Building and using a planning ontology from past data for performance efficiency

Bharath Muppasani[1], Vishal Pallagani[1], Biplav Srivastava[1], Raghava Mutharaju[2], Michael N. Huhns[1], and Vignesh Narayanan[1]

[1] University of South Carolina, USA
[2] Indraprastha Institute of Information Technology, IIIT-Delhi, India
bharath@email.sc.edu, vishalp@mailbox.sc.edu, biplav.s@sc.edu, raghava.mutharaju@iiitd.ac.in, huhns@sc.edu, vignar@sc.edu

Abstract. Ontologies are known for their ability to organize rich metadata information, support identification of novel insights via semantic queries, and promote reuse. In this paper, we consider the problem of automated planning where the objective is to find a sequence of actions that will move an agent from an initial state of the world to a desired goal state. We hypothesize that given the large number of available planners and diverse planning domains, they carry essential information that can be leveraged to identify suitable planners and improve their performance for a domain. We use data on planning domains and planners from the International Planning Competition (IPC) to construct a planning ontology and then, show via experiments in two usecases, that the ontology can lead to selection of promising planners and improving their performance using macros - a form of action ordering constraints extracted from planning ontology. We also make the planning ontology and associated resources available to the community to promote further research.

Keywords: Ontology · Automated Planning · Planner Improvement.

1 Introduction

Automated planning, where the objective is to find a sequence of actions that will transition an agent from the initial state of the world to a desired goal state, is an active sub-field of Artificial Intelligence (AI). The ability to generate plans and make decisions in complex domains, such as robotics, logistics, and manufacturing, has led to significant progress in the automation of planning. Currently, there are numerous planning domains, planners, search algorithms, and associated heuristics in the field of automated planning. Each planner, in conjunction with a search algorithm and heuristic, generates plans with varying degrees of quality, cost, and optimality. The empirical results available for various planning problems, ranked by planner performance and the heuristics used as available in International Planning Competition (IPC), can provide valuable information to identify various tunable parameters to improve planner performance. Traditionally, improving planner performance involves manually curating potential combinations to identify the optimal planner configuration. However, there has been limited effort to model the available information in a structured knowledge representation, such as an ontology, to facilitate efficient reasoning and further enhance planner performance.

To address the challenge of representing planning problems and associated information in a structured manner, we propose an ontology for AI planning. An ontology is a formal representation of concepts and their relationships [6], which enables systematic analysis of planning domains and planners. The proposed ontology captures the features of a domain and the capabilities of planners, facilitating reasoning with existing planning problems, identifying similarities, and suggesting different planner configurations. Planning ontology can also be a useful resource for the creation of new planners as it captures essential information about planning domains and planners, which can be leveraged to design more efficient planning algorithms. Furthermore, ontology can promote knowledge sharing and collaboration within the planning community.

In the field of planning, several attempts have been made to create ontologies to enhance the understanding of planners' capabilities. For instance, Plan-Taxonomy [2] introduced a taxonomy that aimed to explain the functionality of planners. Additionally, authors in [5] present a comprehensive ontology called PLANET, which represents plans in real-world domains and can be leveraged to construct new applications. Nonetheless, the reusability of PLANET is limited as it is not open-sourced. Consequently, researchers face difficulty in extending or replicating the ontology.

This paper outlines our methodology for constructing an ontology to represent AI planning domains, leveraging information obtained from the IPC. Building a planning ontology using data from IPC offers several benefits such as a comprehensive coverage of planning domains, a rich source for various benchmark evaluation metrics, and documentation for the planners. However, the ontology is not limited to the PDDL representation or domains in IPC, and can easily be extended to any. Our contributions are at the intersection of ontologies and AI planning in ontology development and showing benefits in two usecases:

- **Ontology Development**: We developed an ontology for AI planning that can be used to represent and organize knowledge related to planning problems. This ontology provides a structured way to capture the relationships between different planning concepts and enables more efficient and effective knowledge sharing and reuse.
- **Usecase 1: Identifying Most Promising Planner for Performance**: We demonstrate the ontology's usage for identifying the best-performing planner for a specific planning domain using data of IPC-2011.
- **Usecase 2: Macro Selection for Improving Planner Performance**: We demonstrate the usage of ontology to extract domain-specific macros - which are action orderings - and show that they can improve planner performance drastically.

