The 38th Canadian Conference on Artificial Intelligence
DOI: 0

Canadian AI 2025
26—29 May 2025, Calgary, Alberta

Topology-Driven Solver Selection for Stochastic Shortest Path MDPs via Explainable Machine Learning

Mathieu Gravel[†], Jaël Champagne Gareau[†,*]
[†] Université du Québec à Montréal

Abstract

Selecting optimal solvers for complex AI tasks grows increasingly difficult as algorithmic options expand. We address this challenge for Stochastic Shortest Path Markov Decision Processes (SSP-MDPs) — a core model for robotics navigation, autonomous system planning, and stochastic scheduling — by introducing a topology-driven solver selection framework. First, we identify and empirically validate topological features — including strongly connected components, goal-state ratio, goal eccentricity (*i.e.*, maximal distance to a goal), and average actions per state — that critically influence solver performance across synthetic and real-world SSP-MDPs. Using these insights, we propose the first classifier able to predict the fastest MDP solver for a given instance, achieving over 64% accuracy on diverse benchmarks. Counterfactual explainability analysis further clarifies how these features govern solver efficiency. By directly linking topological structures to algorithmic performance, our work streamlines solver selection while enhancing computational efficiency, offering a principled approach to MDP optimization.

Keywords: Markov Decision Processes, Stochastic Shortest Path, Solver Selection, Topological Insights, Classification, Explainable AI, Counterfactual Explanations

1. **Introduction**

Automated planning and scheduling is a core problem in artificial intelligence (AI) with a wide range of applications, including robotics, logistics, and management [1]. When planning problems involve quantifiable uncertainties, Markov Decision Processes (MDPs) are often used as a modeling framework. In particular, Stochastic Shortest Path MDPs (SSP-MDPs) are a subclass of MDPs that focus on reaching one (of possibly many) goal state from a given initial state (or from any state), with the objective of minimizing the expected cost [2]. The transition probabilities are assumed to be known, and each action has an associated cost. Notably, many other MDP subclasses — such as infinite-horizon discounted MDPs, widely used in the reinforcement learning community — can be viewed as special cases of SSP-MDPs [3]. Throughout this paper, we focus exclusively on SSP-MDPs.

The inherent complexity of SSP-MDPs has spurred the development of diverse solvers, each leveraging distinct algorithmic strategies and assumptions tailored to specific structural or cost-related conditions. While this specialization enables performance advantages in targeted scenarios, the sheer variety of available solvers renders manual selection for a given problem both time-intensive and error-prone. To address this challenge, we introduce a topology-driven framework to automate solver selection, anchored in four key contributions:

- An overview of topological features predictive of solver performance.
- A first classifier in the literature able to predict the fastest solver for a given MDP.
- An explanability system for the classifier, to provide human-ready explanations.
- An extensive empirical evaluation across 8,000+ synthetic and real-world SSP-MDP instances, demonstrating our method's generalizability and practical utility.

The remainder of this paper is organized as follows. Section 2 surveys existing SSP-MDP planning algorithms, contextualizes scenarios of solver superiority, and reviews explainable

[*] champagne_gareau.jael@uqam.ca

AI methods for interpreting classifier decisions. Section 3 introduces extreme SSP-MDP configurations, benchmarks solver performance on these domains, and identifies critical observations that underpin our topology-driven approach. Section 4 details the design, training (via machine learning approaches such as iterative random forests), and evaluation of our topology-aware solver-prediction model. Section 5 evaluates the framework's accuracy and generalizability across synthetic and real-world benchmarks. Finally, Section 6 synthesizes key insights and proposes future directions for topology-informed solver selection.

2. Related Work

2.1. Markov Decision Processes

We use the formalism from the work of Mausam and Kolobov [4] to define the Stochastic Shortest Path MDPs (Definition 1). Research on solving such MDPs has led to a variety of algorithms, each leveraging different techniques and optimizations. Broadly, these algorithms can be categorized as follows: classical dynamic programming methods, prioritization-based approaches, heuristic-based approaches, and hybrid methods.

Definition 1. A *Stochastic Shortest Path* (SSP) MDP is a tuple (S, A, T, C, G), where:

- S is a finite set of states;
- A is a finite set of actions;
- $T\colon S \times A \times S \to [0, 1]$ is a transition function, with $T(s, a, s')$ giving the probability of transitioning to state s' when applying action a in state s;
- $C\colon S \times A \to \mathbb{R}^+$ is a cost function, where $C(s, a)$ gives the cost of applying action a while in state s[1];
- $G \subseteq S$ is the set of goal states (which can be assumed to be terminal or sink states).

Two foundational algorithms for solving MDPs are *Value Iteration* (VI) [5] and *Policy Iteration* (PI) [6]. Both rely on iteratively refining either a value function or a policy until convergence, guided by the Bellman equations (Definition 2).

Definition 2. The Bellman equations for SSP-MDPs are, $\forall s \in S$:

$$V(s) = \begin{cases} 0 & \text{if } s \in G, \\ \min_{a \in A} \left[C(s,a) + \sum_{s' \in S} T(s,a,s')V(s) \right] & \text{otherwise.} \end{cases}$$

In practice, Value Iteration is often difficult to outperform for dense MDP topologies — *i.e.*, when most states have actions that connect them to many other states — because common optimizations in other methods (*e.g.*, state-space pruning or state prioritization) tend to be less effective in such scenarios. Dense MDPs arise in various real-world applications, for instance in certain configurations of the popular SysAdmin planning domain [7].

