb{R}^n$ such that $b_i = 0$ for $i \in S_{\min} \cup S_{\max}$.

It is known from the literature Chatterjee et al. [2025] that Problem (1) covers many applications, including linear programs with boolean variables [Senju and Toyoda, 1968], periodic decision-

[1] Denote $[k] = \{1, 2, \cdots, k\}$, for all positive integers k.

Preprint.

making [Markowitz, 1952], verifying neural networks [Katz et al., 2017], constraint satisfaction [Brailsford et al., 1999], and co-evolution in ecosystems [Sigmund and Nowak, 1999].

Baseline: Simple Stochastic Games (SSGs). A fundamental and well-studied subclass of LEMMs is the *Simple Stochastic Game (SSG)* [Condon, 1992]. SSGs naturally arise when the affine operators model probabilistic transitions. Mathematically, an SSG is an LEMM satisfying three restrictive conditions: (1) non-negativity; (2) sum-up-to-one; and (3) halting (the system eventually terminates). A core complexity result of SSGs is that their associated decision problem belongs to the complexity class UP $\cap$ coUP. Because UP $\cap$ coUP is widely believed not to contain NP-hard problems (unless NP = coNP), this complexity upper bound strongly motivates the search for practical iterative algorithms.

Three Generalization Classes and UP $\cap$ coUP Complexity. To explore the algorithmic boundaries of SSGs and their practical applications (cf. [Chatterjee et al., 2025, section 3]), we drop the restrictive "sum-up-to-one" condition to systematically generalize SSGs into three increasingly broad classes. Specifically, we investigate: (Class I) Halting Branching Processes, which maintain non-negativity; (Class II) Absolutely Halting LEMMs, which allow negative weights under strict absolute stability; and (Class III) General Halting LEMMs, which require only asymptotic stability.

Recently, Chatterjee et al. [2025] provided a systematic study of the computational complexities of LEMM subproblems. Remarkably, they proved that despite lacking the strict probabilistic structures of SSGs, the decision problems for all three of these generalized classes retain unique solutions and remain in the UP $\cap$ coUP complexity class.

Iterative Algorithms: Policy and Value Iteration. Because these three classes preserve the core complexity properties of SSGs, they serve as natural candidates for the classic algorithms originally designed to solve SSGs and Markov Decision Processes (MDPs). The two most fundamental algorithms are Policy Iteration (PI) and Value Iteration (VI) Andersson and Miltersen [2009], Condon [1990]: PI iteratively updates the policy by greedily responding to evaluated state values, while VI iteratively updates the value vector by propagating it through the local equations.

Both PI and VI rely heavily on the probabilistic properties of SSGs for their convergence. The central open question of this paper is: *Do the classic iterative algorithms and their convergence guarantees survive when the strict probabilistic structure of SSGs is systematically removed?*

Our Contributions. In this work, we provide a comprehensive algorithmic study on the convergence behaviors of PI, VI, and their variants for the generalized LEMM subproblems in UP $\cap$ coUP. As summarized in Fig. 1, our main contributions are as follows:

(1) **Collapse of Class I:** We show that halting branching processes (Class I) can be linearly reduced to SSGs via a diagonal rescaling transformation. Consequently, Class I is linearly equivalent to SSGs, meaning both PI and VI converge natively with classic guarantees (Theorem 3).

(2) **Absolutely Halting LEMMs (Class II):** When negative weights are introduced, we prove by counterexample that PI diverges. Surprisingly, we prove that VI still converges. To establish this, we introduce a novel diagonal rescaling technique, yielding an explicit convergence rate (Theorem 5).

(3) **General Halting LEMMs (Class III):** We demonstrate that when absolute stability is lost, both PI and VI diverge. To solve this broadest class, we analyze a combinatorial algorithmic family, Simple Policy Iteration (SPI) and its randomized variant (RandSPI), and establish convergence and expected runtime bounds (Theorem 6).

Problem Class	PI	VI	SPI	References
Class I: Halting Branching Process (and SSGs)	✓	✓	✓	Prop. 2, **Thm. 3**
Class II: Absolutely Halting	✘	✔	✔	**Thm. 5**
Class III: General Halting	✘	✘	✔	**Thm. 6**

Figure 1: *Three key generalization classes of SSGs in the complexity class UP $\cap$ coUP: the problem classes and the convergence of* PI, VI *and* SPI *algorithms. The rows are ordered by increasing generality (and thus difficulty) from top to bottom. New algorithmic results are in* **bold and red**.

2 Preliminaries

2.1 Basic Notation

We recall the LEMM defined by a tuple $(S_{\min}, S_{\max}, S_{\text{aff}}, \mathcal{N}, \mathbf{q}, \mathbf{b})$ as in Eq. (1). To analyze the system, it is convenient to express the min and max choices in matrix form. Let $\mathbf{e}_i \in \mathbb{R}^n$ denote the standard basis vector where the i-th element is 1 and all others are 0. We define the sets of all possible min and max policy matrices as:

$$\mathcal{Q}_{\min} = \left\{ [\mathbf{e}_{l_1}, \cdots, \mathbf{e}_{l_{|S_{\min}|}}]^{\mathrm{T}} \mid l_i \in \mathcal{N}(i) \text{ for } i \in S_{\min} \right\},$$
$$\mathcal{Q}_{\max} = \left\{ [\mathbf{e}_{l_{|S_{\min}|+1}}, \cdots, \mathbf{e}_{l_{|S_{\min}|+|S_{\max}|}}]^{\mathrm{T}} \mid l_j \in \mathcal{N}(j) \text{ for } j \in S_{\max} \right\}.$$

Let $\mathbf{Q}_{\text{aff}} = [\mathbf{q}_{|S_{\min}|+|S_{\max}|+1}, \ldots, \mathbf{q}_n]^{\mathrm{T}}$ be the fixed matrix for the affine rows. The set of all possible global transition matrices $\mathcal{Q}$ is the Cartesian product of these choices:

$$\mathcal{Q} = \left\{ \begin{bmatrix} \mathbf{Q}_{\min} \\ \mathbf{Q}_{\max} \\ \mathbf{Q}_{\text{aff}} \end{bmatrix} \middle| \, \mathbf{Q}_{\min} \in \mathcal{Q}_{\min} \text{ and } \mathbf{Q}_{\max} \in \mathcal{Q}_{\max} \right\}.$$

For any matrix $\mathbf{Q}$, we denote its spectral radius by $\varrho(\mathbf{Q})$, and let $|\mathbf{Q}|$ denote the matrix formed by the absolute values of its elements. Furthermore, we denote the convex hull of $\mathcal{Q}$ by $\mathrm{conv}(\mathcal{Q})$.

2.2 Three Generalized Subclasses and UP ∩ coUP Complexity

A fundamental and well-studied subclass of LEMMs is the *Simple Stochastic Game (SSG)* [Condon, 1992]. SSGs naturally arise when the affine operators model probabilistic transitions. Mathematically, an SSG is an LEMM that satisfies three restrictive conditions: (1) **Non-negativity:** $\mathbf{q}_k \geq \mathbf{0}$ and $b_k \geq 0$; (2) **Sum-up-to-one:** $\mathbf{q}_k^{\mathrm{T}}\mathbf{1} + b_k \leq 1$; and (3) **Halting:** The game eventually terminates, meaning $\lim_{k\to\infty} \mathbf{Q}^k = \mathbf{O}$ for all $\mathbf{Q} \in \mathrm{conv}(\mathcal{Q})$.

A central feature of SSGs is that their associated decision problem, determining whether a specific coordinate of the solution is below a given threshold, belongs to the complexity class UP ∩ coUP. This class contains problems that have unique certificates for both "yes" and "no" answers. Because UP ∩ coUP is widely believed not to contain NP-hard problems (unless NP = coNP), this structure strongly motivates the search for practical iterative algorithms.

In this work, to explore the algorithmic boundaries of SSGs, we drop the restrictive "sum-up-to-one" condition (which enforces a strict probabilistic interpretation) and systematically generalize SSGs into three increasingly broad classes:

1. **Class I: Halting Branching Processes.** We drop the sum-up-to-one restriction but maintain non-negative transition weights alongside the general stability of the system. Note that while some literature considers branching processes in a multiplicative sense (e.g., multiplying probabilities for AND procedures Etessami et al. [2012]), we consider them in the linear additive sense here, modeling expected accumulated values which naturally yields linear equations with non-negative coefficients Pliska [1976], Rothblum and Whittle [1982], Etessami et al. [2008]. (*Conditions:* Non-negativity and Halting).
2. **Class II: Absolutely Halting LEMMs.** We drop the non-negativity constraint, allowing negative coefficients. To maintain a well-behaved notion of stability in the presence of negative weights, we require the system to be *absolutely* stable. (*Condition:* $\lim_{k\to\infty} |\mathbf{Q}|^k = \mathbf{O}$ for all $\mathbf{Q} \in \mathcal{Q}$).
3. **Class III: General Halting LEMMs.** This allows arbitrary weights (positive and negative) and only requires general stability. (*Condition:* $\lim_{k\to\infty} \mathbf{Q}^k = \mathbf{O}$ for all $\mathbf{Q} \in \mathrm{conv}(\mathcal{Q})$).[2]

Remarkably, recent work by Chatterjee et al. [2025] proved that despite lacking the strict probabilistic structures of SSGs, the decision problems for all three of these generalized classes retain unique

[2] "For non-negative matrices (Classes I and II), pure-policy stability implies stability over the entire convex hull. However, since this property fails when negative weights are introduced, Class III must explicitly define the halting condition over $\mathrm{conv}(\mathcal{Q})$, following the literature [Chatterjee et al., 2025].

solutions and remain strictly within UP ∩ coUP. Because they preserve this core complexity property, they serve as natural candidates for the classic iterative algorithms originally designed for SSGs.

