

Appendix A Supplementary Material: Demonstration of Usecase

A.1 Usecase 1: Identifying Most Promising Planner

Q: "Which is the best planner for blocksworld domain?"

```
SELECT ?planner
WHERE {
  {
    po:blocksworld po:hasHighRelevance ?highRelevancePlanner
  }
  UNION
  {
    FILTER NOT EXISTS { po:blocksworld po:hasHighRelevance ?
      highRelevancePlanner }
    po:blocksworld po:hasMediumRelevance ?mediumRelevancePlanner
  }
  UNION
  {
    FILTER NOT EXISTS { po:blocksworld po:hasHighRelevance ?
      highRelevancePlanner }
    FILTER NOT EXISTS { po:blocksworld po:hasMediumRelevance ?
      mediumRelevancePlanner }
    po:blocksworld po:hasLowRelevance ?lowRelevancePlanner
  }
}
BIND(COALESCE(?highRelevancePlanner, ?mediumRelevancePlanner, ?
  lowRelevancePlanner) AS ?planner)
LIMIT 1
```

A.2 Usecase 2: Explanation Generation

Plan Explanation: *Natural Language Summary*

The query below extracts the parameters, precondition labels, and effect labels of a specific action (in this case, 'pick-up'). By mapping these technical details to natural language templates, we can generate human-readable explanations illuminating the planner's reasoning at each stage.

Q: "Can you describe the 'pick-up' action in simple terms?"

```
SELECT ?param ?preconditionLabel ?effectLabel
WHERE {
  po:pick-up po:hasParameter ?param ;
    po:hasPrecondition ?preCondition ;
    po:hasEffect ?effect .
  ?preCondition rdf:label ?preconditionLabel .
  ?effect rdf:label ?effectLabel .
}
```

Explaining Non-Selection of Specific Actions:

The query below allows us to extract and compare the preconditions and effects of two different actions. By analyzing this information, we can generate explanations that highlight why one action was preferred over another. This might involve identifying unsatisfied preconditions in the resulting state, comparing the effects in relation to

Table A1 Demonstrating the effectiveness of two different policies employed to choose a planner for problem-solving. This table compares the average nodes expanded (Avg. Exp.) during the search and the resulting plan cost (Avg. Cost) for two policies: the Ontology Policy and the Random (Baseline) Policy. It is observed that the Ontology Policy performs better than the baseline in almost all the domains and problem instances considered.

Domain	Ontology Policy		Random Policy	
	Avg. Exp. $\pm$ Std.	Avg. Cost $\pm$ Std.	Avg. Exp. $\pm$ Std.	Avg. Cost $\pm$ Std.
ipc 1998 - gripper	3.88k$\pm$4.57k	17 $\pm$ 4	4.57k $\pm$ 5.07k	17 $\pm$ 4
ipc 1998 - grid*	19.8k$\pm$27.9k	20 $\pm$ 6	25.7k $\pm$ 36.3k	20 $\pm$ 6
ipc 2000 - blocks	11$\pm$5	7 $\pm$ 1	15 $\pm$ 9	7 $\pm$ 1
ipc 2000 - elevator	10 $\pm$ 7	4 $\pm$ 1	10 $\pm$ 7	4 $\pm$ 1
ipc 2002 - depots	161k$\pm$228k	18 $\pm$ 8	419k $\pm$ 591k	18 $\pm$ 8
ipc 2002 - driverlog	828 $\pm$ 711	13 $\pm$ 4	828 $\pm$ 711	13 $\pm$ 4
ipc 2004 - pipesworld	3.65k$\pm$4.20k	8 $\pm$ 2	11.7k $\pm$ 14.7k	8 $\pm$ 2
ipc 2004 - satellite	348$\pm$485	8 $\pm$ 6	93.2k $\pm$ 132k	8 $\pm$ 6
ipc 2006 - rovers	26$\pm$11	9 $\pm$ 1	153 $\pm$ 199	9 $\pm$ 1
ipc 2006 - storage	7$\pm$5	4 $\pm$ 2	20 $\pm$ 22	4 $\pm$ 2
ipc 2008 - openstacks	865 $\pm$ 611	2 $\pm$ 0	865 $\pm$ 611	2 $\pm$ 0
ipc 2008 - sokoban	5.06k$\pm$6.92k	16 $\pm$ 8	105k $\pm$ 147k	16 $\pm$ 8
ipc 2011 - scanalyzer	8.6k$\pm$1.2k	20 $\pm$ 3	8.7k $\pm$ 1.1k	20 $\pm$ 3
ipc 2011 - elevators	1.5k$\pm$2.5k	52 $\pm$ 5	65k $\pm$ 12k	52 $\pm$ 5
ipc 2011 - transport	170k $\pm$ 200k	491 $\pm$ 40	130k$\pm$150k	491 $\pm$ 40
ipc 2011 - parking*	370k$\pm$30k	18 $\pm$ 2	490k $\pm$ 41k	17 $\pm$ 2
ipc 2011 - woodworking	2.0k$\pm$0.11k	211 $\pm$ 10	20k $\pm$ 2.0k	211 $\pm$ 10
ipc 2011 - floortile**	280k $\pm$ 35k	54 $\pm$ 4	2.1k$\pm$0.15k	49 $\pm$ 5
ipc 2011 - barman	1.3M$\pm$0.1M	90 $\pm$ 8	5.8M $\pm$ 450k	90 $\pm$ 8
ipc 2011 - openstacks	130k$\pm$11k	4 $\pm$ 0	140k $\pm$ 12k	4 $\pm$ 0
ipc 2011 - nomystery	1.7k $\pm$ 0.2k	13 $\pm$ 1	1.7k $\pm$ 0.2k	13 $\pm$ 1
ipc 2011 - pegsol	89k$\pm$7k	6 $\pm$ 0	100k $\pm$ 8k	6 $\pm$ 0
ipc 2011 - visitall	5.0 $\pm$ 0.3	4 $\pm$ 0	5.0 $\pm$ 0.3	4 $\pm$ 0
ipc 2011 - tidybot**	1.2k$\pm$0.1k	17 $\pm$ 2	3.4k $\pm$ 0.25k	33 $\pm$ 3
ipc 2011 - parcuprinter	540 $\pm$ 45	441k $\pm$ 5.2k	420$\pm$35	441k $\pm$ 5.2k
ipc 2011 - sokoban	9.7k$\pm$0.8k	25 $\pm$ 2	160k $\pm$ 12k	25 $\pm$ 2
ipc 2014 - floor-tile**	284k$\pm$275k	54 $\pm$ 6	27.7M	49
ipc 2014 - parking	50.7k$\pm$38.3k	8 $\pm$ 0	202k $\pm$ 100k	8 $\pm$ 0
ipc 2018 - organic-synthesis*	2 $\pm$ 1	1 $\pm$ 0	2 $\pm$ 1	1 $\pm$ 0