A second category of algorithms prioritizes specific states during the update process rather than sweeping through the state space in an arbitrary order. A well-known example is *Improved Prioritized Sweeping* (IPS) [8], which uses the metric $IPS(s) = \frac{Res(s)}{V(s)}$, where $Res(s)$ is the *residual* (also called the *Bellman error*) and indicates how much the value of s would change if it were to be updated with the Bellman equations. Intuitively, the numerator of the metric captures how far the state is from convergence, while the denominator reflects how close the state is to a goal state. These prioritization methods can significantly reduce the number of Bellman backups required for convergence, but the overhead of maintaining

[1]The cost function can also be defined as $C\colon S \times A \times S \to \mathbb{R}^+$, and is generally assumed to be probabilistic. In our experiments, we assume a deterministic cost function, though extending the framework to handle probabilistic costs is straightforward.

the priority queue often offset these gains. Moreover, the choice of priority metric is critical, as different metrics can perform better or worse depending on the given MDP instance.

To mitigate the cost of priority-queue maintenance, *Topological Value Iteration* (TVI) [9] uses the graph structure of the MDP to order the state updates. By computing a topological ordering of the strongly connected components (SCCs), TVI avoids the overhead of managing dynamic priorities altogether. It works particularly well on MDPs containing a large number of SCCs, as the topological ordering can significantly improve state-values propagation and reduce the number of Bellman updates required for convergence.

Heuristic-based methods assume an initial state s_0 is known and require a heuristic function $h\colon S \to \mathbb{R}^+$ that estimates the cost to reach a goal state from each state. Two common algorithms in this category are *Labeled Real-Time Dynamic Programming* (LRTDP) [10] and *Improved Loop And/Or** (ILAO*) [11]. Although heuristics can guide the search efficiently, domain-independent heuristics are often weak, and developing strong domain-specific heuristics requires additional effort. This trade-off can limit the usefulness of heuristic-based methods unless high-quality heuristics are readily available, which is often not the case in practice. Heuristic-based methods are known to perform well on MDPs with a large number of goal states, as the heuristic can guide the search towards the nearest one.

Hybrid solvers merge complementary algorithmic paradigms to balance trade-offs. For example, FTVI [9] integrates TVI's topological ordering with LAO*-style heuristic guidance, combining their complementary strengths while retaining their inherent limitations. Over the past decade, solver development has focused on efficiency gains through two primary strategies: (1) cache optimization (*e.g.*, CECA [12] and eTVI [13]) and (2) parallel computation (*e.g.*, multicore-CECA [14] and pcTVI [15]). These advancements refine existing methods rather than introducing fundamentally new approaches. By exploring these various algorithmic strategies, researchers have provided a diverse toolkit for tackling SSP-MDPs. However, no single solver is universally optimal; each approach excels under specific problem characteristics, motivating the need for robust solver-selection techniques.

2.2. **Explainable AI and Machine Learning**

The opacity of modern machine learning models poses critical challenges in domains requiring auditability, such as healthcare or autonomous systems, where human oversight demands actionable explanations for algorithmic decisions. In the context of MDP solver selection, this lack of transparency risks eroding trust in automated recommendations, particularly when structural domain properties (*e.g.*, goal eccentricity, SCCs) govern solver efficiency. Explainable AI (XAI) addresses this gap by enabling systems to justify their predictions through human-interpretable rationales. For our topology-driven framework, XAI serves two purposes: (1) validating that solver recommendations align with theoretical expectations of MDP structure, and (2) empowering domain experts to refine topological feature extraction based on empirical insights. XAI broadly fall into two categories [16]:

- Interpretable Models (*e.g.*, rule-based systems) prioritize transparency by design, offering intrinsic explanations through human-readable structures like explicit decision paths. These models trade marginal accuracy gains of more opaque approaches for direct analyzability, making them ideal for domains like ours where feature importance must be rigorously mapped (*e.g.*, Iterative Random Forests [17]).
- Model-Agnostic Methods treat models as black boxes, deriving post hoc explanations through techniques like counterfactual analysis or feature attribution. While flexible, they introduce computational overhead and approximate explanations. Local methods (*e.g.*, LIME [18]) explain individual predictions (Why was VI chosen for this MDP?), while global methods (*e.g.*, SHAP [19]) identify overarching feature importance (How does SCC count generally affect solver performance?).

Recent advances in explainable AI (XAI), particularly counterfactual explanations [20], enable "what-if" reasoning by identifying minimal feature perturbations that alter a classifier's prediction from one class to another. For instance, in our framework, this answers questions like "How much must the SCC count increase for TVI to become the recommended solver instead of LRTDP?" The DiCE (Diverse Counterfactual Explanations) algorithm implements this capability through model-agnostic optimization. DiCE generates diverse counterfactuals by (1) minimizing the distance between original inputs and counterfactuals (proximity), (2) maximizing differences between generated examples (diversity), and (3) prioritizing sparse feature changes (actionability). By balancing these objectives via gradient-based optimization, DiCE produces human-interpretable explanations that reveal how specific topological features govern solver recommendations in data.

