Expressivity of Planning with Horn Description Logic Ontologies: Supplementary Material

Submission #10183

Benchmark Description

Our collection of benchmarks consists of a total of 235 instances adapted from the publicly available DL-Lite eKAB benchmark collection (Borgwardt et al. 2021) as well as newly developed high expressivity domains. The benchmarks and the compiler are available in the supplementary material. Each problem instance has two representations: the Horn-$\mathcal{SHIQ}$ eKAB task encoding with an ontology written in Turtle[1] and its compilation into PDDL.

Adapted DL-Lite eKAB benchmarks: We translated the original benchmarks into equivalent representations in our Horn-$\mathcal{SHIQ}$ eKAB task encoding. Detailed descriptions of these domains are available online. In short,

- in Robot (Calvanese et al. 2016), a robot is positioned on a grid without knowing its position and the goal is to reach a target cell. The ontology describes relations between rows and columns.
- The goal of TaskAssign, inspired by (Calvanese et al. 2016), is to hire two electronic engineers for a company, while the ontology describes relations between different job positions.
- The Elevator and Cats benchmarks are inspired by standard planning benchmarks. In the Cats domain, there is a set of packages that contain either cats or bombs and the task is to disarm all bombs. An elevator in the Elevator benchmark can move up and down between floors to serve passengers according to their origins and destinations.
- Both the VTA and TPSA benchmarks are adaptations from older work on semantic web-service composition (Hoffmann et al. 2008). VTA-Roles is a more complex variant of VTA.

High Expressivity Domains:

- Drones models a complex 2D drone navigation problem, in which drones need to be moved while avoiding certain situations; the latter is given by ontology reasoning, involving Horn concept inclusions with qualified existential restrictions occurring negatively and symmetric roles. Grid cells are occupied with different objects like Humans or Trees or weather conditions like LowVisibility or Rain. There is a set of Drones randomly placed on the board. Depending on the distances to other objects, a Drone can enter a critical state (defined by the ontology). The goal is to move the drones such that no two drones in a critical state are next to each other. In the benchmark, instances vary in the board size and the number of drones. We have chosen the instances such that some of them remain hard for the planner to solve. The compilation itself is always very fast.
- Queens generalizes the eight queens puzzle from chess to variable numbers of board sizes, $n \in \{5, \ldots, 10\}$, and queens, $m \in \{n-4, \ldots, n\}$. In the initial state, queens are placed randomly and the ontology contains a symmetric, transitive role to describe which queen movements are legal. Similarly to Drones, the planner must find a sequence of legal moves such that no two queens threaten each other.
- RobotConj is a redesign of Robot, moving complexity from action descriptions into the ontology. The original Robot benchmark encoded static knowledge about 2D-grid cell adjacency in the action descriptions, which via the use of Horn clauses can be encoded much more naturally directly in the ontology. More precisely, in a slightly simplified notation, the action MoveDown contains the two redundant conditional effects

$$\begin{array}{r}\text{(when (and (AboveOf1 ?x) (BelowOf2 ?x))} \\ \text{(Row0 ?x)),}\end{array} \tag{1}$$

  which one can read as "if the robot is above or in row 1 and below row 2, then move the robot to row 0", and

$$\begin{array}{r}\text{(when (Row1 ?x))} \\ \text{(Row0 ?x)),}\end{array} \tag{2}$$

  i.e. "if the robot position is in row 1, then move the robot to row 0". However, encoding the static knowledge that AboveOf1 and BelowOf2 imply Row1 is beyond DL-Lite. Moreover, the actions MoveUp, MoveLeft, and MoveRight have similar redundant effects. For RobotConj we have simplified these descriptions by using axioms like AboveOf1$\sqcap$BelowOf2 $\sqsubseteq$ Row1, which allows us to get rid of (1).

