Expressivity of Planning with Horn Description Logic Ontologies: Appendix

Submission #10183

A Proof of Theorem 2

We first describe the logic Horn-$\mathcal{ALCHOIQ}$ in more detail. We use classical DL terms here, i.e. unary predicates are called *concept names*, binary predicates are *role names*, objects/constants are *individual names* and states are *ABoxes*. We assume that all axioms in Horn-$\mathcal{ALCHOIQ}$ TBoxes are in *normal form* (Carral, Dragoste, and Krötzsch 2018), i.e. they have one of the following shapes, where $C, C_1, \dots, C_n, D$ are concept names, r, s are role names, and a is an individual name:

(i) $C_1 \sqcap \dots \sqcap C_n \sqsubseteq D$
(ii) $C \sqsubseteq \exists r.D$
(iii) $\exists r.C \sqsubseteq D$
(iv) $C \sqsubseteq \leq 1 r.D$
(v) $C \sqsubseteq \{a\}$
(vi) $r \sqsubseteq s$
(vii) $\{a\} \sqsubseteq C$

We added the last axiom type here since our goal is a TBox rewriting that is independent of the ABox, i.e. we need to distinguish the TBox axiom $\{a\} \sqsubseteq C$ from the (equivalent) ABox fact $C(a)$. A first-order interpretation $\mathcal{I}$ satisfies these axioms if it satisfies the sentences

(i) $\forall x.C_1(x) \wedge \dots \wedge C_n(x) \rightarrow D(x)$,
(ii) $\forall x.C(x) \rightarrow \exists y.r(x,y) \wedge D(y)$,
(iii) $\forall x,y.r(x,y) \wedge C(y) \rightarrow D(x)$,
(iv) $\forall x,y,z.C(x) \wedge r(x,y) \wedge D(y) \wedge r(x,z) \wedge D(z) \rightarrow y = z$,
(v) $\forall x.C(x) \rightarrow x = a$,
(vi) $\forall x,y.r(x,y) \rightarrow s(x,y)$, or
(vii) $C(a)$,

respectively. Additionally, there is a bijective and irreflexive function $\cdot^-$ on the set of role names such that $r^{--} = r$ and $\forall x,y.r(x,y) \leftrightarrow (r^-)(y,x)$ is required to hold in all models of a TBox, for all role names r (r^- is called the *inverse* of r).

Following Ortiz, Rudolph, and Šimkus (2010), we will use *Datalog*$^{S,\neg}$ rules to encode reasoning in Horn-$\mathcal{ALCHOIQ}$. Datalog$^{S,\neg}$ extends Datalog$^\neg$ rules by introducing fixed *set sorts* 2^C, where C is a set of constants. Set terms of sort 2^C are built inductively from the constructors $\{c_1, \dots, c_n\}$ and $t_1 \cup t_2$, where $c_1, \dots, c_n \in C$ and t_1, t_2 are set terms of this sort. Every predicate has an associated *sort function* that assigns to each position a unique sort, i.e. either the (normal) *element sort*, or one of the fixed set sorts. This means that the positions of this predicates always accept only terms of the associated sort. The semantics of the resulting (stratified) Datalog$^{S,\neg}$ rules is intuitive (Ortiz, Rudolph, and Šimkus 2010).

To make it easier to compare with the existing constructions (Ortiz, Rudolph, and Šimkus 2010; Carral, Dragoste, and Krötzsch 2018), in the following we write Datalog$^{S,\neg}$ rules with $\rightarrow$ instead of $\leftarrow$. We also use rules with conjunctions of atoms in the head (instead of only one atom).