Let us state the two useful properties of halting LEMMs.

Proposition 1 (Condon 1992, Chatterjee et al. 2025)**.** *For any halting LEMM, the following hold:*

1. *The system has a unique exact solution* $\mathbf{x}^*$.
2. *If the system is non-negative and contains no min operators* ($S_{\min} = \varnothing$)*, its solution can be found by solving a linear program in polynomial time.*

2.3 Policy Iteration and Value Iteration

The classic algorithms for solving SSGs and MDPs are Policy Iteration (PI) and Value Iteration (VI). Both algorithms can be naturally adapted to the general LEMM framework. To do so, we adopt the standard terminology of reinforcement learning: we refer to the state value functions as vectors in $\mathbb{R}^n$, and define the fundamental update rule via the *Bellman operator*. To clearly differentiate the two algorithms, we will use $\mathbf{v}$ for the value vectors in PI, and $\mathbf{u}$ for the iterates in VI. Let $\mathbf{x}^*$ denote the exact unique solution to the system.

The Bellman Operator. The core of both algorithms is the Bellman operator $\mathcal{T}\colon \mathbb{R}^n \to \mathbb{R}^n$, which applies one step of the system's local equations. For any vector $\mathbf{u} \in \mathbb{R}^n$, the updated value $\mathcal{T}(\mathbf{u})$ is defined coordinate-wise:

$$\begin{cases} [\mathcal{T}(\mathbf{u})]_i = \min\limits_{l \in \mathcal{N}(i)} u_l\,, & i \in S_{\min}\,, \\ [\mathcal{T}(\mathbf{u})]_j = \max\limits_{l \in \mathcal{N}(j)} u_l\,, & j \in S_{\max}\,, \\ [\mathcal{T}(\mathbf{u})]_k = \mathbf{q}_k^{\mathsf{T}}\mathbf{u} + b_k\,, & k \in S_{\text{aff}}\,. \end{cases} \tag{2}$$

Policy Iteration (PI). To simplify the presentation, we describe PI for updating max policies (Plmax); the equivalent algorithm updating min policies (Plmin) follows symmetrically. A max policy specifies a choice $l \in \mathcal{N}(j)$ for every $j \in S_{\max}$. As defined in Section 2.1, this policy is represented by a transition matrix $\mathbf{Q}_{\max} \in \mathcal{Q}_{\max}$. Then Plmax alternates between two phases:

- **Policy Evaluation:** For a fixed max policy (represented by $\mathbf{Q}_{\max}$), the system's Bellman operator reduces to a partially-fixed affine operator $\mathcal{T}^{\mathbf{Q}_{\max}}$. The value of this policy, denoted $\mathbf{v} \in \mathbb{R}^n$, is the unique fixed point $\mathbf{v} = \mathcal{T}^{\mathbf{Q}_{\max}}(\mathbf{v})$.
- **Policy Improvement:** Given a value function $\mathbf{v}$, the policy is updated by greedily choosing the neighbor that maximizes the value at each max state. That is, for each $j \in S_{\max}$, we pick $l \in \arg\max_{l' \in \mathcal{N}(j)} v_{l'}$. In matrix form, the new policy is any $\mathbf{Q}'_{\max} \in \arg\max_{\mathbf{Q} \in \mathcal{Q}_{\max}} \mathbf{Q}\mathbf{v}$.

The Plmax procedure is detailed in Algorithm 1.

Algorithm 1 Policy Iteration updating max policies Plmax $(\mathbf{Q}_{\max}^{(0)})$

Require: Initial max policy $\mathbf{Q}_{\max}^{(0)} \in \mathcal{Q}_{\max}$

for $t = 1, 2, \ldots$ **do**
  $\mathbf{v}^{(t-1)} \leftarrow$ fixed point of $\mathcal{T}^{\mathbf{Q}_{\max}^{(t-1)}}$ ▷ Policy Evaluation
  $\mathbf{Q}_{\max}^{(t)} \leftarrow \arg\max_{\mathbf{Q} \in \mathcal{Q}_{\max}} \mathbf{Q}\mathbf{v}^{(t-1)}$ ▷ Policy Improvement (Greedy)
  if $\mathbf{Q}_{\max}^{(t)} = \mathbf{Q}_{\max}^{(t-1)}$ **then**
    return $\mathbf{v}^{(t-1)}$ ▷ Terminate when policy stabilizes
  end if
end for

Value Iteration (VI). The VI algorithm bypasses exact policy evaluation by iteratively applying the full Bellman operator. Starting from an initial value vector $\mathbf{u}^{(0)}$, VI simply computes $\mathbf{u}^{(t)} = \mathcal{T}(\mathbf{u}^{(t-1)})$, as detailed in Algorithm 2.

Algorithm 2 Value Iteration VI $(\mathbf{u}^{(0)})$

Require: Initial value vector $\mathbf{u}^{(0)} \in \mathbb{R}^n$
for $t = 1, 2, \ldots$ **do**
  $\mathbf{u}^{(t)} \leftarrow \mathcal{T}(\mathbf{u}^{(t-1)})$ ▷ Bellman Update
end for

Classic Guarantees for SSGs. For the baseline case of SSGs, both algorithms are guaranteed to converge. The rate of VI convergence is classically determined by the smallest positive transition probability, denoted $p_{\min} = \min\{(\mathbf{q}_k)_j \mid (\mathbf{q}_k)_j > 0,\ k \in S_{\text{aff}},\ j \in [n]\}$, and the minimum positive leak to the halting sink, denoted $\delta_{\min} = \min\{1 - \mathbf{q}_k^\top \mathbf{1} \mid 1 - \mathbf{q}_k^\top \mathbf{1} > 0,\ k \in S_{\text{aff}}\}$. By the halting condition, the leak $\delta_{\min}$ is strictly positive.

Proposition 2 (Classic Analyses for SSGs [Condon, 1990, Chatterjee et al., 2023])**.** *Let an SSG system have the unique solution* $\mathbf{x}^*$.

1. ***PI Convergence:*** *From any initial policy* $\mathbf{Q}^{(0)}_{\max}$, PImax *reaches the exact solution* $\mathbf{x}^*$ *in at most* $\prod_{j \in S_{\max}} |\mathcal{N}(j)|$ *iterations.*
2. ***VI Convergence:*** *For any initial vector* $\mathbf{u}^{(0)}$ *satisfying* $\mathbf{0} \le \mathbf{u}^{(0)} \le \mathbf{x}^*$*, the iterates of* VI *satisfy the following* n*-step linear convergence:*

$$\|\mathbf{x}^* - \mathbf{u}^{(kn)}\|_\infty \le (1 - p_{\min}^{n-1}\delta_{\min})^k \|\mathbf{x}^* - \mathbf{u}^{(0)}\|_\infty\,, \quad \textit{for all } k \ge 1\,.$$

The PI guarantee in Proposition 2 relies on the monotonicity of the sequence of value estimates, which ensures that the algorithm strictly improves the policy at each step and terminates without cycling. However, because the algorithm might traverse a large fraction of the policy space in the worst case, the total iteration bound $\prod_{j \in S_{\max}} |\mathcal{N}(j)|$ is exponential in the number of nodes.

The VI guarantee relies on a bounding initialization (e.g., a sub-solution $\mathbf{0} \le \mathbf{u}^{(0)} \le \mathbf{x}^*$), which ensures that the sequence $\mathbf{u}^{(t)}$ converges monotonically to $\mathbf{x}^*$ from below. Furthermore, because SSG transitions sum to at most one and halt, from any state there is a valid path of length at most $n-1$ to the halting sink. The probability of taking this path is at least $p_{\min}^{n-1}$. Consequently, after every n iterations of the Bellman operator, the maximum error contracts strictly by a factor of $(1 - p_{\min}^{n-1}\delta_{\min})$, yielding the exponential $p_{\min}^{n-1}\delta_{\min}$ dependence in the worst-case runtime bound.

While both algorithms perform excellently in practice and their theoretical guarantees are well understood for SSGs, their worst-case runtimes inherently face exponential barriers. More importantly, it is entirely unclear whether these iterative algorithms and their classic guarantees survive when the strict probabilistic structure of SSGs is removed. In the following sections, we systematically test PI and VI against our generalized classes.

3 Equivalence of Class I and SSGs

We first study **Class I**, which consists of halting LEMMs with non-negative transition weights but no restrictions on their row sums. In the recent systematic study of LEMM complexities by Chatterjee et al. [2025], this class was identified as an intermediate problem class, positioned between general halting LEMMs and SSGs. Because its rows are not bounded by one, it drops the strict probabilistic interpretation of SSGs, and its exact algorithmic relationship to SSGs was left as an open question.

In this section, we significantly simplify the problem hierarchy by showing that Class I actually collapses completely into SSGs.

