PDDLFuse: A Tool for Generating Diverse Planning Domains

Anonymous submission

Abstract

Real-world challenges require planning algorithms that can adapt to diverse and dynamic domains. Traditionally, handcrafted planning domains have been limited in scale and diversity. While advancements in generative AI, including large language models (LLMs), have automated domain creation, they primarily reconstruct existing domains from natural language descriptions rather than generating novel ones. In contrast, domain randomization—an effective technique in reinforcement learning—demonstrates enhanced performance and generalizability by training on a wide variety of randomized domains. Inspired by this, we present PDDLFuse, a tool designed to bring the benefits of domain randomization to the Planning Domain Definition Language (PDDL). PDDLFuse generates diverse and complex planning domains, enabling robust testing and validation of new planners and planning foundation models. Its configurable parameters enable precise control over domain complexity. Initial evaluations reveal that PDDLFuse generates complex domains, as evident by the limited solvability of domain-independent planners and high expressivity measured by the k-WL test on generated domains. Furthermore, foundation models trained on PDDLFuse-generated domains demonstrate significantly improved generalization compared to those trained on traditional existing domains.

Introduction

Automated planning systems are critical for a wide range of applications, including robotics and software management (Ghallab, Nau, and Traverso 2004). Traditionally, planning domains have been manually created, which, while capable of producing complex domains, inherently limits the rate of domain generation and the diversity of available domains. This limitation slows the development and testing of planning algorithms, reducing their robustness and adaptability to new or unseen domains (Chen, Thiébaux, and Trevizan 2024).

Recent advancements in generative AI, particularly large language models (LLMs), have automated planning domain creation. However, these approaches primarily focus on reconstructing existing domains from natural language descriptions, resulting in a limited set of domains for training and evaluation (Oswald et al. 2024). As a result, the potential for improving generalization remains constrained, as the models are still being trained on a narrow set of existing domains. In contrast, generating diverse domains introduces variability that enhances generalization, as shown in existing works in reinforcement learning (Mehta et al. 2020; Ajani, Hur, and Mallipeddi 2023).

Inspired by the success of domain randomization in reinforcement learning, which improves performance and adaptability by training on diverse randomized domains, we propose applying similar principles to the Planning Domain Definition Language (PDDL). Domain randomization demonstrates that diverse training environments significantly enhance an agent's ability to generalize to new, unseen domains, a quality essential for robust planning systems.

PDDLFuse is a tool designed to generate novel planning domains by fusing existing ones rather than merely reconstructing them from natural language descriptions (Oswald et al. 2024; Mahdavi et al. 2024). This approach substantially increases the diversity of domains available for planning research, enabling the development of more adaptable and generalizable planning algorithms. By expanding the range of test domains, PDDLFuse supports comprehensive testing of planning algorithms, validates new planners, and allows rigorous evaluation of planning foundation models on these diverse domains, addressing previously unexplored aspects of planning research. Preliminary evaluations indicate that PDDLFuse generates complex and diverse domains, as evidenced by the limited solvability of domain-independent planners and the higher expressivity observed through the k-WL test. Moreover, foundation models trained on PDDLFuse-generated domains demonstrate superior generalization compared to those trained on traditional datasets.

The following sections cover the background of planning domains, review related works, describe the methodologies employed, and present the experimental results.

Background and Related Works

This section covers the essentials of planning domains, domain-independent planners, and the role of generative AI in domain reconstruction. We highlight the limitations of current methods. We also explore domain randomization, an approach from reinforcement learning that improves algorithm robustness by training with diverse domains, offering potential benefits for planning systems. Detailed discussion in *Supplementary Material*.

Planning Domains and Problems

In the context of automated planning, a *planning domain* is a structured description of an environment consisting of objects, predicates, and actions that an agent can perform. Formally, a planning domain $\mathcal{D}$ is defined by a tuple $(\mathcal{O}, \mathcal{P}, \mathcal{A})$, where:

- $\mathcal{O}$ is a finite set of *objects* that exist within the domain.
- $\mathcal{P}$ is a finite set of *predicates*, where each predicate represents a property or relationship among objects (e.g., `At(location, package)`).
- $\mathcal{A}$ is a finite set of *actions*, where each action $a \in \mathcal{A}$ is defined by a pair $(\text{pre}(a), \text{eff}(a))$:
  - pre(a), the *preconditions* of action a, is a set of predicates that must hold true for a to be executed.
  - eff(a), the *effects* of action a, is a set of predicates that describe the changes in the domain after a is executed.

A *planning problem* specifies a particular task within a domain by defining both an initial and a goal state. Formally, a planning problem $\mathcal{P}$ is defined by the tuple $(\mathcal{D}, s_0, s_g)$, where:

- $\mathcal{D}$ is the domain in which the problem is defined.
- s_0 is the *initial state*, a set of grounded predicates representing the domain file's state before planning begins.
- s_g is the *goal state*, a set of grounded predicates specifying the desired conditions that define the successful completion of the task.

Domain Reconstruction

Automating domain creation through translation from natural language has seen progress, though it still faces limitations in fostering diversity and novelty. Oswald et al. (2024) employed LLMs to replicate planning domains from textual descriptions, closely aligning with existing PDDL specifications, but requiring a reference domain for validation restricts its scope to known domains. Similarly, Mahdavi et al. (2024) used an iterative refinement approach with environment feedback to enhance LLM-generated domains, reducing manual effort but focusing primarily on refining rather than creating new domains, limiting scalability. The "Translate-Infer-Compile" (TIC) tool by Agarwal and Sreepathy (2024) and "AUTOPLANBENCH" by Steina et al. (2024) advance the translation of natural language into structured PDDL, enhancing accuracy through logic reasoning and LLM interaction, yet they remain confined to reconstructing established domains, rather than diversifying the domain pool essential for generalization in planning.

Generalization in Planning

Generalization in automated planning is limited by the lack of diverse training domains, as exemplified by those in the International Planning Competition (IPC). This leads to weaker inductive biases and models prone to overfitting. Recent approaches using Graph Neural Networks (GNNs) and Large Language Models (LLMs) aim to address this but remain restricted by their narrow domain scope. For instance, the graph representations by Chen, Thiébaux, and Trevizan (2024) and the GOOSE tool by Chen, Thiébaux, and Trevizan (2023) show promise in learning domain-independent heuristics, yet face scalability issues and reliance on IPC domains. Similarly, Toyer et al. (2018) employs GNNs to improve plan quality but depends on accessible solvers, limiting real-world applicability. LLMs, even with advanced prompting techniques (Hu et al. 2023; Yao et al. 2023), are constrained by a lack of domain diversity, reducing their effectiveness in novel contexts. Multimodal and code-based models, such as those by Lu et al. (2023); Pallagani et al. (2023); Khandelwal, Sheth, and Agostinelli (2024); Agostinelli, Panta, and Khandelwal (2024), perform well in known distributions but struggle with out-of-distribution domains, highlighting the need for more diverse domains to achieve robust generalization. It is worth noting that most recent works on planning foundation models have relied heavily on GNNs (Chen, Thiébaux, and Trevizan 2023, 2024; Toyer et al. 2018).

Weisfeiler-Leman Test Implementation

Building on the importance of generalization in planning and the reliance on Graph Neural Networks (GNNs) for structured data representation, the Weisfeiler-Leman (WL) test is a critical tool for evaluating the expressivity of domains. The WL test is a widely used algorithm for graph isomorphism testing, which iteratively refines node colorings (labels) based on the colors of neighboring nodes. The k-WL test extends this concept by considering k-tuples of nodes, enabling more powerful discrimination between non-isomorphic graphs. It has been established that the expressivity of standard GNNs is upper-bounded by the 1-WL test (Huang and Villar 2021), meaning that GNNs cannot distinguish between graphs that the 1-WL test considers identical.

Description of the Algorithm:

- **Initialization:** Each k-tuple of nodes are assigned an initial color based on isomorphism-invariant characteristics, such as node degrees or specific substructures within the tuple.
- **Iterative Refinement:** At each iteration, the color of each k-tuple is refined by considering the colors of its neighboring k-tuples. Neighbors include k-tuples that can be formed by substituting any node in the tuple with another node in the graph.
- **Stabilization:** The process continues until no new colors emerge between successive iterations, indicating that the coloring has stabilized.
- **Result Compilation:** The multiset of final colors is computed, serving as a signature of the graph's structure that captures information up to k-th order dependencies.