While counterfactuals have been applied to policy analysis in MDPs [21], our work pioneers their use for solver selection, explicitly linking topological properties to algorithmic efficacy. Such techniques align with our framework's goal of linking topological properties to solver efficacy. While deep learning methods increasingly adopt XAI algorithms (*e.g.*, attention maps for transformers), our work leverages classical interpretable models and model-agnostic techniques to balance precision and explainability, ensuring domain experts can both trust and refine solver recommendations without sacrificing predictive accuracy.

3. Motivation

A recent research paper proposed a list of topological features hypothesized to influence the performance of MDP solvers but did not systematically analyze their individual impacts or evaluate their relative significance [22]. That work also introduced an algorithm for generating synthetic MDPs from predefined graph topologies, which we adopt to create a diverse set of MDP instances for our experiments. In this section, we establish the relevance of topological properties in predicting solver performance by presenting extreme cases of MDP planning domains and benchmarking widely-used solvers on these scenarios.

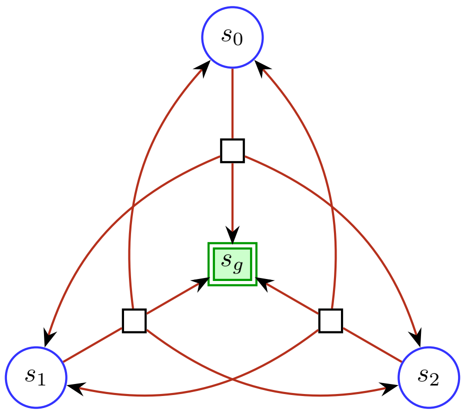

Figure 1. A dense SSP-MDP with three non-goal states. Each state has a single probabilistic action enabling transitions to all other states.

One illustrative extreme case is the *dense* MDP domain, where all n states are mutually reachable. We distinguish two sub-cases: (1) a deterministic variant where each state has $n-1$ actions, each directly transitioning to a distinct neighboring state, and (2) a stochastic variant where each state has a single action with non-zero probability of transitioning to any other state. Figure 1 shows the stochastic sub-case for $n = 3$ (excluding the goal s_g).

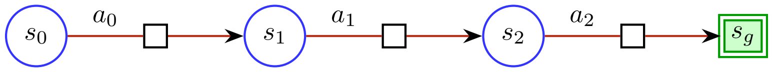

Figure 2. A unidirectional linear SSP-MDP. Each state transitions deterministically to the subsequent state in the chain.

Another extreme SSP-MDP configuration is the linear domain, where states form a single path to the goal (Figure 2). We identify three sub-cases: (1) *unidirectional deterministic*, where each state has a single action transitioning to the next state; (2) *bidirectional deterministic*, where each state has two actions — one advancing to the next state, the other returning to the previous state; and (3) *bidirectional stochastic*, where each state has a single probabilistic action with non-zero probability of progressing or regressing.

Table 1 compares the running time of four MDP solvers — VI, LRTDP, ILAO* and TVI — across diverse domain configurations. The second column, n, denotes the number of states in each domain. Key observations include:

- Heuristic search algorithms (LRTDP, ILAO*) excel in domains with deterministic actions, which can more easily be detected suboptimals and pruned from the state-space (*e.g.*, achieving 0.1 ms in denseDet);
- VI performs optimally in highly stochastic domains (*e.g.*, 196,211 ms in denseProb), as the structure of such MDPs makes it practically impossible to improve the convergence rate by using prioritization or heuristic techniques;
- TVI-based algorithms dominate in structured domains with strongly connected components (SCCs). For instance, in the unidirectional linear domain (linearUniDir), eiTVI completes in 129 ms by exploiting SCCs to perform a single Bellman backup per state, while VI requires 2,594,650 ms due to only doing one useful Bellman backup per complete sweep through the state-space.

Table 1. Running times (ms) obtained for each solvers on the tested domains. The fastest time on each domain is bolded.

Name	n	VI	LRTDP	ILAO*	TVI
linearUniDir	10000000	2594650	564	16912088	**497**
linearBidirDet	100000	1577974	**9**	>1h	1721993
linearBidirProb	130	983	4579	2885	**982**
denseDet	10000	1660	**0.1**	**0.1**	1825
denseProb	10000	**196211**	684211	676912	208256

Another critical observation emerges in the bidirectional stochastic linear domain: testing instances with more than approximately 130 states becomes computationally prohibitive due to LRTDP's exponential runtime growth. This inefficiency arises from LRTDP's reliance on sampling trajectories from the initial state until a goal is reached. In such domains, the probability of reaching the goal diminishes exponentially with the number of states — a behavior analogous to an unbiased random walk — making successful goal attainment increasingly unlikely within practical time constraints. To our knowledge, this work is the first to identify a previously unreported limitation of LRTDP: its inadequacy in domains where even the optimal policy inherently exhibits a low probability of reaching the goal.

4. **Solver Selection Classifier**

This section details our framework for solver selection, including (1) the topological features predictive of solver performance and (2) the design and training of the classifier.