The only difference between Class I and SSGs is the "sum-up-to-one" condition. To bridge this gap, we rely on the basic properties of non-negative halting systems in Proposition 1. Consequently, for the affine vector $\mathbf{b}$, there exists a strictly positive bounding vector $\mathbf{v} \in \mathbb{R}^n_{\ge 1}$ such that

$$\mathbf{v} \ge \mathbf{Q}\mathbf{v} + \mathbf{b} \quad \text{for all } \mathbf{Q} \in \mathcal{Q}\,, \tag{3}$$

and moreover, such a $\mathbf{v}$ can be found in polynomial time by replacing all the $\min$ variables in the LEMM by $\max$ and then solving a linear program (cf. Proposition 1).

By using this vector $\mathbf{v}$ to construct a diagonal rescaling matrix $\Lambda = [\mathrm{diag}(\mathbf{v})]^{-1}$, we can uniformly scale the rows of the system. This purely algebraic transformation strictly bounds the row sums to at most one without altering the fundamental problem. Because Class I already assumes the halting and non-negativity conditions, this transformation directly yields an SSG.

Theorem 3 (Collapse of Class I)**.** *Every LEMM in Class I can be linearly reduced to an SSG of linear size. Consequently, Class I and SSGs are linearly equivalent.*

The detailed proof constructing this rescaling matrix is deferred to Section B.

Algorithmic Implications. Theorem 3 establishes that Class I is computationally equivalent to SSGs up to a linear increase in problem size. Consequently, algorithms designed to solve SSGs can be applied to Class I. This implies that **both PI and VI converge for Class I**, with runtime bounds analogous to those presented in Proposition 2, scaled by the expanded state space. Furthermore, subexponential-time algorithms developed for SSGs, such as the randomized algorithm by Ludwig [1995], are also applicable to Class I.

With Class I fully resolved by its equivalence to our SSG baseline, we dedicate the remainder of this paper to the true algorithmic frontiers: Class II (where negative weights are introduced) and Class III (where absolute stability is lost).

4 Algorithms for Absolutely Halting LEMMs

In this section, we study algorithms for **Class II** (Absolutely Halting LEMMs). In this class, we allow arbitrary negative coefficients while maintaining the strict requirement of absolute stability ($\lim_{k\to\infty} |\mathbf{Q}|^k = \mathbf{O}$ for all $\mathbf{Q} \in \mathcal{Q}$). Because negative weights destroy the inherent monotonicity of the Bellman operator $\mathcal{T}$, the behavior of classic iterative algorithms drastically changes. We first show that Policy Iteration (PI) fails completely, and then we introduce a novel rescaling technique to prove that Value Iteration (VI) surprisingly still converges.

4.1 Divergence of Policy Iteration

The classic convergence guarantee of PI strictly relies on the fact that policy updates monotonically improve the value vector. When negative coefficients are introduced, this monotonicity is broken, causing PI to easily enter an infinite loop, alternating between suboptimal policies.

Example 1 (Divergence of PImax)**.** Consider the following Class II absolutely halting LEMM:

$$\begin{cases} x_1 = \max\{x_4, x_5\}\,, & x_4 = -0.2x_2 + 0.2x_3 + 0.25\,, & x_7 = 0.3x_1 + 0.1x_2 - 0.3x_3 + 0.25\,, \\ x_2 = \max\{x_6, x_7\}\,, & x_5 = 0.3x_1 - 0.6x_3 + 0.25\,, & x_8 = -0.2x_1 + 0.4x_2 + 0.3\,, \\ x_3 = \max\{x_8, x_9\}\,, & x_6 = -0.5x_1 + 0.25\,, & x_9 = -0.7x_2 + 0.3x_3 + 0.3\,. \end{cases}$$

As verified in Section A, this system satisfies the absolute halting condition. Its unique exact solution $\mathbf{v}^*$ corresponds to the fixed-point policy selecting indices $\{4, 7, 8\}$, yielding the max state values $[v_1^*, v_2^*, v_3^*]^{\mathrm{T}} \approx [0.269, 0.252, 0.347]^{\mathrm{T}}$.

However, let us run PImax from an initial policy selecting $\{4, 6, 8\}$, yielding $[v_1^{(0)}, v_2^{(0)}, v_3^{(0)}]^{\mathrm{T}} \approx [0.286, 0.107, 0.286]^{\mathrm{T}}$. By extracting greedy updates iteratively, the algorithm completely misses the exact solution and loops:

$$\begin{aligned} &\textbf{Policy 1: } \mathbf{Q}_{\max}^{(1)} = [\mathbf{e}_4, \mathbf{e}_7, \mathbf{e}_9]^{\mathrm{T}} \implies \mathbf{v}^{(1)} \approx [0.212, 0.309, 0.120]^{\mathrm{T}} \\ &\textbf{Policy 2: } \mathbf{Q}_{\max}^{(2)} = [\mathbf{e}_5, \mathbf{e}_7, \mathbf{e}_8]^{\mathrm{T}} \implies \mathbf{v}^{(2)} \approx [0.049, 0.174, 0.360]^{\mathrm{T}} \\ &\textbf{Policy 3: } \mathbf{Q}_{\max}^{(3)} = [\mathbf{e}_4, \mathbf{e}_6, \mathbf{e}_8]^{\mathrm{T}} \implies \mathbf{v}^{(3)} \approx [0.286, 0.107, 0.286]^{\mathrm{T}} \end{aligned}$$

Because $\mathbf{Q}_{\max}^{(3)} = \mathbf{Q}_{\max}^{(0)}$, the algorithm enters an infinite loop.

4.2 Convergence Analysis of Value Iteration

Unlike PI, VI does not rely on policy evaluation. However, proving its convergence for Class II is highly non-trivial. For SSGs, the classic VI analysis relies on the property that the transition rows sum to at most one, yielding a strict contraction in the standard ℓ_∞-norm after n steps. For absolutely halting LEMMs, this logic seemingly breaks down.

Example 2 (Failure of Unscaled Contraction)**.** Consider a trivial $n = 2$ system with transition matrix:

$$\mathbf{Q} = \begin{bmatrix} 0.8 & 0.8 \\ 0 & 0.8 \end{bmatrix} .$$

The spectral radius is strictly bounded by one ($\varrho(\mathbf{Q}) = 0.8 < 1$), making the system absolutely halting. However, the absolute row sum is $1.6 > 1$. After $n = 2$ steps, the transition matrix becomes:

$$\mathbf{Q}^2 = \begin{bmatrix} 0.64 & 1.28 \\ 0 & 0.64 \end{bmatrix} .$$

The maximum absolute row sum is $\|\mathbf{Q}^2\|_\infty = 1.92 > 1$. Thus, VI does not contract under the standard ℓ_∞-norm within n steps.

To restore the contraction guarantee, we draw inspiration from the diagonal rescaling technique used in the equivalence reduction (Theorem 3).

Diagonal rescaling. To guarantee convergence of VI, a sufficient condition is that the maximum absolute row sum of the transition matrices is strictly less than one. If $\||\mathbf{Q}|\mathbf{1}\|_\infty \le \gamma < 1$ for all $\mathbf{Q} \in \mathcal{Q}$, the Bellman operator $\mathcal{T}$ acts as a strict γ-contraction in the standard ℓ_∞-norm: $\|\mathcal{T}(\mathbf{u}) - \mathcal{T}(\mathbf{u}')\|_\infty \le \gamma\|\mathbf{u} - \mathbf{u}'\|_\infty$.

For general Class II LEMMs, we can achieve a comparable condition by applying a diagonal rescaling. Because the system is absolutely halting ($\max_{\mathbf{Q}\in\mathcal{Q}} \varrho(|\mathbf{Q}|) < 1$), there exists a strictly positive bounding vector $\mathbf{v} \ge \mathbf{1}$ such that $|\mathbf{Q}|\mathbf{v} \le \mathbf{v}$ for all $\mathbf{Q} \in \mathcal{Q}$ (cf. the second property of Proposition 1).

We define the diagonal rescaling matrix $\Lambda = [\mathrm{diag}(\mathbf{v})]^{-1}$ and establish the rescaled absolute transition matrices as $\tilde{\mathbf{Q}} = \Lambda|\mathbf{Q}|\Lambda^{-1}$. By construction, these rescaled matrices are non-negative, and their row sums are bounded by one:

$$\tilde{\mathbf{Q}}\mathbf{1} = \Lambda|\mathbf{Q}|\mathbf{v} \le \Lambda\mathbf{v} = \mathbf{1} .$$

We define $\tilde{p}_{\min}$ as the minimum positive entry, and $\tilde{\delta}_{\min}$ as the minimum positive leak in this rescaled system:

$$\tilde{p}_{\min} \triangleq \min\left\{\tilde{Q}_{ij} \mid \tilde{Q}_{ij} > 0,\ \tilde{\mathbf{Q}} = \Lambda|\mathbf{Q}|\Lambda^{-1},\ \mathbf{Q} \in \mathcal{Q}\right\},$$
$$\tilde{\delta}_{\min} \triangleq \min\left\{1 - [\tilde{\mathbf{Q}}\mathbf{1}]_i \mid [\tilde{\mathbf{Q}}\mathbf{1}]_i < 1,\ \tilde{\mathbf{Q}} = \Lambda|\mathbf{Q}|\Lambda^{-1},\ \mathbf{Q} \in \mathcal{Q},\ i \in [n]\right\}.$$

Because the rescaled system is an absolutely halting process where all row sums are bounded by one, path-leakage analysis establishes that the product of any sequence of n transition matrices is a strict contraction. As proven in Proposition 7 (detailed in the appendix), we have:

$$\|\tilde{\mathbf{Q}}^{(n)} \cdots \tilde{\mathbf{Q}}^{(1)}\mathbf{1}\|_\infty \le 1 - \tilde{p}_{\min}^{n-1}\tilde{\delta}_{\min} < 1 .$$

Rescaled contraction of VI. While the original system may temporarily diverge in the standard ℓ_∞-norm (as demonstrated in Example 2), the linear scaling property of the absolute policy matrices ensures that the Bellman operator strictly contracts every n steps under the Λ-rescaled norm $\|\Lambda\mathbf{x}\|_\infty$.