Domain Randomization and Generalization in Reinforcement Learning

Domain randomization is a key technique in reinforcement learning (RL) that improves robustness and generalizability

Algorithm 1: Weisfeiler-Leman Test Implementation

```
Require: Graph $G = (V, E)$, integer $k$, maximum iterations $T$
 1: Initialize color function $c^{(0)}$ for all $k$-tuples in $V^k$ using initial graph features
 2: for $t = 1$ to $T$ do
 3:     Initialize $newColors$ as an empty map
 4:     for all $k$-tuples $\mathbf{v} = (v_1, v_2, \ldots, v_k) \in V^k$ do
 5:         Construct multiset $M_{\mathbf{v}}$ of colors of neighboring $k$-tuples
 6:         $newColors[\mathbf{v}] \leftarrow \text{Hash}\big(c^{(t-1)}(\mathbf{v}), M_{\mathbf{v}}\big)$
 7:     end for
 8:     $changes \leftarrow$ number of $k$-tuples where $newColors[\mathbf{v}] \neq c^{(t-1)}[\mathbf{v}]$
 9:     Update colors: $c^{(t)} \leftarrow newColors$
10:     if $changes = 0$ then
11:         break
12:     end if
13: end for
14: return Multiset $\{c^{(T)}(\mathbf{v}) \mid \mathbf{v} \in V^k\}$
```

by exposing agents to diverse training domains, thereby improving generalization to unfamiliar domains. Mehta et al. (2020) introduced Active Domain Randomization (ADR), which strategically manipulates challenging environmental parameters to improve policy robustness, particularly in robotic control. Similarly, Ajani, Hur, and Mallipeddi (2023) showed that varying physical properties like surface friction can significantly boost an RL agent's generalization ability. Kang, Chang, and Choi (2024) further refined this with Balanced Domain Randomization (BDR), which focuses training on rare and complex domains to enhance performance under demanding conditions. In non-physical tasks, Koo, Yu, and Lee (2019) applied adversarial domain adaptation to align feature representations across domains, enhancing policy generalization in complex tasks like dialogue systems. These studies demonstrate the power of domain randomization in building resilient AI, aligning with PDDLFuse's goal to generate diverse planning domains to improve generalization in automated planning.

Domain Independent Planners

Domain-independent planners such as Fast Downward (FD)(Helmert 2006) and LPG (Gerevini, Serina et al. 2002) play a crucial role in the advancement of automated planning technologies. These systems are designed to function across a wide range of problem domains by utilizing heuristics that do not rely on specific domain knowledge. FD converts PDDL tasks into a more manageable internal format and employs powerful heuristics like the Fast-Forward (FF) (Hoffmann and Nebel 2001), which simplifies planning by focusing only on positive action effects, and the landmark-cut (lmcut) (Helmert and Domshlak 2009), which identifies essential milestones within a plan to optimize the search process. Conversely, LPG leverages stochastic local search strategies that incrementally refine plans through action-graph and plan-graph techniques, proving highly effective in both propositional and numerical planning contexts (Gerevini, Serina et al. 2002). The use of these planners in research is driven by their ability to efficiently generate solutions in diverse domains, thereby facilitating the development of more robust and adaptable planning systems.

Other Foundation Models

One of the existing foundation models is GOOSE, which represents a significant stride toward domain generalization in planning Chen, Thiébaux, and Trevizan (2023). GOOSE employs the STRIPS Learning Graph (SLG) to represent the planning domains (Chen, Thiébaux, and Trevizan 2024). This model encodes states, actions, and their relationships into a graph structure, facilitating the capture of domain knowledge through representations of preconditions and effects and deleting effects as graph edges.

Approximate Value Iteration

Approximate Value Iteration (AVI) employs a Deep Neural Network (DNN) with parameters θ, approximating the value function through iterative heuristic updates. The central heuristic function h of our planning algorithm is refined iteratively using the update rule:

$$h'(s) = \min_{a \in \mathcal{A}} \left(c(s, a) + h(T(s, a))\right) \tag{1}$$

Here, $c(s, a)$ represents the cost associated with taking action a from state s, and $T(s, a)$ denotes the state transition resulting from action a. The primary objective of this model is to enhance the accuracy of cost-to-go estimates by minimizing the following loss function:

$$L(\theta) = \left(\min_{a \in \mathcal{A}} \left(c(s, a) + h_{\theta^-}(T(s, a))\right) - h_\theta(s)\right)^2 \tag{2}$$

In this model, $h_\theta(s)$ serves as the heuristic estimate of the cost-to-go, while θ^- refers to the parameters of a target network that are periodically synchronized with θ to stabilize the training process, providing a consistent target within the dynamic learning environment.

Batch Weighted A* Search

Batch Weighted A* Search (BWAS) is an enhanced variant of the traditional A* search algorithm, tailored for complex pathfinding and graph traversal tasks with vast state spaces or intricate dynamics (Agostinelli et al. 2019). A* search traditionally operates by using a priority queue where each node's priority is determined by the sum of the actual path cost from the start node to the current node $g(x)$, and an estimated cost from the current node to the goal provided by a heuristic function $h(x)$. The function $f(x) = g(x) + h(x)$ governs this process.

BWAS modifies this by introducing a weighting factor λ and batch processing of nodes. The function becomes:

$$f(x) = \lambda g(x) + h(x) \tag{3}$$

where λ adjusts the emphasis between the heuristic guidance and the actual path cost. A lower λ value accelerates the search but may compromise the path optimality.

Additionally, BWAS employs batch processing, expanding multiple nodes simultaneously instead of one at a time. This method is especially beneficial on parallel computing architectures like GPUs, where the heuristic calculations can be performed concurrently for multiple nodes. By tuning λ and the batch size N, BWAS efficiently balances exploration and exploitation, enhancing its suitability for solving large-scale or computationally demanding problems.

Methods

This section details the procedures and algorithms developed to fuse existing domains and manipulate domain characteristics to generate new and diverse domains, as shown in Algorithm 2. Additional details are provided in the *Supplementary Material.*

Domain Generation

In PDDLFuse, the generation of new planning domains $\mathcal{D} = (\mathcal{O}, \mathcal{P}, \mathcal{A})$ begins by selecting two existing domains and their corresponding problem files as bases. To ensure uniqueness, an initial step systematically renames predicates and action names to avoid overlap between the two domains.

The actions within the domains are then enhanced using a set of hyperparameters that control modifications to preconditions and effects. These parameters include:

- Probability of adding a new predicate to the preconditions (prob_add_pre).
- Probability of adding a new predicate to the effects (prob_add_eff).
- Probability of removing a predicate from the preconditions (prob_rem_pre).
- Probability of removing a predicate from the effects (prob_rem_eff).

Additional parameters include:

- prob_neg: Probability of negating a predicate when adding it to the preconditions or effects.
- rev_flag: Ensures predicate reversibility.
- num_objs: Controls the number of objects, providing further flexibility in domain generation.

This process enables the creation of complex and diverse planning domains along with their respective problem files, facilitating the testing and development of more generalizable planning algorithms.

Problem File Generation

Problem file generation starts with setting the initial state based on num_objs. A sequence of random actions from the generated domain is executed to transition the initial state into a new state, where a subset of true predicates forms the goal state. This process ensures the generated problems are solvable within the domain.

This section outlines the systematic approach employed by PDDLFuse to generate new and diverse planning domains, with an emphasis on parametric controls that enable customization. Subsequent sections will discuss the experimental setup and results for evaluating the effectiveness and utility of these generated domains in planning research.

Algorithm 2: Domain & Problem Generation

```
Require: Two base domains $\mathcal{D}_1 = (\mathcal{O}_1, \mathcal{P}_1, \mathcal{A}_1)$ and $\mathcal{D}_2 = (\mathcal{O}_2, \mathcal{P}_2, \mathcal{A}_2)$
Ensure: No overlapping predicates or actions between $\mathcal{D}_1$ and $\mathcal{D}_2$
 1: $\mathcal{O} \leftarrow \mathcal{O}_1 \cup \mathcal{O}_2$ {Union of objects from both domains}
 2: $\mathcal{P} \leftarrow \mathcal{P}_1 \cup \mathcal{P}_2$ {Union of predicates from both domains}
 3: $\mathcal{A} \leftarrow \emptyset$
 4: for each action $a$ in $\mathcal{A}_1 \cup \mathcal{A}_2$ do
 5:     Define pre($a$) and eff($a$) for new $\mathcal{A}$
 6:     if random() $<$ prob_add_pre then
 7:         Add new predicates to pre($a$) {Expanding precond}
 8:     end if
 9:     if random() $<$ prob_add_eff then
10:         Add new effects to eff($a$) {Expanding effects}
11:     end if
12:     if random() $<$ prob_rem_pre then
13:         Remove predicates from pre($a$) {Simplifying precond}
14:     end if
15:     if random() $<$ prob_rem_eff then
16:         Remove predicates from eff($a$) {Simplifying effects}
17:     end if
18:     Apply prob_neg to negate added predicates
19:     $\mathcal{A} \leftarrow \mathcal{A} \cup \{a\}$ {Incorporating modified action into new domain}
20: end for
21: Generate problem $\mathcal{P}$ using modified $\mathcal{D}_{new} = (\mathcal{O}, \mathcal{P}, \mathcal{A})$ and num_objs
22: Execute actions to derive the goal state from the initial state
23: return $\mathcal{D}_{new}, \mathcal{P}$ {Output new planning domain and problem}
```

Experimental Setup and Results

This section details the experimental setup and results for evaluating PDDLFuse's ability to generate diverse and complex planning domains. The experiments assess the solvability of these domains using both domain-independent planners and foundation models.

Planners used

We evaluate the performance of traditional domain-independent planners, including Fast Downward (FD) with FF and lmcut heuristics and LPG. Additionally, we analyze the generalizability of foundation models by comparing GOOSE (Chen, Thiébaux, and Trevizan 2023) and an AVI-based reinforcement learning (RL) model trained on PDDLFuse-generated domains.