the goal state, or explaining how the chosen action better optimizes certain metrics (e.g., plan length, and resource usage).

Q: *"Why did you perform (pick-up b3) instead of unstack in step 3?"*

```
SELECT ?action ?preconditionLabel ?effectLabel
WHERE {
  ?action po:hasParameter ?param .
  FILTER (?action IN (po:pick-up, po:unstack))

  ?action po:hasPrecondition ?preCondition .
  ?preCondition rdf:label ?preconditionLabel .

  ?action po:hasEffect ?effect .
  ?effect rdf:label ?effectLabel .
}
ORDER BY ?action
```

Appendix B SPAQRL queries for Competency Question

For the evaluation of the competency questions, we have considered a sample knowledge graph, shown in Figure B1, for blocksworld from IPC-2000 domain created using planning ontology shown in Figure 2.

C1: *"What are the different types of planners used in automated planning?"* Identifies and lists the various types of planners utilized in the field of automated planning.

```
PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
```

```
SELECT DISTINCT ?planner
WHERE {
  ?planner a plan-ontology:planner.
}
```

C2: *"What is the relevance of a planner in a given problem domain?"*

Explores the importance and applicability of different planners within a specific problem domain.

```
PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
```

```
SELECT DISTINCT ?domain ?relevance ?planner
WHERE {
  ?domain a plan-ontology:domain;
    rdfs:label "caldera".
  ?planner a plan-ontology:planner.
  ?domain ?relevance ?planner.
}
```