Lemma 4 (n-step contraction of $\mathcal{T}$)**.** *Consider an absolutely halting LEMM rescaled by Λ. For all vectors $\mathbf{u}, \mathbf{u}' \in \mathbb{R}^n$, the n-step Bellman operator $\mathcal{T}^n$ satisfies:*

$$\|\Lambda\big(\mathcal{T}^n(\mathbf{u}) - \mathcal{T}^n(\mathbf{u}')\big)\|_\infty \le (1 - \tilde{p}_{\min}^{n-1}\tilde{\delta}_{\min})\|\Lambda(\mathbf{u} - \mathbf{u}')\|_\infty .$$

We defer the rigorous proof of Lemma 4 to Section C. With this rescaled contraction established, we conclude with providing the asymptotic convergence guarantee of VI for Class II LEMMs.

Theorem 5 (Algorithmic Guarantees for Class II)**.** *Consider an absolutely halting LEMM (Class II) with exact unique solution $\mathbf{x}^*$.*

1. ***PI Divergence:*** PI *fails to converge in general.*
2. ***VI Convergence:*** *For any initial vector $\mathbf{u}^{(0)} \in \mathbb{R}^n$, the iterates $\mathbf{u}^{(t)}$ of* VI $(\mathbf{u}^{(0)})$ *converge linearly to $\mathbf{x}^*$. Specifically, under the Λ-rescaled norm:*

$$\|\Lambda(\mathbf{u}^{(kn)} - \mathbf{x}^*)\|_\infty \le (1 - \tilde{p}_{\min}^{n-1}\tilde{\delta}_{\min})^k\|\Lambda(\mathbf{u}^{(0)} - \mathbf{x}^*)\|_\infty , \quad \textit{for all } k \ge 1 .$$

5 Algorithms for General Halting LEMMs

In this section, we study algorithms for **Class III** (General Halting LEMMs). In this broadest class, we drop both the non-negativity constraint and the absolute stability requirement, assuming only standard asymptotic stability (i.e., $\lim_{k\to\infty} \mathbf{Q}^k = \mathbf{O}$ for all $\mathbf{Q} \in \mathrm{conv}(\mathcal{Q})$).

5.1 Divergence of Value Iteration

We have already established that Policy Iteration (PI) diverges in the presence of negative weights (Example 1), a limitation that inherently carries over to Class III. For Value Iteration (VI), the loss of absolute stability poses a fatal issue. Because the absolute matrices $|\mathbf{Q}|$ can now have spectral radii greater than one, we can no longer find a strictly positive vector to bound the absolute row sums. Consequently, VI loses its rescaled contraction property and can easily enter an infinite loop.

Example 3 (Divergence of VI)**.** Consider the following Class III LEMM:

$$\begin{cases} x_1 = \max\{x_3, x_4\}\,, & x_4 = 0.5x_1 + 1.5x_2 - 1.5\,, \\ x_2 = \max\{x_5, x_6\}\,, & x_5 = -0.5x_1 - 1.0\,, \\ x_3 = -0.9x_1 + 1.8x_2 - 1.5\,, & x_6 = -0.25x_1 - 0.25x_2 - 1.0\,. \end{cases}$$

This system satisfies the general halting condition; as verified in Section A, the spectral radius is strictly less than one for all matrices in the convex hull $\mathrm{conv}(\mathcal{Q})$.

However, if we run the VI algorithm starting from $\mathbf{u}^{(0)} = \left[\frac{3}{7}, \frac{5}{7}, -\frac{3}{5}, -\frac{3}{14}, -\frac{17}{14}, -\frac{9}{7}\right]^{\mathrm{T}}$, the iterates follow the cyclic sequence:

$$\begin{aligned} \mathbf{u}^{(1)} &= [-\tfrac{3}{14}, -\tfrac{17}{14}, -\tfrac{3}{5}, -\tfrac{3}{14}, -\tfrac{17}{14}, -\tfrac{9}{7}]^{\mathrm{T}}\,, \\ \mathbf{u}^{(2)} &= [-\tfrac{3}{14}, -\tfrac{17}{14}, -\tfrac{489}{140}, -\tfrac{24}{7}, -\tfrac{21}{28}, -\tfrac{9}{14}]^{\mathrm{T}}\,, \\ \mathbf{u}^{(3)} &= [-\tfrac{24}{7}, -\tfrac{9}{14}, -\tfrac{489}{140}, -\tfrac{24}{7}, -\tfrac{21}{28}, -\tfrac{9}{14}]^{\mathrm{T}}\,, \\ \mathbf{u}^{(4)} &= [-\tfrac{24}{7}, -\tfrac{9}{14}, \tfrac{3}{7}, -\tfrac{117}{28}, \tfrac{5}{7}, \tfrac{1}{56}]^{\mathrm{T}}\,, \\ \mathbf{u}^{(5)} &= [\tfrac{3}{7}, \tfrac{5}{7}, \tfrac{3}{7}, -\tfrac{117}{28}, \tfrac{5}{7}, \tfrac{1}{56}]^{\mathrm{T}}\,, \\ \mathbf{u}^{(6)} &= [\tfrac{3}{7}, \tfrac{5}{7}, -\tfrac{3}{5}, -\tfrac{3}{14}, -\tfrac{17}{14}, -\tfrac{9}{7}]^{\mathrm{T}}\,. \end{aligned}$$

Because $\mathbf{u}^{(6)} = \mathbf{u}^{(0)}$, the algorithm enters an infinite loop and fails to converge to the exact solution.

5.2 Simple Policy Iteration and Randomized SPI

Since both PI and VI can diverge for Class III, let us now investigate a different family of combinatorial algorithms: Simple Policy Iteration (SPI) and its randomized variant (RandSPI). Unlike standard PI, which greedily updates the policy at *all* states simultaneously, SPI updates the choice at only *one* locally violated state at a time, strictly checking violations according to a fixed topological permutation of states (see Algorithm 3). RandSPI operates recursively (see Algorithm 4): it fixes a random neighbor for the highest-index non-linear state, recursively solves the smaller induced game, and updates the choice only if the fixed choice turns out to be suboptimal.

Algorithm 3 Simple Policy Iteration SPI ($\mathcal{I}$)

Require: $\mathcal{I}(i)$ is a fixed evaluation permutation of $\mathcal{N}(i)$ for each non-linear state $i \in S_{\min} \cup S_{\max}$.

1: Initialize neighbor selection indices $\boldsymbol{\ell} = (1, \ldots, 1)$. Let $\mathbf{Q}$ be formed by choices $\mathcal{I}(i)(\ell_i)$ and fixed affine rows.
2: **loop**
3: $\quad \mathbf{u} \leftarrow (\mathbf{I} - \mathbf{Q})^{-1}\mathbf{b}$ ▷ Evaluate the uniquely induced linear system
4: $\quad \Gamma \leftarrow \{i \in S_{\min} \mid \exists j \in \mathcal{N}(i) \text{ s.t. } u_j < u_i\} \cup \{k \in S_{\max} \mid \exists j \in \mathcal{N}(k) \text{ s.t. } u_j > u_k\}$
5: $\quad$ **if** Γ is empty **then return** $\mathbf{u}$
6: $\quad$ **end if** ▷ No local violations found; exact solution reached
7: $\quad k \leftarrow \min(\Gamma)$; $\ell_k \leftarrow (\ell_k \bmod |\mathcal{N}(k)|) + 1$ ▷ Advance lowest-index violated state
8: $\quad$ Update the k-th row of $\mathbf{Q}$ to match the new choice $\mathcal{I}(k)(\ell_k)$.
9: **end loop**

Algorithm 4 Randomized Simple Policy Iteration RandSPI

if $S_{\min} \cup S_{\max}$ is empty **then return** $(\mathbf{I} - \mathbf{Q}_{\text{aff}})^{-1}\mathbf{b}$
end if
$k \leftarrow \max(S_{\min} \cup S_{\max})$. Draw a uniform random permutation $\mathcal{I}(k)$ of the neighbors $\mathcal{N}(k)$.
for $m = 1, \ldots, |\mathcal{N}(k)|$ **do**
    Fix the choice for state k to $\mathcal{I}(k)(m)$, temporarily converting it into an affine state.
    $\mathbf{u} \leftarrow$ RandSPI(Reduced System)
    if the choice $\mathcal{I}(k)(m)$ is optimal with respect to $\mathbf{u}$ **then return** $\mathbf{u}$
    end if
end for

Rather than applying simultaneous greedy updates, these algorithms rely on single-state combinatorial updates to successfully avoid the cyclic traps introduced by negative weights. Furthermore, every policy-induced subsystem of a halting LEMM inherently remains halting, which guarantees a unique evaluation solution at each step (cf. Proposition 1). Let us conclude with the algorithmic guarantees for this general class.