For GOOSE, we train on plans generated by the Scorpion planner using SLG graph representation over 50 existing planning domains comprising a dataset of 50,000 examples. The training employs a Message Passing Neural Network (MPNN) with 16 message-passing layers, mean aggregation, a hidden dimension of 64, and optimization using a Mean Squared Error (MSE) loss function. The Adam optimizer is used with a batch size 16 and an initial learning rate of 0.001. In contrast, the AVI-based RL model is trained on domains generated by PDDLFuse, utilizing the same SLG graph representation and MPNN parameters as GOOSE. This model undergoes training with a batch size of 100 over 600,000 iterations.

All experiments impose a time constraint of 200 seconds per problem instance. Batch-Weighted A* Search is used with foundation models as a heuristic to solve the problems.

Experiment 1: Solvability Across Depths

For these experiments, we varied the generated domains using the following parameters:

- prob_add_pre = 0.5
- prob_add_eff = 0.5
- prob_rem_pre = 0.3
- prob_rem_eff = 0.5

We generated ten domains per depth level, except for level 0, where we considered five problems each from the Gripper and Blocks World domains. For depth level 1, domains were generated using Gripper and Blocks World as base domains. Subsequent levels iteratively used previously generated domains as the base, progressively increasing complexity. Planner performance was assessed by success rates and path costs to evaluate their efficiency and adaptability.

Table 1: Solvability Across Depth Levels

Depth	**FD(FF)**	**LPG**	**GOOSE**	**AVI-based**
0	9/10	10/10	9/10	10/10
1	10/10	10/10	10/10	9/10
2	10/10	8/10	8/10	9/10
3	9/10	7/10	5/10	8/10
4	8/10	5/10	3/10	8/10
5	8/10	4/10	2/10	7/10

As shown in Table 1, FD(FF) and LPG exhibit high solvability at lower depths but decline as domain complexity increases. LPG's performance drops sharply beyond Depth 1, indicating its limited adaptability to complex domains. FD(FF), while slightly affected, maintains relatively consistent performance across depths.

The GOOSE model starts strong but faces significant challenges with increasing complexity, struggling to generalize to higher depths. In contrast, the AVI-based model demonstrates better generalizability, maintaining high solvability across depths, particularly at higher complexities. This highlights the advantage of training on the diverse and challenging domains generated by PDDLFuse. While AVI-based models require more training time than GOOSE, they also benefit from exposure to more domains and problems, leading to better generalization.

Experiment 2: Solvability Across Parameter Variations

We conducted experiments using five base domains—Blocks-World, Gripper, Depot, Grid, and Satellite—focusing on depth level 1 with num_obj $= 15$. To evaluate planner performance under various configurations, the following parameters were systematically varied:

- `Prob_add_precond` and `Prob_remove_precond`: (0.3, 0.7), (0.5, 0.5), (0.7, 0.3)
- `Prob_add_effect` and `Prob_remove_effect`: (0.3, 0.7), (0.5, 0.5), (0.7, 0.3)

Additionally, we varied prob_neg with values of 0.3, 0.5, and 0.7, keeping rev_flag $=$ True throughout the experiments.

The heat maps in Figure 1 illustrate FD(FF) success rates across different parameter configurations. Each cell represents planner solvability for specific combinations of add/remove probabilities and negation values. The results reveal that moderate negation probabilities (e.g., prob_neg $= 0.5$) and balanced add/remove probabilities (e.g., 0.5, 0.5) generally achieve higher solvability. Conversely, extreme parameter values, such as high negation probabilities or unbalanced add/remove probabilities, lead to reduced success rates. These findings highlight the sensitivity of FD(FF) to parameter variations, emphasizing the need for carefully chosen configurations to balance domain complexity and solvability. The results also showcase how parameter tuning can be used to generate more complex domains, aiding in robust evaluation of domain-independent planners.

The heat maps in Figure 2 illustrate the GOOSE Foundation Model's solvability across different parameter configurations for domains with 15 objects, using the Batch Weighted A* search. The results show that moderate settings, especially with prob_add_precond $= 0.5$ and prob_add_eff $= 0.5$, generally achieve higher solvability, while more extreme configurations reduce success rates. These findings highlight the GOOSE model's sensitivity to parameter variations, demonstrating that balancing the parameters is crucial for maintaining solvability and generalization across diverse planning environments.

The heat maps in Figure 3 display the solvability rates of the AVI-based Foundation Model using the Batch Weighted A* search across varied parameter configurations for domains with 15 objects. Consistently high solvability across all settings, particularly at moderate negation levels, indicates the model's robustness. The results confirm the AVI-based model's generalizability, performing well even under complex conditions and varying parameter. This underscores the importance of training on PDDL-Fuse generated diverse and complex domains, enhancing the model's generalizability. More results in *Supplementary Material.*

Experiment 3: Expressivity

Building upon the solvability experiments, we now evaluate complexity of the generated domains using the Weisfeiler-Leman (WL) test. Domains were generated by systematically varying parameters, including `Prob_add_precond` and `Prob_remove_precond`, `Prob_add_effect` and `Prob_remove_effect`, with values of (0.3, 0.7), (0.5, 0.5), and (0.7, 0.3). Additionally, prob_neg was varied with values of 0.3, 0.5, and 0.7, while rev_flag was set to True. The number of objects was also varied, and domain generation was performed across 5 depth levels. Memory constraints influenced the number of problems generated at higher depths, resulting in 912 domains for Depth 1, 393 for Depth 2, 216 for Depth 3, 61 for Depth 4, and 12 for Depth 5. This variation in parameters ensures a diverse set of domains for analyzing expressivity trends and complexity across depths.

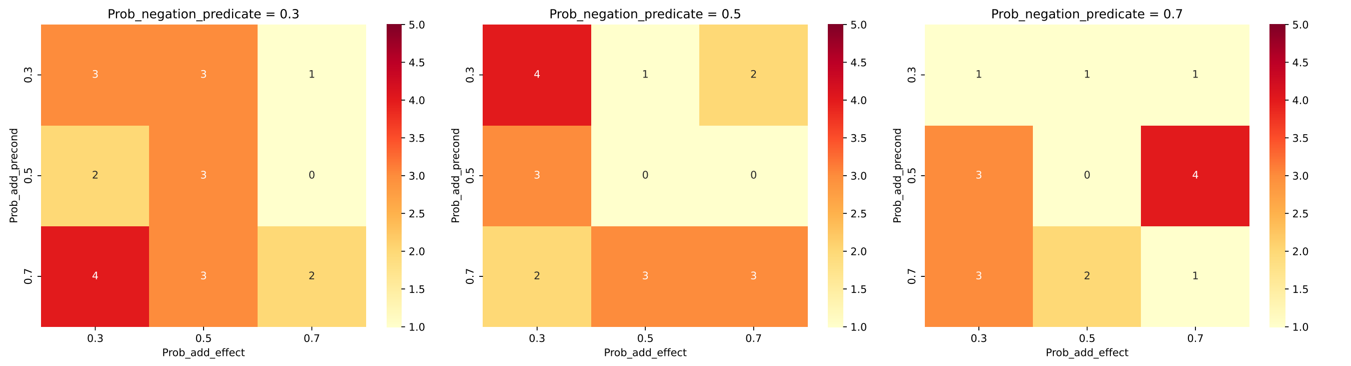

Figure 1: Solvability heat maps for domains with 15 objects, evaluated using the FD(FF) across varied parameter configurations.

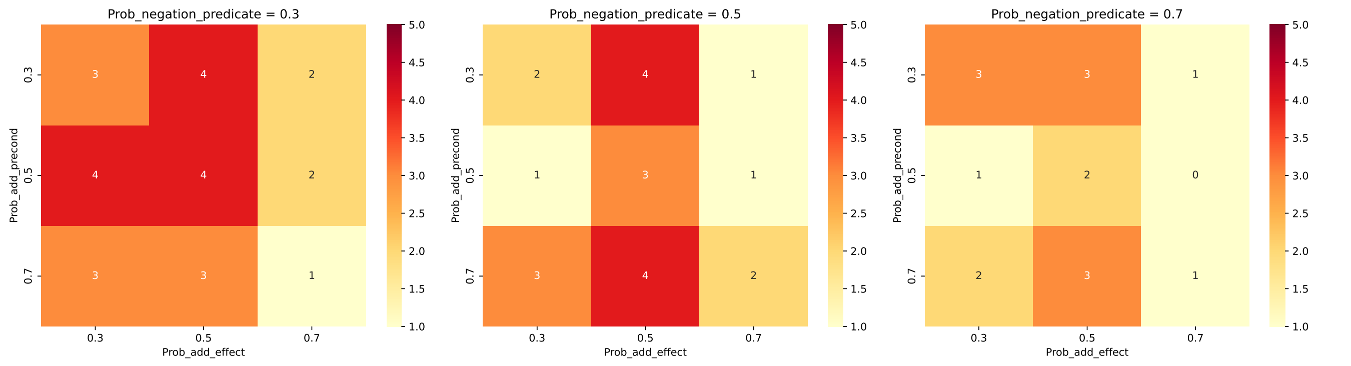

Figure 2: Solvability heat maps for domains with 15 objects, evaluated using the Batch Weighted A* search with GOOSE Foundation Model as heuristic across varied parameter configurations.