Encoding Instance Queries

In the following, let $\mathcal{T}$ be a Horn-$\mathcal{ALCHOIQ}$ TBox in normal form, and **C/R/I** denote the sets of concept names/role names/individual names in $\mathcal{T}$. In addition to the individuals in $\mathbf{I}$, the construction by Carral, Dragoste, and Krötzsch (2018) uses artificial individuals of the form t_X, where X is a set of concept names, and new predicates R that represent a set of role names that have to be satisfied at the same time. We represent such R and X using sets of the sorts $2^{\mathbf{R}}$ and $2^{\mathbf{C} \cup \mathbf{I}}$, respectively. The latter includes named individuals since we want to treat named and anonymous individuals using the same predicates. However, named individuals a will only appear as singleton sets $\{a\}$. Formally, we are not allowed to treat individuals from the ABox in this way, because then the set sorts (which need to be fixed) would depend on the ABox. We nevertheless do this in the following and at the end of this section describe how to translate these Datalog$^{S,\neg}$ rules into Datalog$^\neg$ rules that are ABox-independent.

Instead of an atom $C(x)$ (Carral, Dragoste, and Krötzsch 2018), we now use atoms of the form $\mathsf{concept}(C, X)$, where $C \in \mathbf{C}$ is treated as a new constant and X is a set as described above (i.e. either a set of concept names or a singleton set containing an individual name). Likewise, role atoms $r(x,y)$ and $R(x,y)$ are transformed into $\mathsf{role}(r, X, Y)$ and $\mathsf{roles}(R, X, Y)$, respectively. Additionally, we distinguish sets $X \subseteq \mathbf{C}$ from sets $\{a\}$ with $a \in \mathbf{I}$ by the predicate anon, which is populated by the following rules, for all $C \in \mathbf{C}$:

$$\mathsf{anon}(\{C\}) \tag{1}$$

$$\mathsf{anon}(X) \rightarrow \mathsf{anon}(X \cup \{C\}) \tag{2}$$

We also compute the set of inverse roles of a set R, for all $r \in \mathbf{R}$ (Ortiz, Rudolph, and Šimkus 2010):

$$\mathsf{inv}(\{r\}, \{r^-\}) \tag{3}$$

$$\mathsf{inv}(R, R^-) \to \mathsf{inv}(R \cup \{r\}, R^- \cup \{r^-\}) \tag{4}$$

Additionally, the original rewriting uses atoms of the form $\mathsf{n}(x)$ and $x \approx y$, which we simply adapt to $\mathsf{n}(X)$ and $X \approx Y$, where X, Y are as described above. We also add a predicate ind, which identifies all individual names from the ABox in order to identify query answers in the end. This predicate represents a subset of the predicate n, which will also contain anonymous individuals X that are inferred to represent a unique element in every interpretation.

The original Datalog rewriting starts with several auxiliary rules, which we translate below into their modified Datalog$^{S,\neg}$ form, for all $C \in \mathbf{C}$ and $r \in \mathbf{R}$ (Carral, Dragoste, and Krötzsch 2018, Figure 2):

$$\mathsf{concept}(C, X) \to \mathsf{concept}(\top, X) \tag{5}$$

$$\mathsf{role}(r, X, Y) \to \mathsf{concept}(\top, X) \wedge \mathsf{concept}(\top, Y) \tag{6}$$

$$\mathsf{roles}(R, X, Y) \wedge \mathsf{n}(Y) \wedge \mathsf{inv}(R, R^-) \to \mathsf{roles}(R^-, Y, X) \tag{7}$$

$$\mathsf{roles}(R, X, Y) \wedge r \in R \to \mathsf{role}(r, X, Y) \tag{8}$$

$$C(x) \to \mathsf{concept}(C, \{x\}) \wedge \mathsf{ind}(\{x\}) \tag{9}$$

$$r(x, y) \to \mathsf{role}(r, \{x\}, \{y\}) \wedge \mathsf{ind}(\{x\}) \wedge \mathsf{ind}(\{y\}) \tag{10}$$

$$\mathsf{ind}(X) \to \mathsf{n}(X) \tag{11}$$

$$X \approx Y \to Y \approx X \tag{12}$$

$$X \approx Y \wedge Y \approx Z \to X \approx Z \tag{13}$$

$$\mathsf{concept}(C, X) \wedge X \approx Y \to \mathsf{concept}(C, Y) \tag{14}$$

$$\mathsf{n}(X) \wedge X \approx Y \to \mathsf{n}(Y) \tag{15}$$

$$\mathsf{roles}(R, X, Y) \wedge X \approx Z \to \mathsf{roles}(R, Z, Y) \tag{16}$$

$$\mathsf{roles}(R, X, Y) \wedge Y \approx Z \to \mathsf{roles}(R, X, Z) \tag{17}$$