Theorem 6 (Algorithmic Guarantees for Class III)**.** *Consider a general halting LEMM (Class III) with exact unique solution* $\mathbf{x}^*$.

1. ***PI and VI Divergence:*** *Neither* PI *nor* VI *is guaranteed to converge in general.*
2. ***SPI Convergence:*** *For any fixed neighbor ordering* $\mathcal{I}$*,* SPI *returns the exact solution* $\mathbf{x}^*$ *in no more than* $\prod_{i \in S_{\min} \cup S_{\max}} |\mathcal{N}(i)|$ *iterations.*
3. ***RandSPI Convergence:*** RandSPI *returns the exact solution* $\mathbf{x}^*$ *in an expected number of recursive calls bounded by:*

$$\frac{\prod_{i \in S_{\min} \cup S_{\max}} (|\mathcal{N}(i)| + 1)}{2^{|S_{\min}| + |S_{\max}|}}.$$

The detailed proof of Theorem 6 is deferred to Section D. While the worst-case iteration bound of deterministic SPI matches the total number of policies, its primary theoretical significance is guaranteeing provable convergence for Class III, where both PI and VI may enter cyclic traps. Although it avoids blind enumeration by reacting strictly to local violations, its worst-case guarantee remains exponential. Furthermore, we clarify that the theoretical bound for RandSPI measures the expected number of recursive calls rather than total arithmetic complexity. Nevertheless, because this expected number of calls is exponentially smaller than the naive upper bound of $\frac{\prod_{i \in S_{\min} \cup S_{\max}} |\mathcal{N}(i)|}{2}$, RandSPI still provides a rigorous structural improvement over randomized brute-force enumeration.

6 Discussion

In this work, we systematically charted the algorithmic landscape of LEMM generalizations within the UP $\cap$ coUP complexity class. We established that while halting branching processes are linearly equivalent to simple stochastic games, introducing negative weights in absolutely halting LEMMs causes Policy Iteration to diverge, leaving Value Iteration to converge under a rescaled norm. Furthermore, when absolute stability is lost in general halting LEMMs, both classic iterative algorithms fail, necessitating the combinatorial approach of Simple Policy Iteration. Ultimately, these distinct algorithmic behaviors reveal a clear separation of these generalized problem classes.

A natural question arises: can advanced theoretical breakthroughs for SSGs, such as Ludwig's randomized subexponential-time algorithm [Ludwig, 1995], be extended to these broader classes? While Ludwig's algorithm directly solves Class I due to its linear equivalence to SSGs, it inherently breaks down for Class II and Class III. The central property that Ludwig's algorithm–and indeed standard PI–relies on is the *monotonicity* of the Bellman operator. However, when negative weights are introduced in Class II and Class III, this monotonicity is fundamentally shattered. As demonstrated by our PI counterexamples, a locally "greedy" switch can lead to an incomparable or strictly worse global fixed-point value. Because this essential property that guarantees monotonic progress is lost, Ludwig's subexponential search breaks, leaving the existence of subexponential-time algorithms for Class II and Class III as a compelling open problem for future research.

Acknowledgments and Disclosure of Funding

This work is partially supported by the French Agence Nationale de la Recherche (ANR) ANR-21-CE40-0020, the Austrian Science Fund (FWF) 10.55776/COE12, and the ERC CoG 863818 (ForM-SMArt) grant. Gemini 3.1 Pro is used to help with the writing and analysis.

References

Daniel Andersson and Peter Bro Miltersen. The complexity of solving stochastic games on graphs. In *International Symposium on Algorithms and Computation*, pages 112–121. Springer, 2009. 1

Sally C Brailsford, Chris N Potts, and Barbara M Smith. Constraint satisfaction problems: Algorithms and applications. *European journal of operational research*, 119(3):557–581, 1999. 1

Krishnendu Chatterjee, Tobias Meggendorfer, Raimundo Saona, and Jakub Svoboda. Faster algorithm for turn-based stochastic games with bounded treewidth. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 4590–4605, 2023. doi: 10.1137/1.9781611977554.ch173. 2

Krishnendu Chatterjee, Ruichen Luo, Raimundo Saona, and Jakub Svoboda. Linear equations with min and max operators: Computational complexity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 11, pages 11150–11157, 2025. (document), 1, 1, 2.2, 2, 1, 3

Anne Condon. On algorithms for simple stochastic games. *Advances in computational complexity theory*, 13: 51–72, 1990. doi: 10.1090/dimacs/013/04. 1, 2

Anne Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, 1992. doi: 10.1016/0890-5401(92)90048-K. 1, 2.2, 1

Kousha Etessami, Dominik Wojtczak, and Mihalis Yannakakis. Recursive stochastic games with positive rewards. In *International Colloquium on Automata, Languages, and Programming*, pages 711–723. Springer, 2008. 1

Kousha Etessami, Alistair Stewart, and Mihalis Yannakakis. Polynomial time algorithms for multi-type branching processes and stochastic context-free grammars. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 579–588, 2012. 1

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Proceedings of the 29th International Conference on Computer Aided Verification (CAV)*, pages 97–117, 2017. 1

Walter Ludwig. A subexponential randomized algorithm for the simple stochastic game problem. *Information and computation*, 117(1):151–155, 1995. doi: 10.1006/inco.1995.1035. 3, 6

Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952. URL http://www.jstor.org/stable/2975974. 1

Stanley R Pliska. Optimization of multitype branching processes. *Management Science*, 23(2):117–124, 1976. 1

R Tyrrell Rockafellar and Roger JB Wets. *Variational analysis*. Springer, 1998. 1

Uriel G Rothblum and Peter Whittle. Growth optimality for branching Markov decision chains. *Mathematics of Operations Research*, 7(4):582–601, 1982. 1

Shizuo Senju and Yoshiaki Toyoda. An approach to linear programming with 0-1 variables. *Management Science*, pages B196–B207, 1968. 1

Karl Sigmund and Martin A Nowak. Evolutionary game theory. *Current Biology*, 9(14):R503–R505, 1999. 1

Suvrit Sra, Sebastian Nowozin, and Stephen J Wright. *Optimization for machine learning*. MIT press, 2011. 1

A Certificates for Counterexamples

To support the theoretical divergence claims, we provide mathematical certificates verifying the halting conditions for the counterexamples presented in the main text.

A.1 Halting Certificate for Example 1

Example 1 is a Class II system with three $\max$ states (x_1, x_2, x_3) and two choices per state, yielding $2^3 = 8$ possible deterministic policy matrices. To prove the system is absolutely halting, we verify that the spectral radius of the absolute value matrix $|\mathbf{Q}|$ is less than 1 for all 8 policies ($\max_{\mathbf{Q}\in\mathcal{Q}} \rho(|\mathbf{Q}|) < 1$).

The absolute row vectors corresponding to the available choices are:

$$|r_4| = [0, 0.2, 0.2] \qquad |r_6| = [0.5, 0, 0] \qquad |r_8| = [0.2, 0.4, 0]$$
$$|r_5| = [0.3, 0, 0.6] \qquad |r_7| = [0.3, 0.1, 0.3] \qquad |r_9| = [0, 0.7, 0.3]$$

Constructing the 3×3 absolute transition matrix $|\mathbf{Q}|$ for all 8 combinations of $\{r_4, r_5\} \times \{r_6, r_7\} \times \{r_8, r_9\}$ and computing their spectral radii yields:

$$\rho(|\mathbf{Q}|_{\{4,6,8\}}) \approx 0.474 \qquad \rho(|\mathbf{Q}|_{\{5,6,8\}}) \approx 0.708$$
$$\rho(|\mathbf{Q}|_{\{4,6,9\}}) \approx 0.587 \qquad \rho(|\mathbf{Q}|_{\{5,6,9\}}) \approx 0.809$$
$$\rho(|\mathbf{Q}|_{\{4,7,8\}}) \approx 0.577 \qquad \rho(|\mathbf{Q}|_{\{5,7,8\}}) \approx 0.732$$
$$\rho(|\mathbf{Q}|_{\{4,7,9\}}) \approx 0.758 \qquad \rho(|\mathbf{Q}|_{\{5,7,9\}}) \approx 0.865$$

Because $\rho(|\mathbf{Q}|) \leq 0.865 < 1$ across the entire policy space, the system satisfies the absolute halting condition.

A.2 Halting Certificate for Example 3

Example 3 is a Class III system with two $\max$ states (x_1, x_2). The corresponding transition rows are $r_3 = [-0.9, 1.8]$ and $r_4 = [0.5, 1.5]$ for x_1, and $r_5 = [-0.5, 0]$ and $r_6 = [-0.25, -0.25]$ for x_2.

Any matrix in the convex hull $\mathrm{conv}(\mathcal{Q})$ can be parameterized by choice probabilities $p, q \in [0, 1]$:

$$\mathbf{Q}(p, q) = \begin{bmatrix} p \cdot r_3 + (1-p) \cdot r_4 \\ q \cdot r_5 + (1-q) \cdot r_6 \end{bmatrix} = \begin{bmatrix} 0.5 - 1.4p & 1.5 + 0.3p \\ -0.25 - 0.25q & -0.25 + 0.25q \end{bmatrix}.$$

To prove that $\rho(\mathbf{Q}) < 1$ for all $\mathbf{Q} \in \mathrm{conv}(\mathcal{Q})$, we proceed in two steps. First, we show that at least one policy in the convex hull is halting. Setting $p = 1, q = 1$ yields $\mathbf{Q}(1, 1) = \left[\begin{smallmatrix} -0.9 & 1.8 \\ -0.5 & 0 \end{smallmatrix}\right]$. Its eigenvalues are $-0.45 \pm 0.835i$, giving a spectral radius of $\sqrt{0.9} \approx 0.949 < 1$. Thus, $\mathbf{Q}(1, 1)$ is halting.