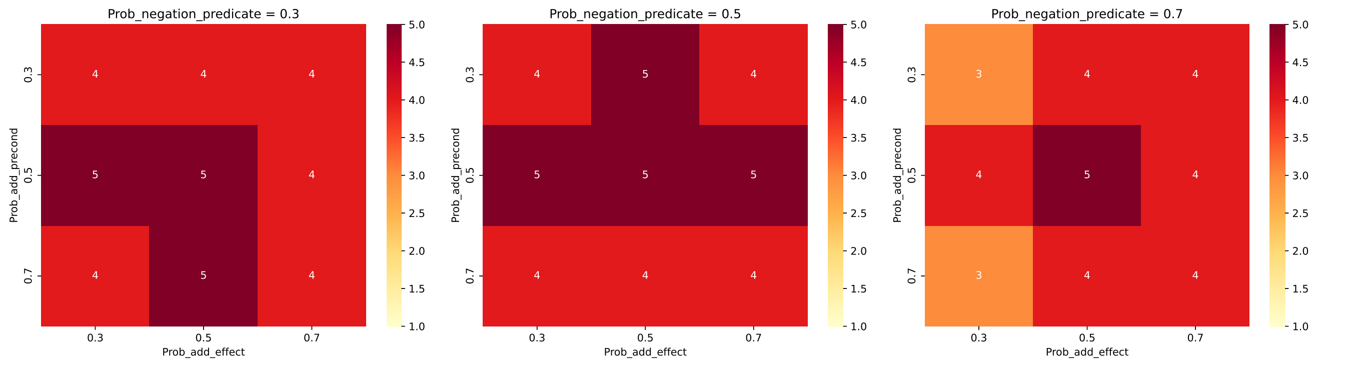

Figure 3: Solvability heat maps for domains with 15 objects, evaluated using the Batch Weighted A* search with AVI-based Foundation Model as heuristic across varied parameter configurations.

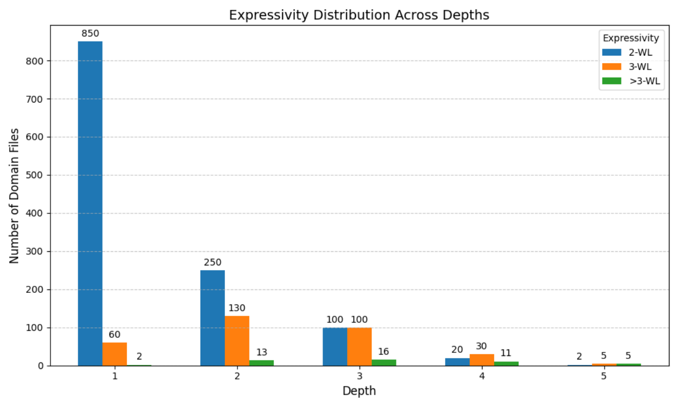

Figure 4: Distribution of planning domains distinguishable by the Weisfeiler-Leman (WL) test across various domain generation depths. As depth increases, the proportion of domains requiring higher-order WL tests ($k > 3$) rises significantly, underscoring a marked increase in structural complexity.

As shown in the Figure 4, we identify the following expressivity trend,

- At **Depth 1**, the majority of domains (850) are distinguishable by the 2-WL test, indicating simple structures. A smaller percentage (7%) requires the 3-WL test, with less than 1% needing higher-order WL tests.
- At **Depth 2**, the number of domains identifiable by the 2-WL test drops to 250, while 130 (34%) need the 3-WL test, and 13 (3.5%) require more complex tests.
- By **Depth 5**, only 2 domains are distinguishable by the 2-WL test, with a substantial increase in domains needing higher-order WL tests—30 (55%) for the 3-WL and 11 (20%) for tests beyond the 3-WL, highlighting significant increases in complexity. Domains in the '>3-WL' category reach the computational limits of the testing procedure, reflecting the challenges of testing higher complexity with available resources.

These observations underscore that as depth increases, the percentage of domains requiring higher-order WL tests increases dramatically. This trend signifies the generation of increasingly intricate domain structures, which serve as a rigorous benchmark for evaluating the generalizability and robustness of planning models.

System Hardware Configuraiton

Experiments were conducted on CPU-only nodes equipped with Intel Xeon Platinum 8480CL processors. These processors feature two sockets with 56 cores each, totaling 112 cores per node. Each node was allocated 500GB of RAM. For each experiment, 14 cores were utilized, providing approximately 608MB of RAM per experiment with a random seed of 42.

Conclusion and Future Work

In this work, we introduced PDDLFuse, a tool designed to generate diverse and complex planning domains that significantly challenge existing planning systems and provide a robust resource for developing and benchmarking planning foundation models. Through systematic parameter variation and depth-based complexity, PDDLFuse generated a wide variety of domains, increasing structural intricacy as evidenced by the Weisfeiler-Leman (WL) expressivity analysis. Our experiments demonstrated that these domains underlines the limitations of current domain-independent planners like Fast Downward and LPG but also highlight the varying generalizability of foundation models such as GOOSE and AVI-based models. The results underscore the importance of diverse and complex training environments in improving model generalizability and robustness.

Future work will focus on enhancing PDDLFuse's capabilities by integrating dynamic feedback mechanisms to adjust parameters based on planner performance. This would enable the generation of adaptive and goal-specific domains, further refining the complexity and diversity of the domains created. Additionally, we plan to explore the integration of knowledge graphs into the domain generation process. By leveraging knowledge graphs, domain generation can be influenced and controlled to align with specific structural or contextual requirements, enabling the creation of more targeted domain variations. Furthermore, knowledge graphs could be utilized to train foundation models that generalize to unseen variations of existing domains without requiring fine-tuning, thereby enhancing their adaptability to novel scenarios. Another avenue for future work involves expanding PDDLFuse to replicate existing domains by precisely tuning parameters to match specific characteristics. This capability would broaden its applicability for benchmarking against known domains while still fostering the exploration of novel variations. Finally, the development of interpretative tools to analyze domain structures and provide insights into their complexity would improve the usability of PDDLFuse, making it an essential tool for advancing AI planning systems.

References

Agarwal, S.; and Sreepathy, A. 2024. TIC: Translate-Infer-Compile for Accurate "Text to Plan" Using LLMs and Logical Representations. In *International Conference on Neural-Symbolic Learning and Reasoning*, 222–244. Springer.

Agostinelli, F.; McAleer, S.; Shmakov, A.; and Baldi, P. 2019. Solving the Rubik's cube with deep reinforcement learning and search. *Nature Machine Intelligence*, 1(8): 356–363.

Agostinelli, F.; Panta, R.; and Khandelwal, V. 2024. Specifying goals to deep neural networks with answer set programming. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 34, 2–10.

Ajani, O. S.; Hur, S.-h.; and Mallipeddi, R. 2023. Evaluating Domain Randomization in Deep Reinforcement Learning Locomotion Tasks. *Mathematics*, 11(23): 4744.

Chen, D. Z.; Thiébaux, S.; and Trevizan, F. 2023. GOOSE: Learning domain-independent heuristics. In *NeurIPS 2023 Workshop on Generalization in Planning*.

Chen, D. Z.; Thiébaux, S.; and Trevizan, F. 2024. Learning Domain-Independent Heuristics for Grounded and Lifted Planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 20078–20086.

Gerevini, A.; Serina, I.; et al. 2002. LPG: A Planner Based on Local Search for Planning Graphs with Action Costs. In *Aips*, volume 2, 281–290.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: theory and practice*. Elsevier.

Helmert, M. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26: 191–246.

Helmert, M.; and Domshlak, C. 2009. Landmarks, critical paths and abstractions: what's the difference anyway? In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 19, 162–169.

Hoffmann, J.; and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14: 253–302.

Hu, H.; Lu, H.; Zhang, H.; Lam, W.; and Zhang, Y. 2023. Chain-of-Symbol Prompting Elicits Planning in Large Langauge Models. *arXiv preprint arXiv:2305.10276*.

Huang, N. T.; and Villar, S. 2021. A short tutorial on the weisfeiler-lehman test and its variants. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 8533–8537. IEEE.

Kang, C.; Chang, W.; and Choi, J. 2024. Balanced Domain Randomization for Safe Reinforcement Learning. *Applied Sciences*, 14(21): 9710.

Khandelwal, V.; Sheth, A.; and Agostinelli, F. 2024. Towards Learning Foundation Models for Heuristic Functions to Solve Pathfinding Problems. *arXiv preprint arXiv:2406.02598*.

Koo, S.; Yu, H.; and Lee, G. G. 2019. Adversarial approach to domain adaptation for reinforcement learning on dialog systems. *Pattern Recognition Letters*, 128: 467–473.

Lu, Y.; Lu, P.; Chen, Z.; Zhu, W.; Wang, X. E.; and Wang, W. Y. 2023. Multimodal Procedural Planning via Dual Text-Image Prompting. *arXiv preprint arXiv:2305.01795*.

Mahdavi, S.; Aoki, R.; Tang, K.; and Cao, Y. 2024. Leveraging Environment Interaction for Automated PDDL Generation and Planning with Large Language Models. *arXiv preprint arXiv:2407.12979*.