In the remainder of the paper, we start with preliminaries about ontologies, followed by automated planning and IPC. We then give an overview of the existing literature on ontologies for planning. Following this, we present a detailed description of the ontology construction process and its usage. We then end the paper with a discussion of the proposed planning ontology and conclude with future research directions.

2 Preliminaries

In this section, we describe the necessary background for automated planning and the significance of the International Planning Competition.

2.1 Ontology

A formal ontology is typically represented as a set of concepts, relations, and axioms. A concept represents a set of objects or entities that share common properties, while a relation represents a connection or association between two or more concepts. Axioms are statements that define the relationships between concepts and relations. It is a formal representation of knowledge that is designed to facilitate automated reasoning and information processing. It acts as a structured vocabulary that describes a domain and promotes interoperability, data integration, and communication between humans and machines. Formally, an ontology O can be represented as a tuple (C, R, A), where C is the set of concepts, R is the set of relations, and A is the set of axioms. Each concept $c \in C$ can be represented as a set of attributes, denoted as $Att(c)$. Similarly, each relation $r \in R$ can be represented as a set of attributes, denoted as $Att(r)$.

Ontology is a branch of philosophy that deals with the nature of existence and being. In the field of computer science, however, ontology refers to a formal representation of knowledge that is designed to facilitate automated reasoning and information processing. It is a structured vocabulary that describes a domain and promotes interoperability, data integration, and communication between humans and machines. Various tools and methodologies, including Protege and ontology editors, are available for ontology creation. Ontologies are increasingly important in artificial intelligence, knowledge engineering, and the semantic web, and researchers are exploring their potential in diverse domains and applications.

2.2 Automated Planning

Automated planning, also known as AI planning, is the process of finding a sequence of actions that will transform an initial state of the world into a desired goal state. It involves constructing a plan or a sequence of actions that will achieve a specified objective while respecting any constraints or limitations that may be present. Formally, automated planning can be defined as a tuple (S, A, T, I, G), where:

- S is the set of possible states of the world
- A is the set of possible actions that can be taken

- T is the transition function that describes the effects of taking an action on the current state of the world
- I is the initial state of the world
- G is the desired goal state

Using this notation, the problem of automated planning can be framed as finding a sequence of actions $\prec a_1, a_2, ..., a_k \succ$ that will transform the initial state I into the goal state G, while respecting any constraints or limitations on the actions. A problem is defined in terms of a domain and a problem instance. The domain defines the possible actions that can be taken and the effects of each action, while the problem instance specifies the initial state of the world and the desired goal state. To solve the planning problem, various techniques can be used, such as search algorithms, constraint-based reasoning, and optimization methods. These techniques involve exploring the space of possible plans and selecting the one that satisfies the objective and any constraints.

Attributes modeled about a domain:

1. **Predicates:** Predicates are fundamental elements in the planning domain that define the properties of the world. They are used to describe the initial and goal states, as well as the preconditions and effects of actions. Predicates are usually defined as logical expressions over a set of variables, where each variable can take on a finite number of values. In the context of planning, predicates are typically used to represent facts about the world that can be true or false, such as the location of an object or the status of a machine.
2. **Actions:** Actions are the basic units of change in the planning domain. They represent atomic operations that can be performed to transform the world from one state to another. Each action has a name, a set of parameters, preconditions that must be satisfied before the action can be executed, and effects that describe the changes that the action makes to the world. Actions can be used to model a wide variety of operations, ranging from simple movements or transformations to complex processes such as planning or decision-making.
3. **Preconditions:** Preconditions are the conditions that must be true before an action can be executed. They are usually defined using predicates and can involve multiple variables. Preconditions can also be negative, which means that a certain condition must not be true for an action to be executed. In the context of planning, preconditions are used to ensure that actions are only executed when the necessary conditions have been met, such as ensuring that a machine is turned off before it is serviced.
4. **Effects:** Effects describe the changes that an action makes to the world. They are usually defined using predicates and can involve multiple variables. Effects can be positive, which means that a certain condition becomes true after the action is executed, or negative, which means that a certain condition becomes false after the action is executed. In the context of planning, effects are used to model the changes that result from executing an action, such as moving an object from one location to another or turning a machine on.