Second, because the convex hull is a connected set, the eigenvalues of $\mathbf{Q}(p, q)$ are continuous over (p, q). For any matrix in the convex hull to have a spectral radius ≥ 1, an eigenvalue must cross the unit circle at $\lambda = 1$, $\lambda = -1$, or as a complex conjugate pair with magnitude 1. This requires $\det(\mathbf{I} - \mathbf{Q}) \leq 0$, $\det(\mathbf{I} + \mathbf{Q}) \leq 0$, or $\det(\mathbf{Q}) \geq 1$. We compute these three determinants for all $p, q \in [0, 1]$:

1. **Bounding** $\det(\mathbf{I} - \mathbf{Q}) > 0$**:**

$$\det \begin{bmatrix} 0.5 + 1.4p & -1.5 - 0.3p \\ 0.25 + 0.25q & 1.25 - 0.25q \end{bmatrix} = 1 + 1.825p + 0.25q - 0.275pq > 0\,.$$

2. **Bounding** $\det(\mathbf{I} + \mathbf{Q}) > 0$**:**

$$\det \begin{bmatrix} 1.5 - 1.4p & 1.5 + 0.3p \\ -0.25 - 0.25q & 0.75 + 0.25q \end{bmatrix} = 1.5 - 0.975p + 0.75q - 0.275pq > 0\,.$$

3. **Bounding** $\det(\mathbf{Q}) < 1$**:**

$$\det \begin{bmatrix} 0.5 - 1.4p & 1.5 + 0.3p \\ -0.25 - 0.25q & -0.25 + 0.25q \end{bmatrix} = 0.25 + 0.425p + 0.5q - 0.275pq \leq 0.9 < 1\,.$$

Because these three conditions hold, the eigenvalues are trapped inside the unit circle for all $(p, q) \in [0, 1]^2$. This certifies that every matrix in the convex hull is halting.

B Missing Proofs for Equivalence

Proof of Theorem 3. Consider a Class I LEMM defined by policy space $\mathcal{Q}$ and affine biases $\mathbf{b} \geq \mathbf{0}$. By non-negativity and the halting condition ($\varrho(\mathbf{Q}) < 1$ for all $\mathbf{Q} \in \mathcal{Q}$), there exists a bounding vector $\mathbf{v} \geq \mathbf{1}$ such that $\mathbf{v} \geq \mathbf{Q}\mathbf{v} + \mathbf{b}$ for all $\mathbf{Q} \in \mathcal{Q}$. The existence of $\mathbf{v}$ follows from the second property in Proposition 1, and it is computable in polynomial time.

Let $\Lambda = [\mathrm{diag}(\mathbf{v})]^{-1}$. To map the system to an SSG with strictly unweighted min/max operators, we construct a bipartite expanded system over states $Y \cup Z$. Let $Y = \{y_1, \ldots, y_n\}$ be the unweighted choice states, and let $Z = \{z_{i,l} \mid i \in S_{\min} \cup S_{\max}, l \in \mathcal{N}(i)\} \cup \{z_k \mid k \in S_{\text{aff}}\}$ be the rescaled affine transition states. The closed-form equivalent LEMM is defined by the equations:

$$\begin{cases} y_i = \min_{l \in \mathcal{N}(i)} z_{i,l}, & i \in S_{\min}\,, \\ y_j = \max_{l \in \mathcal{N}(j)} z_{j,l}, & j \in S_{\max}\,, \\ y_k = z_k, & k \in S_{\text{aff}}\,, \\ z_{i,l} = \left(\frac{v_l}{v_i}\right) y_l, & \forall i \in S_{\min} \cup S_{\max},\ l \in \mathcal{N}(i)\,, \\ z_k = \sum_{m=1}^{n} \left(q_{km} \frac{v_m}{v_k}\right) y_m + \frac{b_k}{v_k}, & \forall k \in S_{\text{aff}}\,. \end{cases}$$

By substituting the Z-equations into the Y-equations, this system algebraically recovers the relation $\mathbf{y} = \Lambda\mathbf{x}$. We now formally verify the three SSG conditions for this expanded system:

1. Non-negativity: Since $\mathbf{Q} \geq \mathbf{O}$, $\mathbf{b} \geq \mathbf{0}$, and $\mathbf{v} > \mathbf{0}$, all transition coefficients and constant biases in the expanded system are non-negative.

2. Sum-up-to-one: For any fixed policy, the sum of the coefficients and the constant bias in each state equation is bounded by 1:

- For any $y \in Y$, the equation selects exactly one $z \in Z$ with coefficient 1 and bias 0, summing to $1 \leq 1$.
- For $z_{i,l}$, there exists a policy matrix $\mathbf{Q} \in \mathcal{Q}$ selecting choice l at state i, which by the bounding inequality yields $v_i \geq v_l$. The sum is $v_l/v_i + 0 \leq 1$.
- For z_k, the bounding inequality $\mathbf{v} \geq \mathbf{Q}\mathbf{v} + \mathbf{b}$ evaluated at row k ensures the sum is $v_k^{-1}(\mathbf{q}_k^{\mathrm{T}}\mathbf{v} + b_k) \leq v_k^{-1} v_k = 1$.

3. Halting: Let $\mathbf{P}$ be the transition matrix of an arbitrary policy in the expanded system. Because the state equations depend strictly on alternating sets, $\mathbf{P}$ has a bipartite block structure:

$$\mathbf{P} = \begin{bmatrix} \mathbf{O} & \mathbf{P}_{YZ} \\ \mathbf{P}_{ZY} & \mathbf{O} \end{bmatrix} \implies \mathbf{P}^2 = \begin{bmatrix} \mathbf{P}_{YZ}\mathbf{P}_{ZY} & \mathbf{O} \\ \mathbf{O} & \mathbf{P}_{ZY}\mathbf{P}_{YZ} \end{bmatrix}.$$

A policy fixes a choice $l(i) \in \mathcal{N}(i)$ for each $i \in S_{\min} \cup S_{\max}$. The matrix $\mathbf{P}_{YZ}$ maps Y to Z: row i has a 1 at column $z_{i,l(i)}$, and row $k \in S_{\text{aff}}$ has a 1 at column z_k. The matrix $\mathbf{P}_{ZY}$ maps Z to Y with the respective rescaled coefficients.

The principal block $\mathbf{P}_{YZ}\mathbf{P}_{ZY}$ represents the two-step transition from Y to Y. Evaluating this product maps exactly to the rescaled matrix of the original system: $\mathbf{P}_{YZ}\mathbf{P}_{ZY} = \Lambda\mathbf{Q}\Lambda^{-1}$, where $\mathbf{Q} \in \mathcal{Q}$ is the original policy making the identical choices $l(i)$.

For any bipartite block matrix, the non-zero eigenvalues of $\mathbf{P}^2$ are exactly the non-zero eigenvalues of $\mathbf{P}_{YZ}\mathbf{P}_{ZY}$. Consequently, the spectral radius of $\mathbf{P}$ satisfies:

$$\varrho(\mathbf{P})^2 = \varrho(\mathbf{P}^2) = \varrho(\mathbf{P}_{YZ}\mathbf{P}_{ZY}) = \varrho(\Lambda\mathbf{Q}\Lambda^{-1}) = \varrho(\mathbf{Q})\,.$$

Because the original system satisfies the halting condition ($\varrho(\mathbf{Q}) < 1$), it follows that $\varrho(\mathbf{P}) < 1$. Thus, the expanded system strictly halts.

Since the expanded system satisfies non-negativity, sum-up-to-one, and halting with unweighted min and max operators, it is formally a Simple Stochastic Game (SSG), completing the linear reduction. □

Remark 1 (Algorithmic Implications of the SSG Reduction). While Theorem 3 establishes a linear reduction from Class I LEMMs to SSGs, this transformation serves primarily as an analytical tool rather than a required algorithmic preprocessing step. To achieve unweighted min/max operators, the reduction introduces intermediate affine states (i.e., the $z_{i,l}$ variables). Because of this expanded bipartite structure, executing standard Value Iteration on the constructed SSG graph would effectively require two iterations to match a single iteration of the Bellman operator applied directly to the original Class I LEMM. Therefore, in practice, iterative algorithms like Value Iteration can and should be run natively on the unexpanded system, thereby enjoying the SSG theoretical guarantees without incurring the computational overhead of constructing the reduction or the $2\times$ iteration penalty.

C Missing Proofs for Absolutely Halting LEMMs

To prove the uniform contraction of the Bellman operator under arbitrary changing policies, we first establish a structural property of the policy space. In general matrix theory, the fact that every individual matrix in a set has a spectral radius strictly less than one does not imply that arbitrary products of these matrices will converge to zero (i.e., fixed-policy stability does not imply a joint spectral radius less than one). However, because our policy space $\mathcal{Q}$ is constructed from independent state-wise choices (a Cartesian product), we can prove that fixed-policy stability is sufficient to guarantee uniform contraction across arbitrary policy sequences.