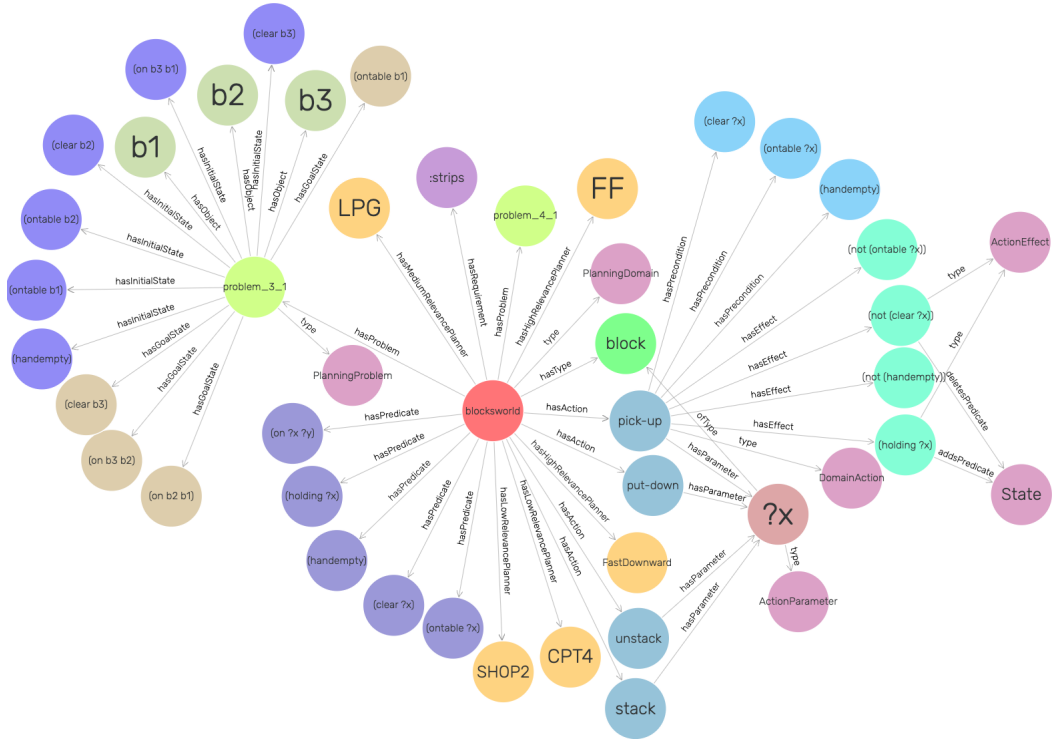


Fig. B1 Knowledge graph representation for blocksworld domain from IPC-2000

C3: *"What are the available actions for a given domain?"* Provides a list of actions that can be performed within a specified domain.

PREFIX plan-ontology: <<https://purl.org/ai4s/ontology/planning#>>
 PREFIX rdfs: <<http://www.w3.org/2000/01/rdf-schema#>>

```
SELECT DISTINCT ?domain ?action
WHERE {
  ?domain a plan-ontology:domain;
  rdfs:label "caldera".
  ?domain plan-ontology:hasMove ?action.
}
```

C4: *"What problems in a domain satisfy a given condition?"* Identifies problems within a domain that meet certain specified conditions.

PREFIX plan-ontology: <<https://purl.org/ai4s/ontology/planning#>>
 PREFIX rdfs: <<http://www.w3.org/2000/01/rdf-schema#>>

```
SELECT DISTINCT ?problem
WHERE {
  ?domain a plan-ontology:domain;
```

```

        rdfs:label "caldera".
    ?problem a plan-ontology:problem.
    ?domain plan-ontology:hasProblem ?problem.
    ?problem plan-ontology:hasGoalState ?condition.
    FILTER(?condition = "specified_condition")
}

```

C5: *"What are all requirements a given domain has?"* Enumerates all the prerequisites or conditions necessary within a particular domain.

```

PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```

```

SELECT DISTINCT ?domain ?requirement
WHERE {
    ?domain a plan-ontology:domain;
        rdfs:label "caldera".
    ?domain plan-ontology:hasRequirement ?requirement.
}

```

C6: *"What is the cost associated with generating a plan for a given problem?"* Determines the cost involved in creating a plan to solve a specified problem.

```

PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```

```

SELECT DISTINCT ?plan ?cost
WHERE {
    ?problem a plan-ontology:problem;
        rdfs:label "problem-01".
    ?problem plan-ontology:hasPlan ?plan.
    ?plan plan-ontology:hasCost ?cost.
}

```

C7: *"How many parameters does a specific action have?"* Counts the number of parameters required for a particular action within a domain.

```

PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```

```

SELECT (COUNT(?parameter) AS ?parameterCount)
WHERE {
    ?action a plan-ontology:action;
        rdfs:label "get_domain". # action of domain caldera
    ?action plan-ontology:hasParameter ?parameter.
}

```

C8: *"What planning type a specific planner belongs to?"* Classifies a given planner into its respective planning type.

```

PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```

```

SELECT DISTINCT ?planner ?planningType
WHERE {
  ?planner a plan-ontology:planner;
           rdfs:label "Delfi1".
  ?planner plan-ontology:ofPlannerType ?planningType.
}

```

C9: *"What requirements does a given planner support?"* Lists the requirements that a specific planner is capable of addressing.

```

PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```

```

SELECT DISTINCT ?planner ?requirement
WHERE {
  ?planner a plan-ontology:planner;
           rdfs:label "Delfi1".
  ?planner plan-ontology:solvesRequirement ?requirement.
}

```

C10: *"What are the different parameter types present in a domain?"* Identifies and lists the various types of parameters present within a specified domain.

```

PREFIX plan-ontology: <https://purl.org/ai4s/ontology/planning#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

```

```

SELECT DISTINCT ?domain ?parameterType
WHERE {
  ?domain a plan-ontology:domain;
           rdfs:label "caldera".
  ?domain plan-ontology:hasParameterType ?parameterType.
}

```