Mehta, B.; Diaz, M.; Golemo, F.; Pal, C. J.; and Paull, L. 2020. Active domain randomization. In *Conference on Robot Learning*, 1162–1176. PMLR.

Oswald, J.; Srinivas, K.; Kokel, H.; Lee, J.; Katz, M.; and Sohrabi, S. 2024. Large Language Models as Planning Domain Generators. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 34, 423–431.

Pallagani, V.; Muppasani, B.; Srivastava, B.; Rossi, F.; Horesh, L.; Murugesan, K.; Loreggia, A.; Fabiano, F.; Joseph, R.; Kethepalli, Y.; et al. 2023. Plansformer Tool: Demonstrating Generation of Symbolic Plans Using Transformers. In *IJCAI*, volume 2023, 7158–7162. International Joint Conferences on Artificial Intelligence.

Steina, K.; Fišera, D.; Hoffmanna, J.; and Kollera, A. 2024. Automating the Generation of Prompts for LLM-based Action Choice in PDDL Planning.

Toyer, S.; Trevizan, F.; Thiébaux, S.; and Xie, L. 2018. Action schema networks: Generalised policies with deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Background and Related Works

In this section, we explore foundational concepts and review significant advancements related to planning domains, generalization in planning, and domain generation techniques. We discuss traditional approaches to domain creation, the use of large language models for domain reconstruction, and challenges related to generalization in planning tasks. Additionally, we highlight domain randomization techniques in reinforcement learning and examine key domain-independent planners, such as Fast Downward and LPG, including the heuristics that support their functionality. This background provides a comprehensive understanding of the context and motivations behind developing our PDDLFuse tool for generating diverse and complex planning domains.

Planning Domains and Problems

In the context of automated planning, a *planning domain* is a structured description of an environment consisting of objects, predicates, and actions that an agent can perform. Formally, a planning domain $\mathcal{D}$ is defined by a tuple $(\mathcal{O}, \mathcal{P}, \mathcal{A})$, where:

- $\mathcal{O}$ is a finite set of *objects* that exist within the domain.
- $\mathcal{P}$ is a finite set of *predicates*, where each predicate represents a property or relationship among objects (e.g., `At(location, package)`).
- $\mathcal{A}$ is a finite set of *actions*, where each action $a \in \mathcal{A}$ is defined by a pair $(\text{pre}(a), \text{eff}(a))$:
  - pre(a), the *preconditions* of action a, is a set of predicates that must hold true for a to be executed.
  - eff(a), the *effects* of action a, is a set of predicates that describe the changes in the environment after a is executed.

A *planning problem* specifies a particular task within a domain by defining both an initial and a goal state. Formally, a planning problem $\mathcal{P}$ is defined by the tuple $(\mathcal{D}, s_0, s_g)$, where:

- $\mathcal{D}$ is the domain in which the problem is defined.
- s_0 is the *initial state*, a set of grounded predicates representing the environment's state before planning begins.
- s_g is the *goal state*, a set of grounded predicates specifying the desired conditions that define the successful completion of the task.

Given a domain $\mathcal{D}$ and problem $\mathcal{P}$, a *plan* π is a sequence of actions $(a_1, a_2, \ldots, a_n)$ that, when applied in order from s_0, transitions the initial state to a state where s_g holds true. The objective of a planner is to find a valid plan π that satisfies s_g, optimizing for factors such as plan length or action cost in some cases.

The structured nature of planning domains and problems allows automated planners to systematically search for solutions, making planning a core component of decision-making systems in AI.

Domain Reconstruction

Several studies have focused on automating domain generation by reconstructing existing domains from natural language descriptions. For example, Oswald et al. (2024) proposed a framework where LLMs are used to reconstruct planning domains from textual descriptions, aligning closely with ground-truth PDDL specifications. However, this approach requires a reference domain for validation, restricting its applicability to pre-existing domains and limiting its capacity to foster novel, unencountered domains. This reliance on predefined standards constrains the generation of diverse domains necessary for improved generalization in planning.

Similarly, Mahdavi et al. (2024) introduced an iterative refinement process that leverages environment feedback to enhance LLM-generated PDDL domains. While this approach reduces human intervention, it remains heavily reliant on environmental validation and is focused on modifying existing structures rather than generating new domains. The need for accessible validation functions also limits its scalability.

The "Translate-Infer-Compile" (TIC) framework by Agarwal and Sreepathy (2024) translates natural language descriptions into structured intermediate representations, later compiled into PDDL task files using a logic reasoner. This method improves translation accuracy but is limited to domain reconstruction, making it unsuitable for creating diverse, novel domains. Steina et al. (2024) also proposed "AUTOPLANBENCH" to convert PDDL files into natural language prompts for LLM-based action choice, enhancing LLM interaction but without expanding the pool of available domains beyond pre-existing templates.

While these studies contribute valuable methods for domain reconstruction, they cannot generate new, diverse domains essential for robust planning generalization. DomGenX addresses this gap by generating novel domains independent of existing templates, significantly enhancing the diversity of available domains. This expanded domain pool enables the training of planning models with stronger inductive biases, enhancing adaptability across a wider range of real-world challenges.

Generalization in Planning

Achieving generalization in planning remains challenging, largely due to the limited diversity in available training domains, such as those in the International Planning Competition (IPC). This lack of domain variety results in weaker inductive biases within machine learning models, as they learn patterns from a narrow training set rather than developing broadly applicable knowledge. Robust inductive bias, which enables models to recognize generalizable patterns across diverse domains, is essential for effective adaptation to new and varied domains. Without this, models are prone to overfitting, limiting their adaptability in real-world applications.

Recent works have attempted to improve generalization by using Graph Neural Networks (GNNs) and Large Language Models (LLMs), though these approaches remain constrained by the limited set of domains. For instance, Chen, Thiébaux, and Trevizan (2024) proposed novel graph representations to learn domain-independent heuristics using GNNs. While effective, this approach faces scalability challenges with large graphs and relies on grounded representations that limit flexibility. Similarly, the GOOSE framework by Chen, Thiébaux, and Trevizan (2023) uses GNNs for learning heuristics, but its reliance on IPC domains restricts its generalization potential.

Toyer et al. (2018) explored GNN-based heuristics aimed at improving plan quality, though their method's dependence on supervised learning with moderately challenging instances assumes accessible solvers, which is impractical for many real-world domains. Likewise, LLMs pre-trained on large-scale text datasets have shown limited success in planning contexts. Despite advancements in prompting techniques (Hu et al. 2023; Yao et al. 2023), LLMs still lack the domain diversity needed for robust generalization.

Studies on multimodal and code-based models, such as those by Lu et al. (2023) and Pallagani et al. (2023), similarly reveal that these models, though effective on in-distribution tasks, struggle to generalize to out-of-distribution domains, further underscoring the need for broader domain diversity to support generalization across varied settings.

Our work introduces DomGenX, a novel domain and problem generator, to address this gap. Unlike prior methods restricted to finite sets of domains, DomGenX combines existing domain files to generate a broad range of novel, randomized domains, substantially increasing the diversity of training data. By training on this expanded set of domains, DomGenX fosters stronger inductive biases, enhancing the potential for planning systems to generalize effectively across diverse and unseen problem domains. This approach ultimately provides a foundation for more adaptable and robust planning algorithms.

Domain Randomization and Generalization in Reinforcement Learning

Domain randomization has proven to be a powerful tool in reinforcement learning (RL). It trains agents across diverse environments, enhancing their adaptability and robustness.

Mehta et al. (2020) introduced Active Domain Randomization (ADR) to tackle high-variance policies seen in zero-shot transfer. By selecting challenging environment parameters, ADR focuses training on difficult domains, improving policy robustness in tasks such as robotic control. In locomotion tasks, Ajani, Hur, and Mallipeddi (2023) evaluated domain randomization by varying environmental parameters like surface friction, showing that specific randomization improves RL agents' generalization to unseen environments, reinforcing the importance of controlled diversity for real-world transfer. Kang, Chang, and Choi (2024) proposed Balanced Domain Randomization (BDR) to address training imbalances by emphasizing rare, challenging domains. This method enhances worst-case performance, making agents more robust in unpredictable settings, which underscores the value of diverse training conditions. Lastly, Koo, Yu, and Lee (2019) used adversarial domain adaptation for RL in dialog systems to align feature representations across domains, thus improving policy generalization in complex, non-physical tasks.

These studies collectively illustrate that training over diverse domains strengthens generalization across RL tasks by diversifying training contexts, aligning with DomGenX's goal to generate domains for enhanced generalization in planning.

Domain Independent Planner

Fast Downward Planner. Fast Downward is a highly versatile planning system that operates based on the planning graph concept, employing a more efficient representation known as multi-valued planning tasks. Developed by Torsten Helmert, Fast Downward has been prominent in the planning community and successful in several planning competitions. It efficiently translates PDDL (Planning Domain Definition Language) tasks into a compact internal representation, facilitating more effective planning solutions. The system's modularity allows for the application of various search algorithms and heuristics, tailored to specific problem types.