4.1. Topological Feature Engineering

We identified eight topological features that capture critical structural properties of SSP-MDPs, grouped into three categories:

Graph Scale & Complexity:

- **State Count** $|S|$: Directly impacts computational complexity and influences solver suitability (*e.g.*, SCC decomposition becomes less beneficial for small $|S|$).
- **Action Density**: Average actions per state, reflecting branching factor sensitivity.
- **Stochasticity**: Average probabilistic transitions per action, quantifying the level of uncertainty inside the planning domain.

Connectivity Structure:

- **Strongly Connected Components (SCCs)**: Total SCC count, critical for TVI-based algorithms leveraging topological ordering.
- **Max SCC Size**: Size of the largest SCC, indicating connectivity skew (*e.g.*, one dominant SCC vs. balanced partitioning).
- **Clustering Coefficient**: $\mathfrak{C} = \frac{1}{|S|} \sum_{s \in S} \frac{e_s}{k_s(k_s-1)}$, where e_s = directly connected state pairs from s, and k_s = directly reachable states from s ($\mathfrak{C} = 0$ if $k_s < 2$). Measures local "cliquishness" of transitions.

Goal-Oriented Properties:

- **Goal Density**: Ratio of goal states $\frac{|G|}{|S|}$, influencing heuristic solvers like ILAO*.
- **Goal Eccentricity**: $\mathcal{G} = \min_{g \in G} \max_{s \in S} \bar{d}(s, g)$, where $\bar{d}(s, g)$ = minimal action steps (ignoring costs) to reach g from s. Quantifies worst-case planning horizon.

4.2. Classifier Design and Training

Algorithm 1 Train Solver Selection Classifier

```
Require: MDP instances $\mathcal{I}$, solvers $\mathcal{S}$, features $\mathcal{F}$
1: for all $I \in \mathcal{I}$ do
2:     Extract features $\mathbf{F}(I)$ from $I$ using $\mathcal{F}$
3:     for all $S \in \mathcal{S}$ do
4:         Solve $I$ with $S$; record runtime $T(S, I)$
5:     Label $I$ with fastest solver arg $min_{S \in \mathcal{S}}$
6: Train the classifier $R$ on $\{(\mathbf{F}(I), \text{label}(I))\}$
7: Extract features importance from $R$
8: return $R$
```

Prior heuristic rules for solver selection (*e.g.*, "use TVI for domains with many SCCs") are limited in scope and fail to generalize across heterogeneous MDPs. To address this, we train an interpretable classifier (Iterative Random Forest [17]) on a corpus of $8,000+$ MDP. Each instance is labeled with the empirically fastest solver (among VI, LRTDP, ILAO* and TVI), enabling the model to learn mappings between topological features and optimal solver choices. Algorithm 1 outlines the training process. The framework supports configurable feature sets to balance training time and model complexity. For example, computing the clustering coefficient (Section 4.1) incurs $\mathcal{O}(n^2)$ time or space complexity (depending on the implementation), making it impractical for MDPs with over a million states. To mitigate this, users can choose to exclude computationally intensive features during configuration. To refine the feature set explainability, we employ two complementary models.:

- a global classifier (LightGBM [23]) predicts the fastest solver, prioritizing accuracy while maintaining explanability through feature importance rankings;

- solver-specific classifiers (Iterative Random Forests [17]) predict runtime distributions for individual solvers, capturing high-order feature interactions to explain edge cases (*e.g.*, why TVI underperforms in some cases despite high SCC counts).

To ensure transparency, we generate (1) counterfactual explanations [19], which identify minimal feature perturbations that alter solver recommendations (e.g., "TVI becomes optimal if SCCs increase by 20%"), and (2) rule-based insights, which extract decision rules from Iterative Random Forests using Gini impurity, highlighting thresholds (e.g., "Goal density > 0.1 favors LRTDP"). This dual approach balances predictive performance with actionable insights, enabling domain experts to validate and refine recommendations.

5. **Experiments**

This section evaluates the effectiveness of our topology-driven solver selection framework through three key steps: (1) we detail the synthetic and real-world MDP instances used for training and testing, emphasizing their structural diversity, (2) we describe the hardware and solver configurations to ensure reproducibility of runtime measurements and (3) we rigorously evaluate the global classifier's accuracy, dissect feature importance via counterfactual explanations, and validate individual solver performance predictions.

5.1. **MDP Instances used for training**

The classifier was trained on MDP instances generated from seven distinct models (Table 2), selected to represent diverse topological characteristics. The first four models — synthetic graph-theoretical structures — were converted into SSP-MDPs using the algorithm proposed by [22]. They cover various topological properties: Kronecker and Erdös-Rényi models are often less dense and have a larger number of SCCs, whereas Barabási-Albert and Watts-Strogatz (SmallWorld) models are denser and have usually fewer SCCs. The remaining three models are established real-world MDP benchmarks. The *Layered* domain is a parametric planning benchmark introduced in TVI's foundational work to study the impact of SCCs. Key parameters include SCC count (layers), states per layer, actions per state, and stochastic branching factors. The *Wetfloor* domain is a navigation problem used in the ICAPS international planning competitions [24], where agents crosses a grid while avoiding slippery tiles that induce stochastic motion. Finally, the *Single-Armed Pendulum* (SAP) domain is a continuous-state optimal control problem requiring precise balancing of an inverted pendulum. Its deterministic instance generator limits dataset diversity — each size n produces a single instance, forcing reliance on impractically large n for scalability.