5. **Constants:** Constants are values that are fixed and do not change during the execution of the planning problem. They are used to represent objects or entities in the world that have a fixed value, such as the speed limit on a road. Constants can be used to simplify the planning problem by reducing the number of variables that need to be considered, and by providing a fixed set of values that can be used in predicates and actions.
6. **Types:** Types are used to classify objects or entities in the world based on their attributes or properties. They are used to define the domain of values that a variable can take on, and can be used to constrain the values that are assigned to variables. In the context of planning, types are typically used to group related objects or entities together, such as cars or bicycles, and to specify the properties that are common to all members of a type, such as their color or size.

Attributed modeled about a problem instance from a domain are:

1. **Name:** The name of the planning problem.
2. **Domain:** The name of the planning domain that the problem belongs to.
3. **Requirements:** A list of requirements that the planner must satisfy in order to solve the problem. Requirements can include things like durative actions, conditional effects, or negative preconditions.
4. **Objects:** A list of objects that are present in the planning problem. Objects are typically defined in terms of their type and name.
5. **Initial State:** A description of the initial state of the world, including the values of all relevant predicates.
6. **Goal State:** A description of the desired goal state of the world, including the values of all relevant predicates.

Each of these attributes are essential for defining a complete planning problem that can be solved by an automated planner.

2.3 International Planning Competition (IPC)

IPC serves as a significant means of assessing and comparing various planning systems. By presenting new planners and benchmark problems each year, the competitions aim to stimulate the advancement of new planning methodologies and reflect current trends and challenges in the field. The competition comprises multiple tracks, each covering various planning problems such as classical, temporal, and probabilistic planning. These tracks include benchmark problems that evaluate the performance of planners concerning parameters such as plan quality, plan length, and run time. The results of these competitions provide insights into the current state-of-the-art in planning and help identify the strengths and weaknesses of different planning systems. IPC can serve as an excellent starting point for building a planning-related ontology as the benchmark problems used in these competitions can provide a comprehensive overview of the domain and the types of problems that planners need to solve.

3 Related Work

The use of ontology-based knowledge representation and reasoning has been extensively studied in various domains, including automated planning. This section focuses on the applications of ontology-based knowledge representation and reasoning in the context of planning and related domains. In [11], an ontology is constructed for the Joint Forces Air Component Commander (JFACC) to represent knowledge from the air campaign domain. The ontology is modularized to facilitate data organization and maintenance, but its applicability is domain-specific, unlike our approach. In [13], the authors automate the knowledge discovery workflow using ontology and AI planning, creating a Knowledge Discovery (KD) ontology to represent the KD domain and converting its variables to a Planning Domain Definition Language (PDDL) format to obtain the PDDL domain. The ontology's objects represent initial and goal states, forming the KD task, which represents a specific problem. The authors use the Fast-Forward (FF) planning system to generate the required plans. In a survey of ontology-based knowledge representation and reasoning in the planning domain, [4] suggest that knowledge reasoning approaches can draw new conclusions in non-deterministic contexts and assist with dynamic planning. In [5], a reusable ontology, PLANET, is proposed for representing plans. PLANET includes representations for planning problem context, goal specification, plan, plan task, and plan task description. However, PLANET does not include representations for some entities commonly associated with planning domains, such as resources and time. Our planning ontology draws inspiration from PLANET and appends more metadata for planner improvement. In [1], a domain-independent approach is presented that advances the state of the art by augmenting the knowledge of a planning task with pertinent goal opportunities. The authors demonstrate that incorporating knowledge obtained from an ontology can aid in producing better-valued plans, highlighting the potential for planner enhancement using more tuning parameters, which are captured in our planning ontology. In general, these studies demonstrate the potential of ontology-based knowledge representation and reasoning in the planning domain, including applications such as representing plans, aiding in air campaign planning, automating knowledge discovery workflows, and developing context-aware planning services.

4 Planning Ontology

This section covers the construction of planning ontology to capture the essential details of automated planning. We will discuss the considerations, challenges, benefits, and limitations of using ontologies for automated planning, to provide a better understanding of how they can improve the efficiency and effectiveness of automated planning systems.