- **FF Heuristic** : The FF heuristic, or "Fast-Forward" heuristic, is central to the Fast Downward planner, developed by Joerg Hoffmann and Bernhard Nebel. It is recognized for its rapid and effective planning capabilities, mainly due to its approach of ignoring the delete lists of actions, which simplifies the search process significantly. This heuristic generates relaxed plans by considering only the positive effects of actions, enabling quick heuristic calculations and efficient plan generation.
- **lmcut Heuristic** : The landmark-cut (lmcut) heuristic, another innovative heuristic used within the Fast Downward framework, calculates the minimum cost of achieving all necessary landmarks in a planning task. A landmark is a fact or a set of facts that must be true at some point in every valid plan. This heuristic identifies critical paths and bottlenecks in the plan's causal graph, aiding in the formulation of more efficient solutions.

LPG Planner. The LPG (Local Search for Planning Graphs) planner utilizes stochastic local search techniques to effectively handle propositional and numerical planning problems. Developed by Alfonso Gerevini and Ivan Serina, LPG iteratively refines a candidate plan through a combination of action-graph refinement and plan-graph expansion, showcasing robust performance across diverse domains.

Methods

In this section, we provide detailed descriptions of the steps used within PDDLFuse to ensure the generation of diverse and solvable planning domains. The methods outlined include handling overlapping predicate and action names when combining two domains, dynamically generating action sequences to simulate goal states, and validating the generated domains and problems against PDDL 3.1 standards. These processes enhance the robustness and generalizability of generated domains, providing a reliable foundation for testing planning algorithms. Detailed algorithms for each method are discussed below.

Handling Overlapping Predicate and Action Names

Handling overlapping predicates and action names is crucial for maintaining the integrity and uniqueness of domain definitions when combining two existing domains. Overlapping elements can cause logical conflicts and inaccuracies in domain behavior during planning tasks. This algorithm identifies overlaps and systematically renames the conflicting elements in one domain to prevent ambiguity.

Algorithm 3: Handling Overlapping Names

```
Require: Two domains $\mathcal{D}_1$ and $\mathcal{D}_2$ with potential overlapping predicates and actions.
Ensure: Unique predicates and actions across $\mathcal{D}_1$ and $\mathcal{D}_2$.
 1: Initialize set $\mathcal{O}_1$ for unique objects from $\mathcal{D}_1$.
 2: Initialize set $\mathcal{O}_2$ for unique objects from $\mathcal{D}_2$.
 3: for each predicate or action in $\mathcal{D}_1$ and $\mathcal{D}_2$ do
 4:     if exists in both $\mathcal{D}_1$ and $\mathcal{D}_2$ then
 5:         Rename in $\mathcal{D}_2$.
 6:     end if
 7: end for
 8: return Updated $\mathcal{D}_1$ and $\mathcal{D}_2$ with unique names.
```

Action Generator Process

The Action Generator is designed to dynamically produce a sequence of actions based on the specifications of a given domain's initial state. It simulates random actions from the domain's action set, transforming the initial state into a new state that can potentially serve as a goal state for planning problems. The goal state is determined by selecting a subset of predicates active in the reached state, ensuring that there exists a sequence of actions that leads to this state.

Algorithm 4: Action Generator Process

```
Require: Generated domain $\mathcal{D} = (\mathcal{O}, \mathcal{P}, \mathcal{A})$, Initial state $s_0$
 1: Initialize current_state $\leftarrow s_0$
 2: Initialize action_sequence $\leftarrow []$
 3: for $i = 1$ to $N$ do
 4:     Select $a \in \mathcal{A}$ randomly such that preconditions of $a$ are satisfied in current_state
 5:     Apply $a$ to current_state
 6:     Append $a$ to action_sequence
 7:     Update current_state based on the effects of $a$
 8: end for
 9: Define goal state $s_g$ as a subset of predicates true in current_state
10: return action_sequence, $s_g$
```

Validator Process

The Validator ensures that the generated domains and problems conform to PDDL 3.1 standards, utilizing a parser to validate the structural and logical correctness of the domain and problem definitions. It checks for consistency in the domain's actions and the feasibility of achieving the problem's goal state from its initial state based on the defined actions.

Algorithm 5: Validation Process

```
Require: Domain $\mathcal{D} = (\mathcal{O}, \mathcal{P}, \mathcal{A})$, Problem $\mathcal{P} = (\mathcal{D}, s_0, s_g)$
 1: Parse $\mathcal{D}$ and $\mathcal{P}$ using a PDDL 3.1 parser
 2: Check for syntactical correctness of $\mathcal{D}$ and $\mathcal{P}$
 3: Verify logical consistency: all actions in $\mathcal{A}$ must correctly transform predicates from $s_0$ to achieve $s_g$
 4: if all checks pass then
 5:     Return "Validation Successful"
 6: else
 7:     Return "Validation Failed"
 8: end if
```

Results

In this section, we present the outcomes of our experiments, evaluating the PDDLFuse tool's efficacy in generating complex planning domains and the performance of domain-independent planners across various configurations. We assess the robustness of our validator, analyze planner solvability under different parameter variations, and examine planner performance across increasing domain depths. These results underscore the versatility and challenge of the generated domains, demonstrating the value of PDDLFuse for advancing research in automated planning. Detailed findings are discussed below.

Evaluating the Validator

To assess the robustness and accuracy of our validator, we conducted evaluations across eight diverse domains, each with 20 problem instances. The problem data was sourced from the study by Chen, Thiébaux, and Trevizan (2023), which used the Scorpion planner to generate optimal plans for each domain and problem file. This evaluation focused on two primary aspects: (1) verifying that the validator could correctly interpret and check the syntax of both the domain and problem files, and (2) ensuring that it could execute each optimal plan step-by-step to reach the specified goal state.

Table 2: Validator Performance Across Domains

Domain	Reached Goal State	Success Rate
Blocks	Yes	100%
Ferry	Yes	100%
Gripper	Yes	100%
N-Puzzle	Yes	100%
Sokoban	Yes	100%
Spanner	Yes	100%
Visitall	Yes	100%
Visitsome	Yes	100%

As summarized in Table 2, the validator demonstrated perfect performance, successfully reaching the goal state in

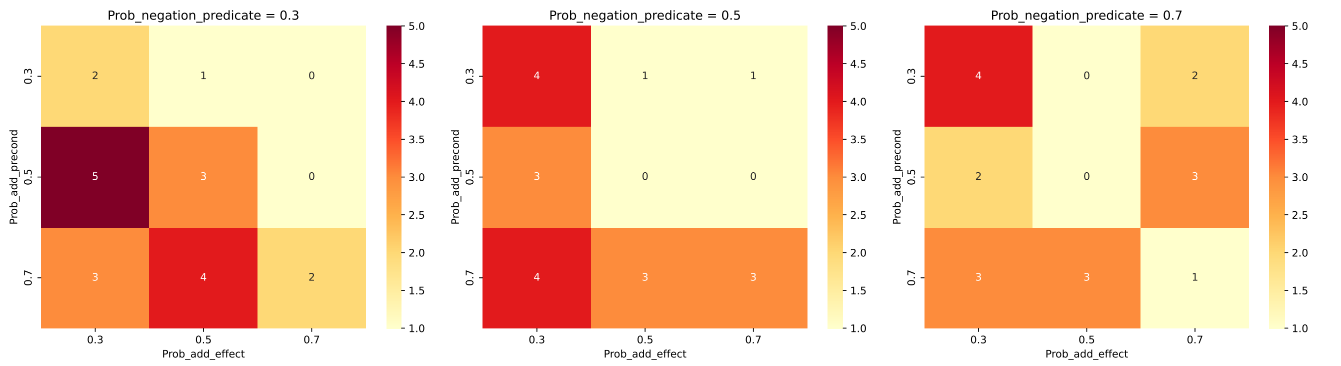

Figure 5: Solvability heat maps for domains with 15 objects, evaluated using the Fast Downward planner with lmcut heuristic across varied parameter configurations.

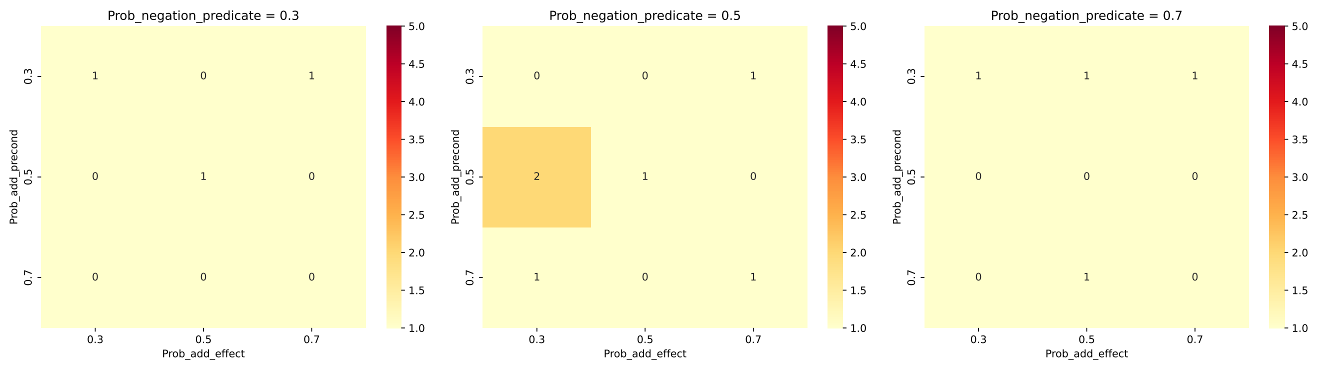

Figure 6: Solvability heat maps for domains with 15 objects, evaluated using the LPG Planner across varied parameter configurations.