Table 2. MDP Models used for the classifier training

Name	Reference	Number of instances
Erdös-Rényi	[25]	1614
Barabási-Albert	[26]	1989
Watts-Strogatz (SmallWorld)	[27]	1968
Kronecker	[28]	1448
Layered	[9]	880
WetFloor	[29]	200
Single-Armed Pendulum (SAP)	[30]	91

We extracted the eight topological features outlined in Section 4.1 across 8,190 MDP instances. The dataset was partitioned into an 80-20 training-test split to evaluate generalization performance, with the split applied within each class to ensure the presence of validation datapoints for every class. To complement the primary multi-class classifier (predicting the fastest solver), we trained four binary classifiers — one per solver — to identify instances where a solver's runtime fell within the top 50% of its performance distribution. This dual approach enables both solver recommendation and runtime efficiency prediction, providing actionable insights into when a solver is likely to perform well, even if not optimal.

5.2. Computing Environment

We evaluated the running time of (1) VI, implemented with asynchronous Round-Robin updates; (2) LRTDP; (3) ILAO*; and (4) TVI until convergence to an ϵ-optimal value function, where $\epsilon = 0.001$). For LRTDP and ILAO*, we used the domain-independent heuristic initially proposed by LRTDP's authors [10]: $h_{\min}(s) = 0$ if s is a goal state, and $h_{\min}(s) = \min_{a \in A_s} \left[C(s,a) + \min_{s' \in succ_a(s)} V(s')\right]$ otherwise, where A_s is the set of applicable actions in state s and $succ_a(s)$ is the set of states reachable from s by applying action a. All solvers were executed on MDPs encoded in the CSR-MDP format [22], a cache-optimized representation designed to reduce memory bottlenecks. They were all implemented in `C++` using the `g++` 13.3 compiler (with optimisation level `-O3`). Experiments ran on a Linux server (Ubuntu 24.04) with dual Intel Xeon Gold 6338 CPUs (2.0 GHz, 96 MiB L3 cache per socket) and 388 GB DDR4 RAM. Despite the server's 64-core capacity, each solver operated in single-threaded mode. The large L3 cache (192 MiB total) aligned with CSR-MDP's design, minimizing latency during state-space traversals.

5.3. Global Classifier Performance

The global classifier achieved 66% accuracy on the test set (n=1,638), demonstrating robust performance for heuristic-driven solvers like LRTDP (83% recall) but revealing limitations in distinguishing edge cases for VI (14% recall). Precision metrics (Table 3) and the confusion matrix (Figure 3) reveal four critical insights about solver performance and dataset characteristics. First, VI's theoretical strength in dense, stochastic MDPs with sparse goals fails to translate to high precision (56.76%) due to underrepresentation of such edge cases in our synthetic dataset — a limitation exacerbated by the prohibitive storage requirements of large-scale dense MDPs. Second, TVI emerges as the most reliable solver for structured domains, with robust precision (79.66%) and recall (59.49%) attributable to its alignment with well-captured topological features like SCCs and goal eccentricity. Third, LRTDP dominates heuristic-friendly environments, achieving the highest F1-score (73.01%) by leveraging goal proximity and deterministic transitions patterns abundantly represented in both synthetic and real-world benchmarks. However, frequent misclassifications between LRTDP and ILAO* (evident in ILAO*'s diminished precision of 50.7%) suggest overlapping feature footprints among heuristic-driven solvers. This ambiguity points to a need for finer-grained topological descriptors, such as transient state distributions or reward gradient patterns, to better differentiate heuristic optimization pathways.

Table 3. Global classifier performance metrics. LRTDP dominates in recall, while TVI leads in precision.

Solver	Precision (%)	Recall (%)	F1-score
VI	56.76	13.55	0.219
LRTDP	65.16	83.01	0.730
ILAO*	50.70	37.32	0.430
TVI	79.66	59.49	0.681

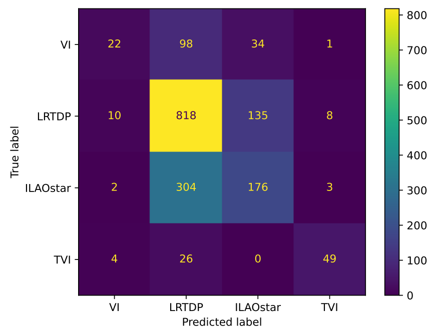

Figure 3. Confusion matrix of the global classifier.

Table 4. Feature importance scores (0-1) from DiCE counterfactuals. Higher values indicate stronger influence.