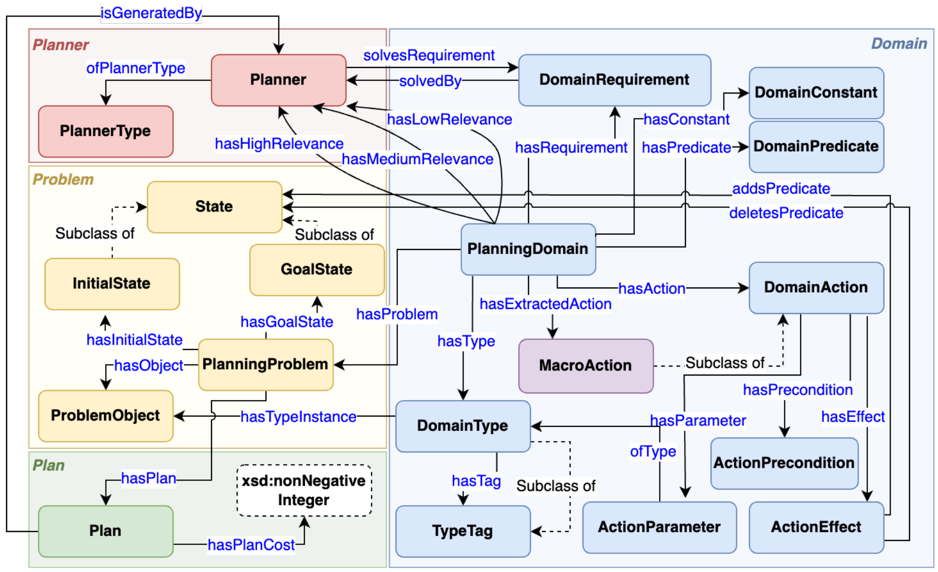

Fig. 1: Planning ontology capturing different concepts of automated planning domain, problem, plan, and planner performance separated into categories (shown as colored rectangles)

4.1 Competency Questions

Competency questions for an ontology are focused on the needs of the users who will be querying the ontology. These questions are designed to help users explore and understand the concepts and relationships within the ontology, and to find the information they need within the associated knowledge base. By answering these questions, the ontology can be better scoped and tailored to meet the needs of its users. SPARQL queries for each of these questions can be found at our GitHub Repository[2].

- C1: What are the different types of planners used in automated planning?
- C2: What is the relevance of planners in a given problem domain?
- C3: What are the available actions for a given domain?
- C4: What problems in a domain satisfy a given condition?
- C5: What are all requirements a given domain has?
- C6: What is the cost associated with generating a plan for a given problem?
- C7: How many parameters does a specific action have?
- C8: What planning type a specific planner belongs to?
- C9: What requirements does a given planner support?
- C10: What are the different `types` present in a domain?

4.2 Design

An ontology is a formal and explicit representation of concepts, entities, and their relationships in a particular domain. In this case, ontology is concerned with the domain of automated planning, which refers to the process of generating a sequence of actions to achieve a particular goal within a given set of constraints. The ontology aims to provide a structured framework for organizing and integrating knowledge about this domain, which can be useful in various applications, such as designing planning algorithms, extracting best-performing planners given a domain, or learning domain-specific macros.

Figure 1 shows an ontology that aims to encompass the various concepts of automated planning separated into categories of `Domain`, `Problem`, `Plan`, and `Planner`. The ontology for automated planning is composed of 19 distinct classes and 25 object properties. These classes and properties are designed to represent the various elements of the automated planning domain and its associated problems.

Domain The Domain category of the ontology comprises classes that represent the general characteristics of the planning domain. These classes include the Domain `- Requirements`, `Types`, `Predicates`, `Constraints`, and `Actions` that can be used to solve problems in that domain. The classes in the Domain category are designed to provide a structured framework for organizing and integrating knowledge about the problem domain, which can be useful in various applications such as designing planning algorithms, extracting best-performing planners given a domain, or learning domain-specific macros.

Problem The Problem category of the ontology includes classes that represent specific problems within a given domain. These classes are designed to capture the details of a particular problem, such as the `Objects` defined in the problem, which is an instance of the 'types' defined in the planning domain, the `Initial State` of the problem, and the `Goal State` which are a subclass of the parent class `State` which describes the current state of the problem domain.

Plan The Plan category of the ontology includes classes that represent the sequence of actions that must be taken to solve a given problem. The `Plan` class is used to store the knowledge about the plans that planners generate for specific problems. The plan cost for each plan is stored as a data property (non-negative integer) of the `Plan` class. This enables planners to be compared based on the quality of the plans they generate, as well as the cost of those plans.