100% of cases across all tested domains. Each optimal plan was validated without error, affirming the validator's ability to reliably execute complex action sequences across a range of domain types.

This experiment underscores the validator's essential role in confirming both the syntactic integrity and operational feasibility of plans within diverse domains. By accurately verifying optimal plans across varied problem sets, the validator supports robust evaluation and validation of planning solutions, ensuring that generated plans align with intended outcomes across different planning contexts. This high level of reliability is critical for advancing automated planning systems that depend on accurate and interpretable validation processes.

Solvability Across Parameter Variations

We conducted experiments using five base domains—Blocks-World, Gripper, Depot, Grid, and Satellite—focusing on depth level 1 with num_obj = 15. To evaluate planner performance under various configurations, the following parameters were systematically varied:

- `Prob_add_precond` and `Prob_remove_precond`: (0.3, 0.7), (0.5, 0.5), (0.7, 0.3)
- `Prob_add_effect` and `Prob_remove_effect`: (0.3, 0.7), (0.5, 0.5), (0.7, 0.3)

Additionally, we varied prob_neg with values of 0.3, 0.5, and 0.7, keeping rev_flag = True throughout the experiments.

Figure 5 Analysis: The solvability heat maps illustrate the performance of the Fast Downward planner with the lmcut heuristic on domains with 15 objects across varied parameter configurations. Each cell's value indicates the number of solvable instances within a specific configuration, combining different probabilities for adding and removing preconditions and effects, as well as varying negation probabilities. Results indicate that configurations with balanced add/remove probabilities and moderate negation values yield better solvability, while more extreme parameter settings pose greater challenges. This highlights the limitations of the lmcut heuristic in handling high-complexity domains generated by extreme parameter values.

Figure 6 Analysis: Figure 6 presents the solvability heat maps for the LPG planner on domains with 15 objects, analyzed across a range of parameter settings. Like the FD, the LPG planner shows a decrease in solvability for configurations with high add/remove probabilities or negation values, indicating its sensitivity to complex domain setups. These

findings emphasize LPG's adaptability in certain configurations and reveal its struggles in randomized, diverse domains, further underscoring the necessity of diverse domain configurations for robust planner evaluation.

Solvability Across Parameter Variations and Depth

We evaluated the solvability of the generated domains across varying depths by conducting experiments with diverse domain configurations. Each configuration was defined by a unique combination of parameters, including probabilities for adding and removing precondition and effect predicates, negation probabilities, the number of objects, and predicate reversibility settings. Due to memory constraints on our system during the data generation process, the number of problems generated at each depth and the depths vary across configurations.

The tables below summarize the results, presenting the number of problems each planner (Fast Downward with FF and lmcut heuristics and LPG) successfully solves for different depths, parameter values, and domain characteristics. These results highlight the diversity and complexity of the generated domains, as even robust domain-independent planners face challenges in solving them, underscoring the effectiveness of our domain generator in creating complex domains.

Depth	FD(ff)	FD(lmcut)	LPG
1	5/5	5/5	2/5
2	5/5	5/5	0/5
3	5/5	5/5	0/5
4	1/1	1/1	0/1

Table 3: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	5/5	1/5	0/5
2	5/5	5/5	1/5
3	5/5	4/5	0/5
4	5/5	4/5	0/5
5	5/5	4/5	0/5

Table 4: Solvability of Generated Domains for Depth 1 to 5, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	5/5	1/5	3/5
2	5/5	4/5	1/5
3	3/5	3/5	0/5
4	3/3	3/3	0/3

Table 5: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	5/5	0/5	1/5
2	2/5	2/5	0/5
3	4/4	4/4	0/4

Table 6: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	5/5
2	2/5	4/5	2/5
3	4/5	4/5	0/5
4	4/5	4/5	1/5

Table 7: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	1/5	0/5
2	3/5	3/5	2/5
3	1/5	4/5	2/5
4	2/2	2/2	0/2

Table 8: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	1/5
2	1/5	3/5	1/5
3	5/5	5/5	0/5
4	5/5	4/5	0/5
5	2/2	2/2	0/2

Table 9: Solvability of Generated Domains for Depth 1 to 5, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	0/5
2	3/5	3/5	2/5
3	4/5	4/5	0/5
4	5/5	5/5	0/5
5	2/2	2/2	0/2

Table 10: Solvability of Generated Domains for Depth 1 to 5, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	5/5
2	4/5	4/5	0/5
3	5/5	5/5	0/5
4	0/1	0/1	0/1

Table 11: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	3/5
2	3/5	3/5	3/5
3	4/5	4/5	0/5

Table 12: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	3/5
2	2/5	2/5	1/5
3	2/3	2/3	0/3

Table 13: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	2/5
2	4/5	4/5	1/5
3	3/4	3/4	0/4

Table 14: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	2/5
2	3/5	3/5	1/5
3	4/4	4/4	0/4

Table 15: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	4/5
2	4/5	3/5	0/5
3	3/4	3/4	0/4

Table 16: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	2/5
2	3/5	4/5	0/5

Table 17: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	2/5	3/5
2	4/5	4/5	0/5
3	2/4	2/4	0/4

Table 18: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	1/5
2	3/5	3/5	0/5
3	4/5	4/5	0/5

Table 19: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	3/5
2	2/5	2/5	0/5
3	3/4	3/4	0/4

Table 20: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	0/5
2	5/5	4/5	1/5
3	4/5	4/5	0/5

Table 21: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	1/5
2	3/5	3/5	1/5
3	5/5	4/5	0/5
4	4/5	4/5	0/5
5	1/1	0/1	0/1

Table 22: Solvability of Generated Domains for Depth 1 to 5, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	3/5
2	5/5	4/5	0/5
3	5/5	5/5	0/5
4	5/5	5/5	0/5
5	2/2	2/2	0/2

Table 23: Solvability of Generated Domains for Depth 1 to 5, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	1/5
2	4/5	4/5	0/5
3	5/5	4/5	0/5
4	2/3	3/3	0/3

Table 24: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	1/5
2	3/5	4/5	1/5
3	3/4	3/4	0/4

Table 25: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	2/5
2	5/5	3/5	0/5
3	2/4	3/4	0/4

Table 26: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	1/5
2	5/5	4/5	0/5
3	5/5	5/5	0/5

Table 27: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	1/5	0/5
2	5/5	5/5	0/5
3	5/5	5/5	0/5

Table 28: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	1/5
2	5/5	3/5	0/5
3	5/5	5/5	1/5
4	2/3	3/3	0/3

Table 29: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	1/5	0/5
2	3/5	4/5	1/5
3	4/4	4/4	0/4

Table 30: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	4/5
2	4/5	4/5	0/5
3	4/4	4/4	0/4

Table 31: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	3/5
2	3/5	3/5	0/5
3	4/5	4/5	0/5

Table 32: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	1/5
2	3/5	3/5	0/5
3	5/5	4/5	0/5
4	3/3	3/3	0/3

Table 33: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	2/5
2	4/5	3/5	0/5
3	5/5	5/5	0/5
4	5/5	5/5	0/5

Table 34: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	4/5
2	4/5	4/5	0/5
3	4/5	5/5	0/5

Table 35: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	2/5
2	5/5	5/5	0/5
3	1/1	1/1	0/1

Table 36: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	0/5	4/5
2	3/5	3/5	0/5

Table 37: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	2/5
2	5/5	4/5	0/5
3	2/4	2/4	0/4

Table 38: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	1/5	0/5
2	4/5	3/5	0/5
3	1/1	1/1	0/1

Table 39: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	0/5
2	4/5	3/5	0/5

Table 40: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	1/5
2	3/5	4/5	0/5
3	4/4	2/4	0/4

Table 41: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	0/5
2	3/5	3/5	0/5
3	4/4	3/4	0/4

Table 42: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	4/5	0/5
2	5/5	4/5	0/5
3	4/5	5/5	0/5
4	2/2	2/2	0/2

Table 43: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	2/5	1/5
2	5/5	3/5	0/5

Table 44: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	0/5
2	5/5	3/5	0/5
3	5/5	4/5	0/5

Table 45: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	0/5
2	3/5	3/5	0/5
3	4/4	4/4	0/4

Table 46: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	2/5
2	2/5	3/5	0/5
3	5/5	5/5	0/5
4	5/5	5/5	0/5

Table 47: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	0/5
2	4/5	4/5	0/5
3	5/5	5/5	0/5
4	3/3	3/3	0/3

Table 48: Solvability of Generated Domains for Depth 1 to 4, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	3/5	2/5
2	2/5	3/5	0/5

Table 49: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	0/5	1/5
2	3/5	0/5	0/5
3	5/5	0/5	0/5

Table 50: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	0/5	0/5
2	4/5	0/5	0/5

Table 51: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	0/5	0/5
2	3/5	0/5	0/5
3	4/4	0/4	0/4

Table 52: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	5/5	2/5
2	4/5	3/5	0/5
3	3/4	0/4	0/4

Table 53: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	0/5
2	4/5	4/5	0/5
3	3/4	4/4	0/4