[1] https://www.w3.org/TR/turtle/

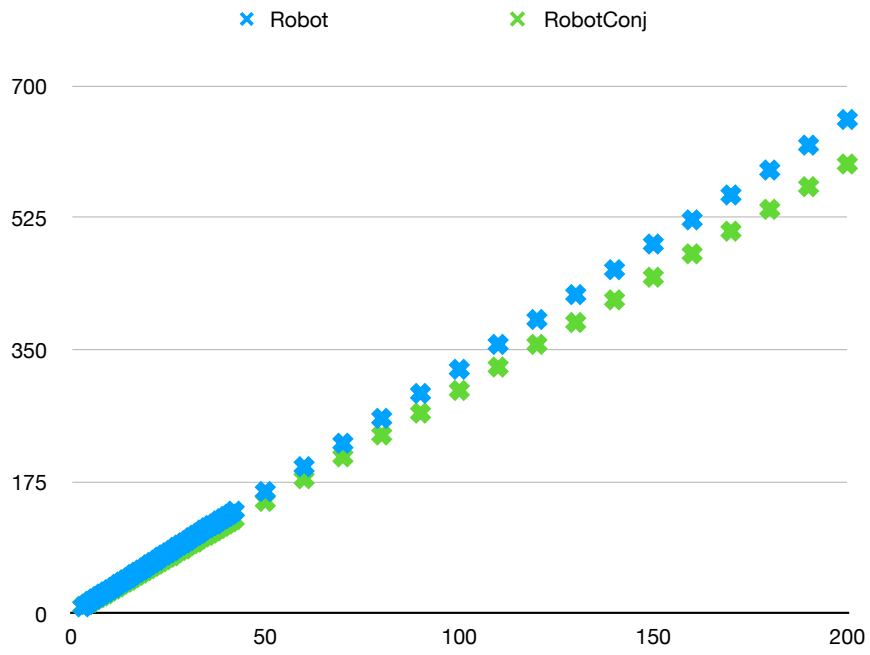

Figure 1: Sizes of summarized domain and ontology files (Kb).

Robots Comparison

In this section we additionally show computational benefits of RobotConj.

We use Robot and RobotConj to analyze how our compilation performs as a function of ontology size. In both domains, the ontology size is directly proportional to size of the grid, see Figure 1. Figures 1-3 depict the results for 56 instances of each domain, obtained by scaling the grid from 3×3 to 200×200 (see the horizontal axes of the figures). The increased complexity of RobotConj's ontology does not affect the performance of the compilation. Compilation of RobotConj is actually slightly, but consistently, faster than Robot. Even that largest tested instance could be compiled within less than 20 seconds, attesting the practicability of our approach. The simpler action description in RobotConj turned out advantageous for the planner, whose run time could be reduced considerably. We have also observed differences between the versions with respect to planner time on Figure 3 measured with FD[2] 20.06: *RobotConj* has performed better. All experiments were run on a computer with an Intel Core i5-4590 CPU@3.30GHz processor, and run time and memory cutoffs of 600 seconds and 8 GBs.

References

Borgwardt, S.; Hoffmann, J.; Kovtunova, A.; and Steinmetz, M. 2021. Making DL-Lite Planning Practical. In Bienvenu, M.; and Lakemeyer, G., eds., *Proc. of the 18th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'21)*. IJCAI. Short paper. To appear.

Calvanese, D.; Montali, M.; Patrizi, F.; and Stawowy, M. 2016. Plan Synthesis for Knowledge and Action Bases. In Kambhampati, S., ed., *Proc. of the 25th Int. Joint Conf. on Artificial Intelligence (IJCAI'16)*, 1022–1029. AAAI Press.

Hoffmann, J.; Weber, I.; Scicluna, J.; Kacmarek, T.; and Ankolekar, A. 2008. Combining Scalability and Expressivity in the Automatic Composition of Semantic Web Services. In *8th International Conference on Web Engineering (ICWE'08)*.