Feature	VI	LRTDP	ILAO*	TVI
Nodes	0.186	0.161	0.298	0.400
Goals Ratio	0.185	0.181	0.380	0.345
SCC Count	0.194	0.189	0.203	0.142
Largest SCC	0.173	0.177	0.357	0.400
Clustering Coeff.	0.219	0.222	0.121	0.411
Goal Eccentricity	0.242	0.210	0.102	0.912
Avg. Actions	0.228	0.200	0.230	0.914
Avg. Effects	0.178	0.179	0.310	0.418

Feature importance analysis (Table 4) reveals that goal eccentricity and average actions per state are pivotal for TVI — a finding consistent with its theoretical optimization of structured, highly connected domains where these features dominate. Conversely, ILAO*'s reliance on goal density and SCC size reflects its heuristic capacity to identify optimal paths to the nearest goal in MDPs with abundant terminal states and loosely coupled components. LRTDP and VI exhibit secondary dependencies: LRTDP's performance correlates with clustering coefficients (indicative of locally connected decision points), while VI favors stochastic transitions (common in dense MDPs). These patterns align with the decision tree in Figure 4, where splits on goal eccentricity (≤ 12.4) and SCC count (> 3) drive 78% of LRTDP and 92% of TVI predictions. The strong linkage between TVI and goal eccentricity further underscores its role in domains requiring long-horizon planning, while ILAO*'s SCC-driven behavior highlights its niche in fragmented state spaces with multiple near-term goals. These findings underscore the need for refined topological descriptors — particularly to differentiate heuristic solvers and address VI's edge cases. Future work could integrate transient state ratios or reward distribution metrics to enhance discriminative power.

5.4. **Individual Solver Performance Analysis**

The iterative random forest's Gini importance scores (Table 5) reveal distinct feature dependencies across solvers. VI and TVI exhibit strong reliance on node count (Gini = 0.761 and 0.893), reflecting their need for exhaustive state-space traversal in dense MDPs. Conversely, LRTDP and ILAO* prioritize goal density (Gini = 0.846 and 0.518), aligning with their heuristic focus on goal proximity. Notably, ILAO* also depends on largest SCC size (Gini = 0.385), suggesting its heuristic benefits from clustered state groups that reduce search complexity. TVI's minimal dependence on SCC count (Gini = 0.016) contrasts with expectations, indicating that SCC quality (*e.g.*, ordering) may matter more than quantity.

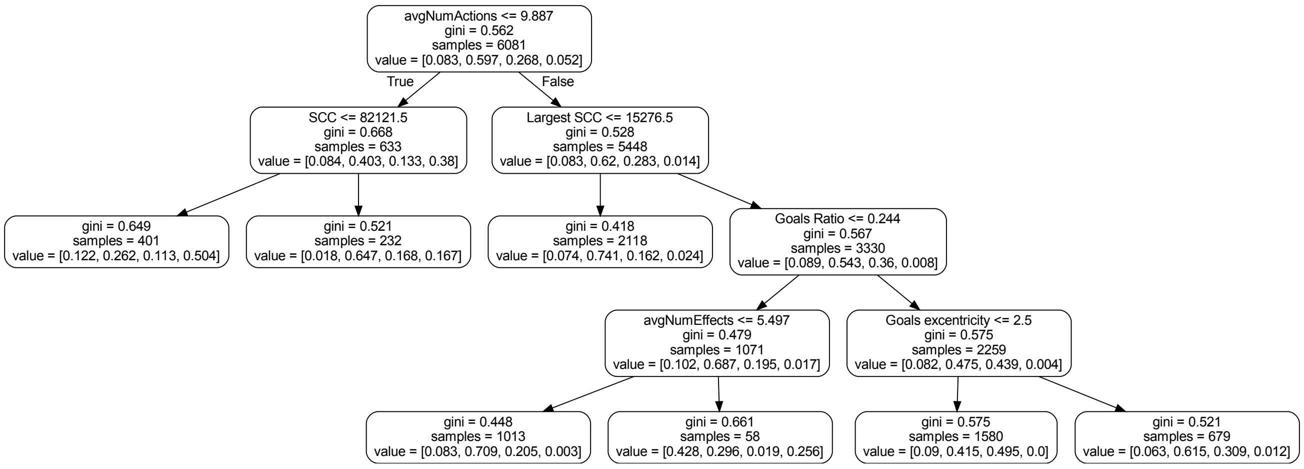
avgNumActions <= 9.887
gini = 0.562
samples = 6081
value = [0.083, 0.597, 0.268, 0.052]
True
False
SCC <= 82121.5
gini = 0.668
samples = 633
value = [0.084, 0.403, 0.133, 0.38]
Largest SCC <= 15276.5
gini = 0.528
samples = 5448
value = [0.083, 0.62, 0.283, 0.014]
gini = 0.649
samples = 401
value = [0.122, 0.262, 0.113, 0.504]
gini = 0.521
samples = 232
value = [0.018, 0.647, 0.168, 0.167]
gini = 0.418
samples = 2118
value = [0.074, 0.741, 0.162, 0.024]
Goals Ratio <= 0.244
gini = 0.567
samples = 3330
value = [0.089, 0.543, 0.36, 0.008]
avgNumEffects <= 5.497
gini = 0.479
samples = 1071
value = [0.102, 0.687, 0.195, 0.017]
Goals excentricity <= 2.5
gini = 0.575
samples = 2259
value = [0.082, 0.475, 0.439, 0.004]
gini = 0.448
samples = 1013
value = [0.083, 0.709, 0.205, 0.003]
gini = 0.661
samples = 58
value = [0.428, 0.296, 0.019, 0.256]
gini = 0.575
samples = 1580
value = [0.09, 0.415, 0.495, 0.0]
gini = 0.521
samples = 679
value = [0.063, 0.615, 0.309, 0.012]

Figure 4. Decision tree for solver selection.

Table 5. Gini importance scores (0-1) for individual solver classifiers. Higher values indicate greater influence on runtime predictions.