Planner The Planner category of the ontology includes classes that capture the details of planner performance from previous IPCs. Specifically, `Planning Domain` relevance to a `Planner` is classified based on the percentage of problems they have successfully solved, which is then categorized into three levels of relevance to the planner: *low*, *medium*, and *high*. By incorporating this information into the ontology, planners can be evaluated based on their performance in different problem domains, and more informed decisions can be made about which planners to use for a given problem. In addition, this information can be used to guide the development of new planners and to evaluate their performance against established benchmarks.

4.3 Accessing Planning Ontology

We have taken various measures to ensure that our planning ontology follows the FAIR principles [12] of being Findable, Accessible, Interoperable, and Reusable. To assist users in exploring and utilizing our ontology, we have made it accessible through a persistent URL[1] and our GitHub repository[2]. Our repository contains ontology model files, mapping scripts, and utility scripts that extract information from PDDL domains and problems into intermediary JSON format and add the extracted data as triples using our model ontology, creating a knowledge graph. We provide sample SPARQL queries that address the ontology's competency questions mentioned earlier. Moreover, our ontology documentation, which is accessible through the GitHub repository, provides a comprehensive overview of the ontology's structure, concepts, and relations, including ontology visualization. This documentation serves as a detailed guide for users to comprehend the ontology's applications in the automated planning domain. We also provide the scripts and results from the ontology evaluation, which are presented as use cases of our ontology in later sections, in our repository, along with accompanying documentation.

---

[1] PURL - https://purl.org/ai4s/ontology/planning

[2] https://github.com/BharathMuppasani/AI-Planning-Ontology

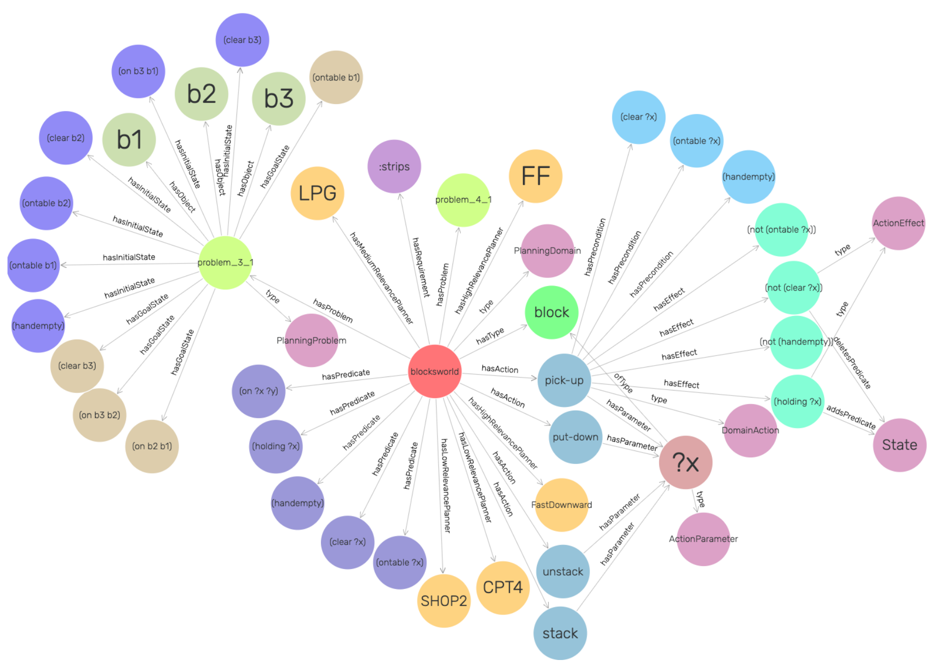

Fig. 2: Knowledge graph representation for blocksworld domain from IPC-2000

5 Usage of Planning Ontology

In the following section, we show the evaluation of a few competency questions and discuss two use cases of our planning ontology.

Evaluation of Competency questions: For the evaluation of the competency questions, we have considered a sample knowledge graph, shown in Figure 2, for `blocksworld` from IPC-2000 domain created using planning ontology shown in Figure 1.

1. C1: What are the different types of planners used in automated planning?
   Question Type: Extracting planner information.
   Sufficiency Condition: There should exist at least one individual for `Planner` class.

S.No	Planner
1	FF
2	FastDownward
3	LPG
4	CPT4
5	SHOP2

Table 1: Results for C1 with knowledge graph in Figure 2

2. C2: What is the relevance of planners in 'blocksworld' domain?
   Question Type: Extracting best planner for a domain.
   Sufficiency Condition: There should exist at least one `Planner` individual having either of the relevance properties with 'blocksworld' individual of `PlanningDomain` class.