Table 54: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	0/5
2	5/5	4/5	0/5
3	4/4	4/4	0/4

Table 55: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	2/5	0/5
2	4/5	5/5	0/5
3	3/3	3/3	0/3

Table 56: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 5.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	5/5	1/5
2	5/5	5/5	0/5
3	3/3	3/3	0/3

Table 57: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	2/5	1/5
2	1/1	1/1	0/1

Table 58: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	3/5
2	2/5	2/5	1/5

Table 59: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	2/5
2	1/1	1/1	0/1

Table 60: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	2/5
2	4/5	4/5	1/5

Table 61: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	4/5	2/5
2	1/2	1/2	0/2

Table 62: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	1/5
2	4/5	4/5	0/5

Table 63: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	3/5
2	1/1	1/1	0/1

Table 64: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	1/5
2	3/5	3/5	0/5

Table 65: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	2/5
2	2/5	2/5	0/5

Table 66: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	1/5
2	0/1	0/1	0/1

Table 67: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	4/5	2/5
2	0/1	0/1	0/1

Table 68: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	2/5
2	0/1	0/1	0/1

Table 69: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	1/5
2	0/1	0/1	0/1

Table 70: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	1/5
2	0/1	0/1	0/1

Table 71: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	2/5
2	3/5	2/5	0/5

Table 72: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	1/5
2	0/1	0/1	0/1

Table 73: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	2/5
2	0/1	0/1	0/1

Table 74: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	1/5
2	4/5	4/5	1/5

Table 75: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	2/5	1/5
2	4/5	4/5	0/5
3	3/3	3/3	0/3

Table 76: Solvability of Generated Domains for Depth 1 to 3, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	2/5
2	1/1	0/1	0/1

Table 77: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	5/5	3/5	2/5
2	4/5	3/5	0/5

Table 78: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	2/5
2	1/1	1/1	1/1

Table 79: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	2/5
2	1/1	1/1	0/1

Table 80: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	1/5
2	3/5	2/5	0/5

Table 81: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	0/5
2	1/1	1/1	0/1

Table 82: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	3/5
2	1/1	1/1	0/1

Table 83: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	2/5
2	2/2	2/2	0/2

Table 84: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	4/5	0/5
2	1/1	1/1	0/1

Table 85: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	0/5
2	1/2	1/2	0/2

Table 86: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	0/5	2/5
2	1/1	1/1	0/1

Table 87: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	0/5
2	1/1	1/1	0/1

Table 88: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	0/5
2	1/1	1/1	0/1

Table 89: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	1/5
2	1/1	1/1	0/1

Table 90: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	0/5
2	0/1	0/1	0/1

Table 91: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	2/5
2	1/1	1/1	0/1

Table 92: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	0/5
2	1/1	1/1	0/1

Table 93: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	5/5	0/5	0/5
2	1/1	0/1	0/1

Table 94: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	2/5
2	2/2	2/2	0/2

Table 95: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	0/5
2	0/1	1/1	0/1

Table 96: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	2/5
2	1/1	1/1	0/1

Table 97: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	4/5	2/5
2	5/5	5/5	0/5

Table 98: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	3/5	0/5
2	5/5	5/5	0/5

Table 99: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	0/5
2	1/1	0/1	0/1

Table 100: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	0/5
2	1/1	1/1	0/1

Table 101: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	0/5
2	1/1	0/1	0/1

Table 102: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	0/5
2	1/1	1/1	0/1

Table 103: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	1/5	0/5
2	1/1	1/1	0/1

Table 104: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	0/5
2	1/1	1/1	0/1

Table 105: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	1/5
2	1/1	1/1	0/1

Table 106: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	1/5
2	1/1	1/1	0/1

Table 107: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	0/5
2	5/5	5/5	0/5

Table 108: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	0/5
2	1/1	1/1	0/1

Table 109: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	1/5
2	1/1	1/1	0/1

Table 110: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 10.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	1/5

Table 111: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	1/5

Table 112: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	4/5	0/5
2	1/1	0/1	0/1

Table 113: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	5/5	5/5	1/5

Table 114: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	4/5	1/5

Table 115: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/4	2/4	1/4

Table 116: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	1/5	0/5

Table 117: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	0/5

Table 118: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	0/5

Table 119: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	1/5
2	0/1	0/1	0/1

Table 120: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	0/5	1/5
2	0/1	0/1	0/1

Table 121: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	0/5	1/5

Table 122: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	0/5	1/5

Table 123: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/4	1/4	1/4

Table 124: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	1/5

Table 125: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/4	0/4	1/4

Table 126: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	2/5	1/5
2	0/1	0/1	0/1

Table 127: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/4	1/4	1/4

Table 128: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	5/5	0/5
2	1/1	1/1	0/1

Table 129: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	2/5

Table 130: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	2/5

Table 131: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	5/5	2/5

Table 132: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	0/5

Table 133: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	4/5	1/5
2	1/1	1/1	0/1

Table 134: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	1/5

Table 135: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	1/5
2	1/1	1/1	0/1

Table 136: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	1/5

Table 137: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	1/5

Table 138: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	0/5

Table 139: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/4	0/4	1/4

Table 140: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	0/5
2	0/1	0/1	0/1

Table 141: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	2/5

Table 142: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	0/5

Table 143: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/5	0/5	2/5

Table 144: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	0/5

Table 145: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	1/5	0/5
2	1/1	1/1	0/1

Table 146: Solvability of Generated Domains for Depth 1 to 2, with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	4/5	3/5	0/5

Table 147: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/4	3/4	1/4

Table 148: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	4/5	1/5

Table 149: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/4	3/4	1/4

Table 150: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	0/5

Table 151: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/4	2/4	0/4

Table 152: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	4/5	0/5

Table 153: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	2/5	1/5

Table 154: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	0/5

Table 155: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/4	4/4	1/4

Table 156: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	3/5	1/5

Table 157: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	1/5

Table 158: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/5	2/5	0/5

Table 159: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/4	2/4	1/4

Table 160: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	3/5	3/5	1/5

Table 161: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	0/4	2/4	0/4

Table 162: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	1/5	1/5	0/5

Table 163: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/4	2/4	0/4

Table 164: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = 15.0

Depth	FD(ff)	FD(lmcut)	LPG
1	2/2	1/2	1/2

Table 165: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	2/2	2/2	1/2

Table 166: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	1/3

Table 167: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/2	1/2	1/2

Table 168: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/3	1/3	0/3

Table 169: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/3	1/3	1/3

Table 170: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.3, prob_rem_pre = 0.7, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/3	1/3	1/3

Table 171: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/2	0/2	0/2

Table 172: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 173: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 174: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	1/2

Table 175: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 176: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.5, prob_rem_pre = 0.7, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 177: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 178: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	1/2

Table 179: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 180: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 181: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	1/2

Table 182: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.3, prob_add_eff = 0.7, prob_rem_pre = 0.7, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/4	0/4	0/4

Table 183: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 184: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/2	1/2	0/2

Table 185: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/2	1/2	0/2

Table 186: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	1/2	2/2	0/2

Table 187: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	2/3	2/3	0/3

Table 188: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.3, prob_rem_pre = 0.5, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 189: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 190: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 191: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 192: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 193: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	2/3	2/3	1/3

Table 194: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.5, prob_rem_pre = 0.5, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 195: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 196: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 197: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 198: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 199: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 200: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.5, prob_add_eff = 0.7, prob_rem_pre = 0.5, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 201: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 202: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 203: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	0/3

Table 204: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 205: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	1/3

Table 206: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.3, prob_rem_pre = 0.3, prob_rem_eff = 0.7, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	1/3

Table 207: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/2	0/2	0/2

Table 208: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	1/3

Table 209: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	1/3

Table 210: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	FD(ff)	FD(lmcut)	LPG
1	0/3	0/3	1/3

Table 211: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	**FD(ff)**	**FD(lmcut)**	**LPG**
1	2/3	2/3	1/3

Table 212: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.5, prob_rem_pre = 0.3, prob_rem_eff = 0.5, negation = 0.7, predicate reversibility = False, number of objects = None

Depth	**FD(ff)**	**FD(lmcut)**	**LPG**
1	2/2	2/2	0/2

Table 213: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = True, number of objects = None

Depth	**FD(ff)**	**FD(lmcut)**	**LPG**
1	2/2	1/2	1/2

Table 214: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.3, predicate reversibility = False, number of objects = None

Depth	**FD(ff)**	**FD(lmcut)**	**LPG**
1	1/2	2/2	0/2

Table 215: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = True, number of objects = None

Depth	**FD(ff)**	**FD(lmcut)**	**LPG**
1	1/2	1/2	0/2

Table 216: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.5, predicate reversibility = False, number of objects = None

Depth	**FD(ff)**	**FD(lmcut)**	**LPG**
1	0/2	0/2	0/2

Table 217: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = True, number of objects = None

Depth	**FD(ff)**	**FD(lmcut)**	**LPG**
1	1/2	1/2	1/2

Table 218: Solvability of Generated Domains for Depth 1 , with probability values: prob_add_pre = 0.7, prob_add_eff = 0.7, prob_rem_pre = 0.3, prob_rem_eff = 0.3, negation = 0.7, predicate reversibility = False, number of objects = None