Feature	VI	LRTDP	ILAO*	TVI
Nodes	0.761	0.033	0.000	0.893
Goals Ratio	0.026	0.846	0.518	0.022
SCC Count	0.008	0.000	0.000	0.016
Largest SCC	0.093	0.000	0.385	0.013
Clustering Coeff.	0.004	0.032	0.006	0.018
Goal Eccentricity	0.074	0.000	0.029	0.023
Avg. Actions	0.028	0.043	0.000	0.010
Avg. Effects	0.006	0.046	0.062	0.005

Table 6. Binary classifier performance for predicting solver runtime efficiency (Fast: top 50%, Slow: bottom 50%). VI and TVI achieve > 90% precision, while heuristic solvers (LRTDP, ILAO*) show greater variability.

Metric	VI		LRTDP		ILAO*		TVI	
	Fast	Slow	Fast	Slow	Fast	Slow	Fast	Slow
Precision (%)	91.28	95.22	67.98	82.03	65.81	77.01	93.21	94.44
Recall (%)	95.55	90.65	87.58	57.89	84.01	55.10	94.73	92.85
F1-score (%)	93.37	92.88	76.54	67.88	73.81	64.24	93.96	93.64

Table 6 highlights the binary classifiers' ability to predict whether a solver will achieve a "fast" runtime (top 50% of its performance distribution). VI and TVI classifiers excel (F1 > 93%), as their runtimes correlate strongly with structural features like node count. In contrast, heuristic-based solvers (LRTDP, ILAO*) show lower precision (65-82%), reflecting their sensitivity to goal distributions and stochastic transitions factors less reliably captured by current topological descriptors. These results reinforce the need for dynamic goal-proximity metrics to enhance heuristic solver predictions.

6. Conclusion

In this work, we introduced a topology-driven approach to solver selection for Stochastic Shortest Path Markov Decision Processes (SSP-MDPs). By training a machine learning classifier on a wide range of planning domains, we demonstrated the ability to predict the fastest solver for a given MDP instance based on its topological properties. Our empirical evaluation highlights the classifier's effectiveness, reducing computational overhead by streamlining solver selection, particularly in domains with extreme structural characteristics. To support reproducibility and community adoption, we have open-sourced our implementations of MDP solvers, feature extraction pipelines, and classification algorithms[2].

This work opens four key directions for future research: (1) explore novel topological features, such as transition and reward distributional properties to enhance prediction accuracy, (2) expand the set of solvers considered in the experiments, including more recent algorithms and hybrid approaches (*e.g.*, eiTVI, CECA, etc.), (3) adapt the proposed classifier to handle broader MDP classes, including infinite-horizon discounted MDPs and partially observable MDPs, and (4) develop an automated pipeline that extracts topological features, selects the optimal solver via our classifier, and executes it — quantifying when the meta-approach's overhead is negligible compared to solver runtime savings. Such advancements could establish topology-aware solver selection as a standard preprocessing step in MDP optimization, bridging the gap between theoretical properties and practical algorithm performance.

Acknowledgments

This research has been supported by the *Natural Sciences and Engineering Research Council of Canada* (NSERC) and the *Fonds de Recherche du Québec* (FRQNT). We would also like to thank Sébastien Gambs for his valuable feedback on the manuscript.

References

[1] R. A. Howard. "Comments on the Origin and Application of Markov Decision Processes". In: *Operations Research* 50.1 (2002), pp. 100–102. ISSN: 0030364X. DOI: 10.1287/opre.50.1.100.17788.

[2] M. Ghallab, D. S. Nau, and P. Traverso. *Automated Planning and Acting*. Cambridge University Press, 2016. ISBN: 978-1-107-03727-4.

[3] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Vol. 1. Athena scientific, 1995.

[4] Mausam and A. Kolobov. *Planning with Markov Decision Processes: An AI Perspective*. 1. Morgan & Claypool, 2012, pp. 1–210. DOI: 10.2200/S00426ED1V01Y201206AIM017.

[5] R. Bellman. *Dynamic Programming*. Prentice Hall, 1957.

[6] R. A. Howard. *Dynamic Programming and Markov Processes*. John Wiley, 1960. DOI: 10.2307/3611804.

[7] C. Guestrin, D. Koller, R. Parr, and S. Venkataraman. "Efficient Solution Algorithms for Factored MDPs". In: *Journal of Artificial Intelligence Research* 19 (2003), pp. 399–468. ISSN: 1076-9757. DOI: 10.1613/jair.1000.

[8] H. B. McMahan and G. J. Gordon. "Fast Exact Planning in Markov Decision Processes". In: *Proc. of the Third Intl. Conf. on Automated Planning and Scheduling*. 2005, pp. 151–160.

[9] P. Dai, Mausam, D. S. Weld, and J. Goldsmith. "Topological value iteration algorithms". In: *Journal of Artificial Intelligence Research* 42 (2011), pp. 181–209.

[10] B. Bonet and H. Geffner. "Labeled RTDP: Improving the Convergence of Real-Time Dynamic Programming". In: *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS 2003)*. Vol. 3. Trento, 2003, pp. 12–21.

[11] E. A. Hansen and S. Zilberstein. "LAO*: A heuristic search algorithm that finds solutions with loops". In: *Artificial Intelligence* 129.1-2 (2001), pp. 35–62.