S.No	Domain	Relation	Planner
1	blocksworld	hasLowRelevancePlanner	CPT4
2	blocksworld	hasLowRelevancePlanner	SHOP2
3	blocksworld	hasHighRelevancePlanner	FF
4	blocksworld	hasHighRelevancePlanner	FastDownward
5	blocksworld	hasMediumRelevancePlanner	LPG

Table 2: Results for C2 with knowledge graph in Figure 2

3. C3: What are the available actions for 'blocksworld' domain?
   Question Type: Extracting domain information.
   Sufficiency Condition: For the 'blocksworld' individual of `PlanningDomain`, there must be at least one `DomainAction` individual with the relation `hasAction`.

S.No	Domain	Relation	Action
1	blocksworld	hasAction	put-down
2	blocksworld	hasAction	pick-up
3	blocksworld	hasAction	stack
4	blocksworld	hasAction	unstack

Table 3: Results for C3 with knowledge graph in Figure 2

4. C4: Which problems in 'blocksworld' have problems with the goal state of 'b1' being on the table?
   Question Type: Extracting problem information
   Sufficiency Condition: For the 'blocksworld' individual of `PlanningDomain`, there must be at least one `PlanningProblem` individual with the relation `hasProblem` and the problem should have '(ontable b1)' `GoalState`.

S.No	**Domain**	**Relation**	**Problem**
1	blocksworld	hasProblem	problem_3_1

Table 4: Results for C4 with knowledge graph in Figure 2

5. C5: What are all requirements a given domain has?
   Question Type: Extracting domain information
   Sufficiency Condition: For the 'blocksworld' individual of `PlanningDomain`, there must exist at least one `DomainRequirement` individual with the relation `hasRequirement`.

S.No	**Domain**	**Relation**	**Requirement**
1	blocksworld	hasRequirement	:strips

Table 5: Results for C5 with knowledge graph in Figure 2

Usecase 1: Identifying Most Promising Planner - One of the major challenges in the field of artificial intelligence (AI) is the automated selection of the best-performing planner for a given planning domain. This challenge arises due to the vast number of available planners and the diversity of planning domains. To address this challenge, our approach proposes the use of an ontology to represent the features of the planning domain and the capabilities of planners.

The ontology for planning aims to capture the connection between the Planning Domain and the Planner by indicating the relevance of a planner to a specific domain. We made use of data acquired from International Planning Competitions (IPCs) to furnish specific details regarding the relevance of planners. The IPC results provide us with relevant details on the planners that took part in the competition and the domains that were evaluated during that particular year. This information includes specifics on how each planner performed against all the domains that participated.

To show the usage of extracting the most promising planners for a given domain, we have used IPC-2011 data[3] (optimal track). The ontology was populated with data acquired from the IPC-2011, which provided relevant details on the planners that took part in the competition and the domains that were evaluated during IPC-2011. A relevance relation of either *low*, *medium*, or *high* was assigned to each planner based on the percentage of problems they solved

[3] http://www.plg.inf.uc3m.es/ipc2011-deterministic/

Domain	Ontology Policy		Random Policy	
	Avg Expanded	Avg Plan Cost	Avg Expanded	Avg Plan Cost
scanalyzer	**8588**	20	8706	20
elevators	**1471**	52	64541	52
transport	165263	491	**132367**	491
parking*	**367910**	18	488830	17
woodworking	**1988**	211	19844	211
floortile**	283724	54	**2101**	49
barman	**1275078**	90	5816476	90
openstacks	**132956**	4	139857	4
nomystery	1690	13	1690	13
pegsol	**89246**	6	101491	6
visitall	5	4	5	4
tidybot**	**1173**	17	3371	33
parcprinter	541	441374	**417**	441374
sokoban	**9653**	25	156600	25

Table 6: UseCase 1. Lower numbers are better unless indicated. Best result is is in bold.

in a given domain. In this experiment, we consider that the experimental environment has four planners available: Fast Downward Stone Soup 1[4], LM-Cut[4], Merge and Shrink[4], and BJOLP[4]. We evaluate 3 problem instances of each domain from IPC-2011 with 2 policies for selecting planners to generate plans for each of these problem instances -

1. **Random Policy:** To solve each problem instance, this policy selects a random planner from the available planners.
2. **Ontology Policy:** To solve each problem instance, this policy extracts the information on the best planner for the problem domain from the ontology populated with IPC-2011 data.