[2]http://www.fast-downward.org/Releases/20.06

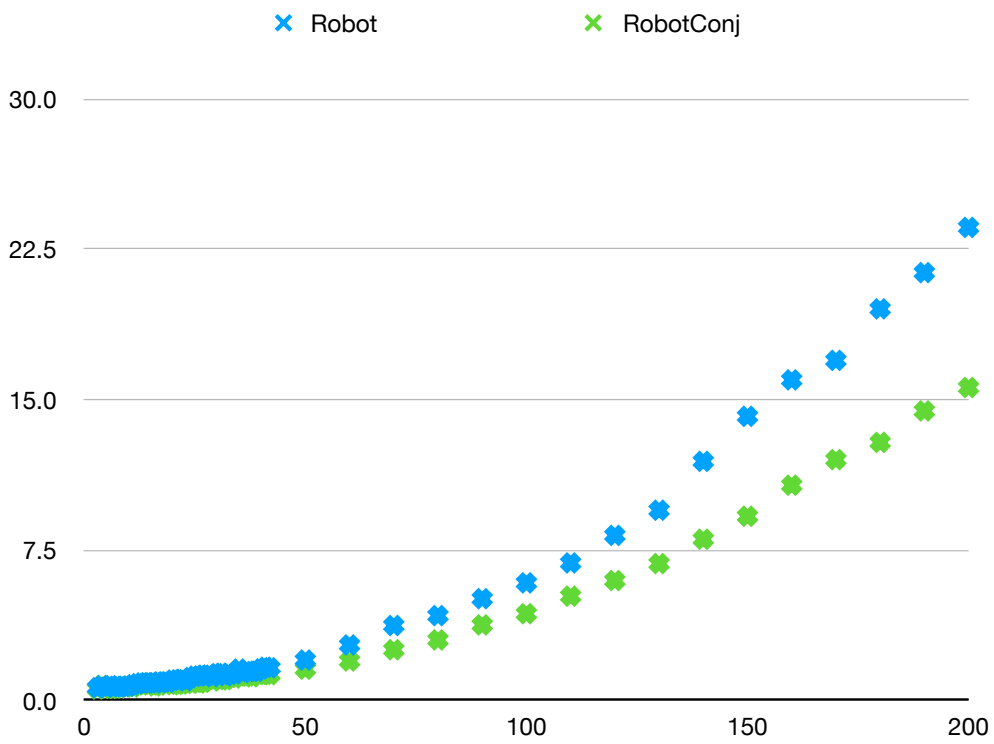

Figure 2: Compilation time (s).

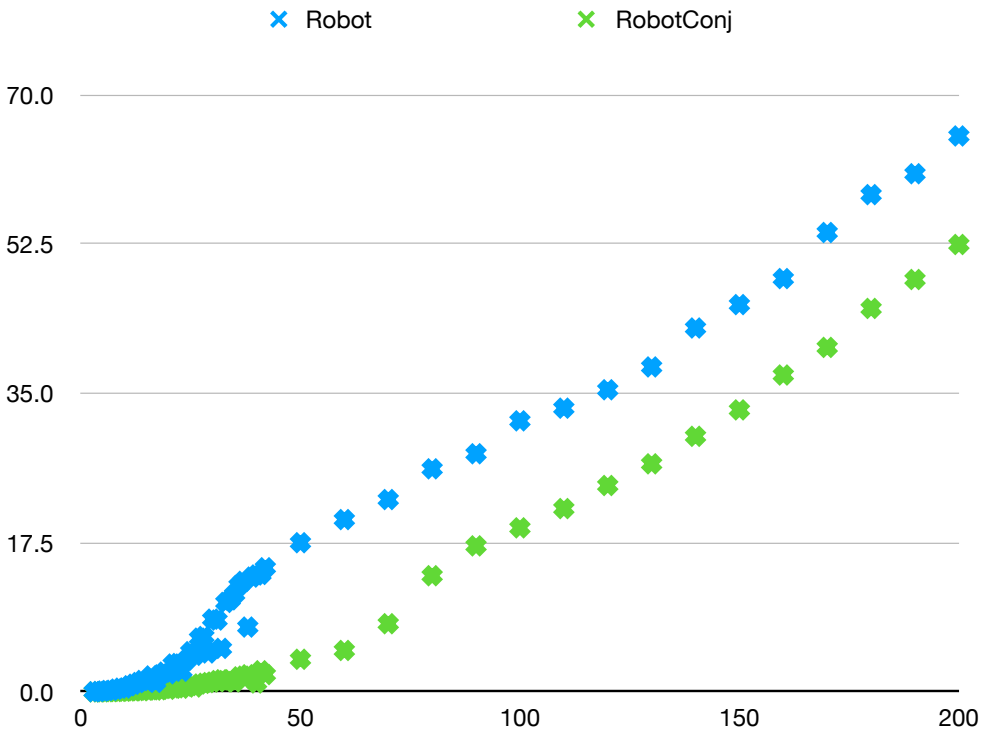

Figure 3: Planner time (s).