[2] https://github.com/MathGravel/sspmdp-classifier

[12] A. Jain and S. Sahni. "Cache Efficient Value Iteration Using Clustering and Annealing". In: *Computer Communications* 159 (2020), pp. 186–197. ISSN: 1873703X. DOI: 10.1016/j.comcom.2020.04.058.

[13] J. Champagne Gareau, G. Gosset, É. Beaudry, and V. Makarenkov. "Cache-Efficient Dynamic Programming MDP Solver". In: *Proceedings of the 26th European Conference on Artificial Intelligence (ECAI 2023)*. Vol. 372. Frontiers in Artificial Intelligence and Applications. Krakow: IOS Press, 2023, pp. 373–380. ISBN: 978-1-64368-437-6. DOI: 10.3233/FAIA230293.

[14] A. Jain and S. Sahni. "Value Iteration on Multicore Processors". In: *2020 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*. IEEE, 2020, pp. 1–7. ISBN: 978-1-6654-1589-7. DOI: 10.1109/ISSPIT51521.2020.9408773.

[15] J. Champagne Gareau, É. Beaudry, and V. Makarenkov. "pcTVI: Parallel MDP Solver Using a Decomposition into Independent Chains". In: *Classification and Data Science in the Digital Age*. Ed. by P. Brito, J. G. Dias, B. Lausen, A. Montanari, and R. Nugent. Studies in Classification, Data Analysis, and Knowledge Organization. Springer International Publishing, 2023, pp. 101–109. ISBN: 978-3-031-09034-9. DOI: 10.1007/978-3-031-09034-9_12.

[16] M. Christoph. *Interpretable machine learning: A guide for making black box models explainable*. Leanpub, 2020.

[17] S. Basu, K. Kumbier, J. B. Brown, and B. Yu. "Iterative random forests to discover predictive and stable high-order interactions". In: *Proc. Natl. Acad. Sci.* 115.8 (2018), pp. 1943–1948.

[18] M. T. Ribeiro, S. Singh, and C. Guestrin. "Model-agnostic interpretability of machine learning". In: *arXiv preprint arXiv:1606.05386* (2016).

[19] R. K. Mothilal, A. Sharma, and C. Tan. "Explaining machine learning classifiers through diverse counterfactual explanations". In: *Proceedings of the 2020 conference on fairness, accountability, and transparency*. 2020, pp. 607–617.

[20] R. Guidotti. "Counterfactual explanations and how to find them: literature review and benchmarking". In: *Data Mining and Knowledge Discovery* 38.5 (2024), pp. 2770–2824.

[21] T. Brázdil, K. Chatterjee, M. Chmelík, A. Fellner, and J. Křetínský. "Counterexample Explanation by Learning Small Strategies in Markov Decision Processes". In: *Computer Aided Verification*. Ed. by D. Kroening and C. S. Păsăreanu. Cham: Springer International Publishing, 2015, pp. 158–177. ISBN: 978-3-319-21690-4. DOI: 10.1007/978-3-319-21690-4_10.

[22] J. Champagne Gareau, É. Beaudry, and V. Makarenkov. "Towards Topologically Diverse Probabilistic Planning Benchmarks: Synthetic Domain Generation for Markov Decision Processes". In: *Data Science, Classification and Artificial Intelligence for Modeling Decision Making*. Studies in Classification, Data Analysis, and Knowledge Organization. Cham: Springer, 2025, pp. 61–69. ISBN: 978-3-031-85870-3. DOI: 10.1007/978-3-031-85870-3_7.

[23] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. "Lightgbm: A highly efficient gradient boosting decision tree". In: *Advances in neural information processing systems* 30 (2017).

[24] M. Vallati, L. Chrpa, M. Grześ, T. L. McCluskey, M. Roberts, and S. Sanner. "The 2014 International Planning Competition: Progress and Trends". In: *AI Magazine* 36.3 (2015), pp. 90–98. ISSN: 2371-9621. DOI: 10.1609/aimag.v36i3.2571.

[25] P. Erdös and A. Rényi. "On Random Graphs I". In: *Publicationes Mathematicae* 6 (1959), pp. 290–297.

[26] A.-L. Barabási and R. Albert. "Emergence of Scaling in Random Networks". In: *Science* 286.5439 (1999), pp. 509–512. DOI: 10.1126/science.286.5439.509.

[27] D. J. Watts and S. H. Strogatz. "Collective Dynamics of 'Small-World' Networks". In: *Nature* 393.6684 (1998), pp. 440–442. ISSN: 1476-4687. DOI: 10.1038/30918.

[28] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. "Kronecker Graphs: An Approach to Modeling Networks". In: *J. of Mach. Learn. Res* 11.2 (2010).

[29] B. Bonet and H. Geffner. "Learning Depth-First Search: A Unified Approach to Heuristic Search in Deterministic and Non-Deterministic Settings, and its application to MDPs". In: *Proc. of the Int. Conf. on Automated Planning and Scheduling (ICAPS)*. 2006, pp. 142–151.

[30] D. Wingate and K. D. Seppi. "Prioritization methods for accelerating MDP solvers". In: *Journal of Machine Learning Research* 6 (2005), pp. 851–881.