Proposition 7 (Uniform n-step Contraction for Cartesian Matrix Families)**.** *Let $\mathcal{Q}$ be a finite family of $n \times n$ matrices with non-negative entries satisfying three properties:*

1. *Row Sub-stochasticity: $\mathbf{Q}\mathbf{1} \leq \mathbf{1}$ for all $\mathbf{Q} \in \mathcal{Q}$.*
2. *Cartesian Product Structure: For any $\mathbf{Q}, \mathbf{Q}' \in \mathcal{Q}$ and any index $i \in [n]$, the matrix formed by replacing the i-th row of $\mathbf{Q}$ with the i-th row of $\mathbf{Q}'$ is also in $\mathcal{Q}$.*
3. *Fixed-Policy Halting: $\varrho(\mathbf{Q}) < 1$ for all $\mathbf{Q} \in \mathcal{Q}$.*

Let $p_{\min} = \min\{Q_{ij} \mid Q_{ij} > 0, \mathbf{Q} \in \mathcal{Q}\}$. Define the minimum positive leak as $\delta_{\min} = \min\{1 - [\mathbf{Q}\mathbf{1}]_i \mid [\mathbf{Q}\mathbf{1}]_i < 1, \mathbf{Q} \in \mathcal{Q}, i \in [n]\}$. For any arbitrary sequence of n matrices $\mathbf{Q}^{(1)}, \mathbf{Q}^{(2)}, \ldots, \mathbf{Q}^{(n)} \in \mathcal{Q}$, their product $\mathbf{P} = \mathbf{Q}^{(n)}\mathbf{Q}^{(n-1)} \cdots \mathbf{Q}^{(1)}$ satisfies:

$$\|\mathbf{P}\mathbf{1}\|_\infty \leq 1 - p_{\min}^{n-1}\delta_{\min}\,.$$

Proof of Proposition 7. Because $\mathbf{Q}\mathbf{1} \leq \mathbf{1}$ and $\mathbf{Q} \geq \mathbf{O}$ for all $\mathbf{Q} \in \mathcal{Q}$, it holds that $\mathbf{P}\mathbf{1} \leq \mathbf{1}$. We will directly prove that every element of $\mathbf{P}\mathbf{1}$ is bounded by $1 - p_{\min}^{n-1}\delta_{\min}$.

Let $i_0 \in [n]$ be an arbitrary initial state. Define the sequence of applied matrices as $\mathbf{M}_k = \mathbf{Q}^{(n-k+1)}$ for $k \in [n]$, yielding the product $\mathbf{P} = \mathbf{M}_1\mathbf{M}_2 \cdots \mathbf{M}_n$.

We track the states reachable from i_0 at each step. Let $S_0 = \{i_0\}$, and for $k \in [n]$, define the reachable sets $S_k = \{j \in [n] \mid \exists i \in S_{k-1} \text{ s.t. } [\mathbf{M}_k]_{ij} > 0\}$. By the definition of $p_{\min}$, any state $i \in S_k$ is reached with an accumulated transition weight of at least $[\mathbf{M}_1 \cdots \mathbf{M}_k]_{i_0,i} \geq p_{\min}^k$.

Define the cumulative reachable sets $R_k = \bigcup_{m=0}^{k} S_m$. This yields a nested sequence $R_0 \subseteq R_1 \subseteq \cdots \subseteq R_n \subseteq [n]$. Because R_0 contains exactly 1 state and the maximum possible size is n, the cardinality can strictly increase at most $n-1$ times. By the Pigeonhole Principle, there must exist an index $k^* \in \{0, \ldots, n-1\}$ such that $R_{k^*} = R_{k^*+1}$. Let $V = R_{k^*}$.

We construct a fixed policy matrix $\mathbf{Q}^* \in \mathcal{Q}$. For each state $i \in V$, let $m(i)$ be the smallest integer such that $i \in S_{m(i)}$, noting that $0 \leq m(i) \leq k^* \leq n-1$. We set the i-th row of $\mathbf{Q}^*$ to exactly match the i-th row of $\mathbf{M}_{m(i)+1}$. For states $i \notin V$, we assign arbitrary valid rows from $\mathcal{Q}$. The Cartesian Product property guarantees $\mathbf{Q}^* \in \mathcal{Q}$.

Under this fixed matrix $\mathbf{Q}^*$, any positive transition from a state $i \in V$ goes to a state j where $[\mathbf{M}_{m(i)+1}]_{ij} > 0$. By the definition of our sets, this requires $j \in S_{m(i)+1}$. Because $m(i) \leq k^*$, we have:

$$S_{m(i)+1} \subseteq R_{m(i)+1} \subseteq R_{k^*+1} = V\,.$$

Thus, the subset of states V is completely closed under $\mathbf{Q}^*$.

Because $\mathcal{Q}$ has the Fixed-Policy Halting property, $\varrho(\mathbf{Q}^*) < 1$. This requires that $\mathbf{Q}^*$ cannot contain any closed, perfectly row-stochastic submatrices (otherwise it would have an eigenvalue of 1). Therefore, there must exist at least one state $i^* \in V$ such that its row in $\mathbf{Q}^*$ strictly sums to less than 1.

Since the i^*-th row of $\mathbf{Q}^*$ is identical to the i^*-th row of $\mathbf{M}_{m(i^*)+1}$, it follows that $[\mathbf{M}_{m(i^*)+1}\mathbf{1}]_{i^*} < 1$. By the definition of the minimum leak $\delta_{\min}$, this implies:

$$\mathbf{M}_{m(i^*)+1}\mathbf{1} \leq \mathbf{1} - \delta_{\min}\mathbf{e}_{i^*} ,$$

where $\mathbf{e}_{i^*}$ is the standard basis vector.

We directly evaluate the effect of this localized leak on the total product. Because all matrices are non-negative and sub-stochastic, multiplying by the preceding sequence of matrices preserves the inequality:

$$\begin{aligned}\mathbf{P}\mathbf{1} \leq \mathbf{M}_1 \cdots \mathbf{M}_{m(i^*)+1}\mathbf{1} &\leq \mathbf{M}_1 \cdots \mathbf{M}_{m(i^*)}(\mathbf{1} - \delta_{\min}\mathbf{e}_{i^*}) \\ &= \mathbf{1} - \delta_{\min}\mathbf{M}_1 \cdots \mathbf{M}_{m(i^*)}\mathbf{e}_{i^*} .\end{aligned}$$

Evaluating the i_0-th coordinate yields:

$$[\mathbf{P}\mathbf{1}]_{i_0} \leq 1 - \delta_{\min}[\mathbf{M}_1 \cdots \mathbf{M}_{m(i^*)}]_{i_0,i^*} .$$

Because $i^* \in S_{m(i^*)}$, the sequence reaches i^* at step $m(i^*)$ with weight $[\mathbf{M}_1 \cdots \mathbf{M}_{m(i^*)}]_{i_0,i^*} \geq p_{\min}^{m(i^*)}$. Since $m(i^*) \leq n-1$ and $p_{\min} \leq 1$, we have $p_{\min}^{m(i^*)} \geq p_{\min}^{n-1}$. Substituting this into the bound gives:

$$[\mathbf{P}\mathbf{1}]_{i_0} \leq 1 - p_{\min}^{n-1}\delta_{\min} .$$

Because the choice of the initial state i_0 was arbitrary, this upper bound applies to all elements, establishing $\|\mathbf{P}\mathbf{1}\|_\infty \leq 1 - p_{\min}^{n-1}\delta_{\min}$. □

Proof of Lemma 4. We first establish a pointwise bound for a single application of the Bellman operator $\mathcal{T}$. Let $\mathbf{u}, \mathbf{u}' \in \mathbb{R}^n$ be two arbitrary value vectors.

For any affine row $k \in S_{\text{aff}}$, the absolute difference is exactly linear:

$$|[\mathcal{T}(\mathbf{u})]_k - [\mathcal{T}(\mathbf{u}')]_k| = |\mathbf{q}_k^{\mathrm{T}}(\mathbf{u} - \mathbf{u}')| \leq |\mathbf{q}_k|^{\mathrm{T}}|\mathbf{u} - \mathbf{u}'| .$$

For any max row $j \in S_{\max}$, let $l = \arg\max_{m\in\mathcal{N}(j)} u_m$ and $l' = \arg\max_{m\in\mathcal{N}(j)} u'_m$. Because $u_l \geq u_{l'}$ and $u'_{l'} \geq u'_l$, we can bound the difference:

$$u_{l'} - u'_{l'} \leq u_l - u'_{l'} \leq u_l - u'_l .$$

Consequently, $|[\mathcal{T}(\mathbf{u})]_j - [\mathcal{T}(\mathbf{u}')]_j| \leq \max(|u_l - u'_l|, |u_{l'} - u'_{l'}|)$. The absolute difference is bounded by the absolute difference at a valid transition choice. The same logic symmetrically holds for the min rows.