The evaluation results are presented in Table 6, which provides details on the average number of nodes expanded and the average plan cost for each policy in a given domain. The table provides a comprehensive summary of the performance of different planners in terms of their efficiency and effectiveness. An ideal planner is expected to generate a solution with low values for both these metrics. By comparing the performance of the planners selected using the two policies, we demonstrate the effectiveness of the Ontology Policy in selecting the best-performing planner for a given planning domain. Table 6 presents the results of our evaluation, indicating the average number of nodes expanded and plan cost for each policy in a given domain. The Ontology Policy, designed to select the best-performing planner for a given domain, outperformed the Random Policy in terms of the average number of nodes expanded to find a solution. Moreover, the Random Policy failed to solve problems in the 'parking' (1 out of 3), 'floortile' (2 out of 3), and 'tidybot' (2 out of 3) domains, which highlights the limitations of choosing a planner randomly.

[4] https://www.fast-downward.org/IpcPlanners

Domains	Extracted Action Relations
`blocksworld`	`unstack` precedes `put-down`, `pick-up` precedes `stack`, `put-down` precedes `unstack`, `stack` precedes `pick-up`, `unstack` precedes `stack`, `put-down` precedes `pick-up`, `stack` precedes `unstack`
`driverlog`	`drive-truck` precedes `unload-truck`, `drive-truck` precedes `load-truck`, `board-truck` precedes `drive-truck`, `walk` precedes `board-truck`
`grippers`	`pick` precedes `move`, `move` precedes `drop`

Table 7: Extracted action relations, ordered based on their frequency, for domains `blocksworld`, `driverlog`, and `grippers`.

Usecase 2: Macro Selection for Improving Planner Performance - While automated planning has been successful in many domains, it can be computationally expensive, especially for complex problems. One approach to improve efficiency is by using macro-operators, which are sequences of primitive actions that can be executed as a single step. However, identifying useful macro-operators manually can be time-consuming and challenging. Authors in [3] introduce a novel method for improving the efficiency of planners by generating macro-operators. The proposed approach involves analyzing the inter-dependencies between actions in plans and extracting macro-operators that can replace primitive actions without losing the completeness of the problem domain. The soundness and complexity of the method are assessed and compared to other existing techniques. The paper asserts that the generated macro-operators are valuable and can be seamlessly integrated into planning domains without losing the completeness of the problem.

Based on the ontology depicted in Figure 1, we extract macro-operators that can enhance the efficiency of planners. To demonstrate this, we have considered three different domains: `blocksworld`(bw), `driverlog`(dl), and `grippers`(gr), presented in IPC-2000, 2002, and 1998 respectively. We initially developed a knowledge graph using the ontology represented in Figure 1 for the three domains of interest. Subsequently, we employed a SPARQL query to retrieve the stored plans for these domains. We then examined these plans to identify the sequences of action pairs and ranked them based on their frequency of occurrence. To improve the effectiveness of this technique, it is essential to consider both the frequency of occurrence of action pairs and the properties of the domain. Specifically, the precondition and effect of actions should be analyzed to ensure that the first action leads to the precondition of the second action in the pair. We employed another SPARQL query to extract the preconditions and effects associated with each of these actions. We analyzed the resulting action pairs to verify their validity of occurrence, thereby filtering out pairs that did not have a combined effect. The results of this extraction process are shown in Table 7. These action relations are stored back into the knowledge graph in the `MacroAction` class and can be utilized by planners to enhance their efficiency.

Table 8 shows the comparison of a planner performance given the original domain and macros-enabled version of the domain. For this evaluation, we have

Domain	Original Domain			Domain With Macros		
	Avg. Exp.	Avg. Eval.	Avg. Gen.	Avg. Exp.	Avg. Eval.	Avg. Gen.
blocksworld	20219	59090	106321	18	310	359
gripper	2672	10660	30871	510	3974	11468
driverlog	3753	17849	45753	14888	720008	209760

Table 8: Comparison of planner performance between original and macro-enabled versions of three planning domains, showing the average number of nodes expanded, evaluated, and generated.

considered the FastDownward planner [7] with LM-Cut Heuristic [8] to generate plans for 20 problems of varying complexities for each domain. We evaluate the performance of each domain based on the average number of nodes expanded, evaluated, and generated to find a solution. This study demonstrates that macro operators can enhance the planner performance in most of the domains tested, with the exception of the `driverlog` domain. In this domain, the planner performs worse when macro operators are included, as they increase the average number of nodes expanded, evaluated, and generated. This is due to the fact that the macro operators introduce more actions to the domain, which increases the branching factor and challenges the heuristic to select the optimal action at each step. Hence, the applicability of macro operators depends on the features of the domain and the planner. Macro operators can facilitate the planning process by decreasing the search depth, but they can also hinder it by increasing the search width. A potential improvement is to use a more informative heuristic that guides the planner to choose the best action at each step.

6 Conclusion

In this work, we build and share a planning ontology that provides a structured representation of concepts and relations for planning, allowing for efficient extraction of domain, problem, and planner properties. The ontology's practical utility is demonstrated in identifying the best-performing planner for a given domain and extracting macro operators using plan statistics and domain properties. Standardized benchmarks from IPC domains and planners offer an objective and consistent approach to evaluating planner performance, enabling rigorous comparisons in different domains to identify the most suitable planner. The planning ontology can aid researchers and practitioners in automated planning, and its use can simplify planning tasks and boost efficiency. As the field of AI planning continues to evolve, planning ontology can play a crucial role in advancing the state-of-the-art while leveraging the past.

Future work could explore the use of mixed reasoning strategy with both ontologies (top-down) and LLMs (bottom-up) knowledge [9]. For instance, one could leverage the planning ontology in the context of LLMs, which have recently shown promise for automated planning [10]. Moreover, the application of this mixed reasoning approach could be extended to complex domains, such as multi-agent systems, where coordinating actions between multiple agents is crucial.

References

1. Babli, M., Onaindia, E., Marzal, E.: Extending planning knowledge using ontologies for goal opportunities. arXiv preprint arXiv:1904.03606 (2019)
2. Bihl, T.J., Cox, C., Machin, T.: Towards a taxonomy of planning for autonomous systems. In: 2019 IEEE National Aerospace and Electronics Conference (NAECON). pp. 74–79 (2019). https://doi.org/10.1109/NAECON46414.2019.9058324
3. Chrpa, L.: Generation of macro-operators via investigation of action dependencies in plans. The Knowledge Engineering Review **25**(3), 281–297 (2010)
4. Gayathri, R., Uma, V.: Ontology based knowledge representation technique, domain modeling languages and planners for robotic path planning: A survey. ICT Express **4**(2), 69–74 (2018)
5. Gil, Y., Blythe, J.: Planet: A shareable and reusable ontology for representing plans. In: AAAI Workshop on Representational Issues for Real-world Planning Systems. Website: https://www.isi.edu/b̃lythe/planet/. vol. 114 (2000)
6. Guarino, N., Oberle, D., Staab, S.: What is an ontology? Handbook on ontologies pp. 1–17 (2009)
7. Helmert, M.: The fast downward planning system. Journal of Artificial Intelligence Research **26**, 191–246 (2006)
8. Helmert, M., Domshlak, C.: Lm-cut: Optimal planning with the landmark-cut heuristic. Seventh international planning competition (IPC 2011), deterministic part pp. 103–105 (2011)
9. Mittal, S., Joshi, A., Finin, T.W.: Thinking, fast and slow: Combining vector spaces and knowledge graphs. ArXiv **abs/1708.03310** (2017)
10. Pallagani, V., Muppasani, B., Murugesan, K., Rossi, F., Horesh, L., Srivastava, B., Fabiano, F., Loreggia, A.: Plansformer: Generating symbolic plans using transformers. In: On Arxiv at: https://arxiv.org/abs/2212.08681 (2022)
11. Valente, A., Russ, T., MacGregor, R., Swartout, W.: Building and (re) using an ontology of air campaign planning. IEEE Intelligent Systems and their Applications **14**(1), 27–36 (1999)
12. Wilkinson, M.D., Dumontier, M., Aalbersberg, I.J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.W., da Silva Santos, L.B., Bourne, P.E., et al.: The fair guiding principles for scientific data management and stewardship. Scientific data **3** (2016)
13. Žáková, M., Křemen, P., Železnỳ, F., Lavrač, N.: Automating knowledge discovery workflow composition through ontology-based planning. IEEE Transactions on Automation Science and Engineering **8**(2), 253–264 (2010)