Therefore, there exists a policy matrix $\bar{\mathbf{Q}} \in \mathcal{Q}$, composed of these worst-case pointwise choices, such that:

$$|\mathcal{T}(\mathbf{u}) - \mathcal{T}(\mathbf{u}')| \leq |\bar{\mathbf{Q}}||\mathbf{u} - \mathbf{u}'| .$$

Multiplying both sides by the diagonal rescaling matrix $\Lambda = [\mathrm{diag}(\mathbf{v})]^{-1}$, and inserting $\Lambda^{-1}\Lambda = \mathbf{I}$, we obtain:

$$\Lambda|\mathcal{T}(\mathbf{u}) - \mathcal{T}(\mathbf{u}')| \leq \Lambda|\bar{\mathbf{Q}}|\Lambda^{-1}\Lambda|\mathbf{u} - \mathbf{u}'| = \tilde{\mathbf{Q}}|\Lambda(\mathbf{u} - \mathbf{u}')| , \tag{4}$$

where $\tilde{\mathbf{Q}} = \Lambda|\bar{\mathbf{Q}}|\Lambda^{-1}$ is a matrix from the rescaled absolute policy space $\tilde{\mathcal{Q}}$.

By iteratively applying this 1-step bound n times, there exists a sequence of matrices $\tilde{\mathbf{Q}}^{(1)}, \ldots, \tilde{\mathbf{Q}}^{(n)} \in \tilde{\mathcal{Q}}$ such that:

$$|\Lambda\big(\mathcal{T}^n(\mathbf{u}) - \mathcal{T}^n(\mathbf{u}')\big)| \leq \Big(\tilde{\mathbf{Q}}^{(n)} \cdots \tilde{\mathbf{Q}}^{(1)}\Big)|\Lambda(\mathbf{u} - \mathbf{u}')| . \tag{5}$$

Let $\mathbf{P} = \tilde{\mathbf{Q}}^{(n)} \cdots \tilde{\mathbf{Q}}^{(1)}$. The family of matrices $\tilde{\mathcal{Q}}$ inherently satisfies all three conditions of Proposition 7: (1) $\tilde{\mathbf{Q}}\mathbf{1} \leq \mathbf{1}$ by the definition of the bounding vector $\mathbf{v}$; (2) the Cartesian product structure is preserved from $\mathcal{Q}$; and (3) absolute halting guarantees $\max_{\tilde{\mathbf{Q}}\in\tilde{\mathcal{Q}}} \varrho(\tilde{\mathbf{Q}}) = \max_{\mathbf{Q}\in\mathcal{Q}} \varrho(|\mathbf{Q}|) < 1$.

Applying Proposition 7 yields the uniform row-sum bound $\|\mathbf{P1}\|_\infty \le 1 - \tilde{p}_{\min}^{n-1}\tilde{\delta}_{\min}$. Taking the ℓ_∞-norm of Eq. (5) gives:

$$\begin{aligned}\|\Lambda\big(\mathcal{T}^n(\mathbf{u}) - \mathcal{T}^n(\mathbf{u}')\big)\|_\infty &\le \|\mathbf{P1}\|_\infty \|\Lambda(\mathbf{u} - \mathbf{u}')\|_\infty \\ &\le (1 - \tilde{p}_{\min}^{n-1}\tilde{\delta}_{\min})\|\Lambda(\mathbf{u} - \mathbf{u}')\|_\infty ,\end{aligned}$$

which concludes the proof. □

Proof of Theorem 5. Part (1) follows directly from the counterexample provided in Example 1.

For Part (2), let $\mathbf{x}^*$ be the unique exact solution, which satisfies the fixed point equation $\mathcal{T}(\mathbf{x}^*) = \mathbf{x}^*$. Applying Lemma 4 with $\mathbf{u}' = \mathbf{x}^*$, we directly obtain the n-step linear convergence in the rescaled norm:

$$\|\Lambda(\mathbf{u}^{(kn)} - \mathbf{x}^*)\|_\infty \le (1 - \tilde{p}_{\min}^{n-1}\tilde{\delta}_{\min})^k \|\Lambda(\mathbf{u}^{(0)} - \mathbf{x}^*)\|_\infty , \quad \text{for all } k \ge 1 .$$

□

D Missing Proofs for General Halting LEMMs

Proof of Theorem 6. (1) The divergence of PImax and VI follows from the counterexamples provided in Example 1 and Example 3.

(2) **SPI Convergence:** We proceed by induction on the number of non-linear variables, $N = |S_{\min}| + |S_{\max}|$. Let the non-linear variables be indexed as $1, 2, \ldots, N$ according to a fixed, arbitrary ordering.

Base Case ($N = 0$): The system is affine. By the halting condition, $\varrho(\mathbf{Q}_{\text{aff}}) < 1$, which implies $(\mathbf{I} - \mathbf{Q}_{\text{aff}})$ is invertible. The algorithm solves the linear system and returns the unique exact solution in one iteration.

Inductive Step: Assume the algorithm returns the unique exact solution in at most $\prod_{i=1}^{N-1} |\mathcal{N}(i)|$ iterations for any general halting LEMM with $N - 1$ non-linear variables. Consider a system with N non-linear variables. The uniqueness of the exact solution $\mathbf{x}^*$ of a halting system follows from the first property in Proposition 1.

At any step, state N is assigned a choice $c \in \mathcal{N}(N)$. Because the index of N is greater than all other non-linear states, SPI updates states $1, \ldots, N - 1$ as long as $\min(\Gamma) < N$. This is equivalent to executing SPI on the $(N - 1)$-variable subsystem where the choice at state N is fixed to c.

Fixing the choice at state N restricts the policy space to a Cartesian subset of $\mathcal{Q}$. Consequently, this subsystem inherits the general halting condition ($\varrho(\mathbf{Q}) < 1$). By the first property in Proposition 1, this subsystem possesses a unique solution, denoted $\mathbf{u}^{(c)}$.

By the inductive hypothesis, the inner loop of SPI reaches $\mathbf{u}^{(c)}$ in at most $\prod_{i=1}^{N-1} |\mathcal{N}(i)|$ iterations. At $\mathbf{u}^{(c)}$, there are no violations among the first $N - 1$ equations. SPI then evaluates the set of violated states Γ for the full system.

- If $\mathbf{u}^{(c)} = \mathbf{x}^*$, then $\mathbf{u}^{(c)}$ satisfies all equations of the full system. Thus, $\Gamma = \varnothing$, and the algorithm terminates.
- If $\mathbf{u}^{(c)} \neq \mathbf{x}^*$, then $\mathbf{u}^{(c)}$ is not a solution to the full system. Because $\mathbf{x}^*$ is the unique solution to the full system, $\mathbf{u}^{(c)}$ must violate at least one equation. Since states $1, \ldots, N - 1$ are satisfied at $\mathbf{u}^{(c)}$, the violation must occur at state N. Therefore, $\Gamma = \{N\}$, which yields $\min(\Gamma) = N$. The algorithm then advances the choice at state N to the next element in $\mathcal{I}(N)$.

Because the algorithm advances the choice at state N whenever $\mathbf{u}^{(c)} \neq \mathbf{x}^*$, it enumerates the choices in $\mathcal{I}(N)$. Since the original problem has a unique solution $\mathbf{x}^*$, there exists at least one choice at state N that induces $\mathbf{x}^*$. The algorithm evaluates at most $|\mathcal{N}(N)|$ choices at state N. For each choice, the $(N - 1)$-variable subsystem requires at most $\prod_{i=1}^{N-1} |\mathcal{N}(i)|$ iterations. The total number of iterations

is bounded by:

$$|\mathcal{N}(N)| \cdot \prod_{i=1}^{N-1} |\mathcal{N}(i)| = \prod_{i=1}^{N} |\mathcal{N}(i)|,$$

which completes the induction.

(3) **RandSPI Convergence:** Let $\mathcal{T}(k)$ denote the number of recursive calls required to solve a system with k non-linear variables. At each recursive level k, there exists at least one optimal choice in $\mathcal{N}(k)$ that yields the exact solution.

Because RandSPI evaluates the choices at state k according to a permutation $\mathcal{I}(k)$ drawn uniformly at random, the expected index of the optimal choice is $(|\mathcal{N}(k)| + 1)/2$. If the optimal choice is located at index j, the algorithm makes j recursive calls. Taking the expectation over the uniform distribution yields the recurrence relation:

$$\mathbb{E}[\mathcal{T}(k)] \leq \sum_{j=1}^{|\mathcal{N}(k)|} \frac{j}{|\mathcal{N}(k)|} \cdot \mathbb{E}[\mathcal{T}(k-1)] = \frac{|\mathcal{N}(k)| + 1}{2} \mathbb{E}[\mathcal{T}(k-1)].$$

Unrolling this recursion over all non-linear states yields the final upper bound:

$$\mathbb{E}[\mathcal{T}(|S_{\min}| + |S_{\max}|)] \leq \frac{\prod_{i \in S_{\min} \cup S_{\max}} (|\mathcal{N}(i)| + 1)}{2^{|S_{\min}| + |S_{\max}|}}.$$

□

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the claims made, including the contributions made in the paper and important assumptions.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the remaining open question and potential future work in the conclusion section of the paper.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The presentation of the results are self-contained and all the assumptions are clearly stated. The proofs are either in the main paper or in the appendix, and they are correct.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [N/A]

   Justification: This paper does not include experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [N/A]

Justification: This paper does not include experiments.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [N/A]

Justification: This paper does not include experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [N/A]

Justification: This paper does not include experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [N/A]

   Justification: This paper does not include experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

   Justification: This paper is conducted in accordance with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release any data or models.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

    Justification: The paper does not use any existing assets.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release any new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not include crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not include research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are only used for writing, editing, and formatting purposes in this paper, and do not impact the core methodology, scientific rigor, or originality of the research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.