Rules (9) and (10) were further modified from their originals to convert the ABox predicates into our Datalog$^{S,\neg}$ syntax in an ABox-independent way. In a slight abuse of notation, on the left-hand side of Rule (9), C is treated as a concept name, and on the right-hand side C is treated as a constant (similarly for Rule (10)).

The axioms of the TBox $\mathcal{T}$ are translated into the following Datalog$^{S,\neg}$ rules (Carral, Dragoste, and Krötzsch 2018, Figures 3 and 4). For every axiom of the form (i):

$$\mathsf{concept}(C_1, X) \wedge \cdots \wedge \mathsf{concept}(C_n, X) \to \mathsf{concept}(D, X) \tag{18}$$

For axioms of the form (ii):

$$\mathsf{concept}(C, X) \to \mathsf{role}(r, X, \{D\}) \tag{19}$$

For axiom type (iii):

$$\mathsf{role}(r, X, Y) \wedge \mathsf{concept}(C, Y) \to \mathsf{concept}(D, X) \tag{20}$$

$$\mathsf{concept}(C, X) \wedge \mathsf{roles}(R^-, X, Y) \wedge r \in R \wedge \mathsf{inv}(R, R^-) \wedge \mathsf{anon}(Y) \to \mathsf{roles}(R^-, X, Y \cup \{D\}) \tag{21}$$

Axiom type (iv) requires more complex rules:

$$\begin{aligned} \mathsf{concept}(D, Y) \wedge \mathsf{role}(r^-, Y, X) \wedge \\ \mathsf{concept}(C, X) \wedge \mathsf{role}(r, X, Z) \wedge \mathsf{concept}(D, Z) \wedge \mathsf{n}(Z) \to Y \approx Z \end{aligned} \tag{22}$$

$$\begin{aligned} \mathsf{concept}(C, X) \wedge r \in R \wedge \mathsf{roles}(R, X, Y) \wedge \mathsf{concept}(D, Y) \wedge \\ \mathsf{anon}(Y) \wedge r \in S \wedge \mathsf{roles}(S, X, Z) \wedge \mathsf{concept}(D, Z) \wedge \mathsf{anon}(Z) \to \mathsf{roles}(R \cup S, X, Y \cup Z) \end{aligned} \tag{23}$$

$$\begin{aligned} \mathsf{concept}(C, X) \wedge r \in R \wedge \mathsf{inv}(R, R^-) \wedge \mathsf{concept}(D, Y) \wedge \mathsf{roles}(R^-, Y, X) \wedge \\ r \in S \wedge \mathsf{inv}(S, S^-) \wedge \mathsf{anon}(Z) \wedge E \in Z \wedge \mathsf{roles}(S, X, Z) \wedge \mathsf{concept}(D, Z) \to \mathsf{concept}(E, Y) \wedge \\ \mathsf{roles}(R^- \cup S^-, Y, X) \end{aligned} \tag{24}$$

$$\mathsf{concept}(D, Y) \wedge \mathsf{role}(r^-, Y, X) \wedge \mathsf{concept}(C, X) \wedge \mathsf{n}(X) \to \mathsf{n}(Y) \tag{25}$$

In addition, we need the following rule that completes the translation of axioms of types (ii)–(iv) by adding the newly created anonymous individuals to the required concepts:

$$\mathsf{role}(r, X, Y) \wedge C \in Y \to \mathsf{concept}(C, Y) \tag{26}$$

For axiom type (v):

$$\mathsf{concept}(C, X) \to \{a\} \approx X \wedge \mathsf{n}(\{a\}) \tag{27}$$

For axiom type (vi), we use the predicate sup that connects each role name to the set of all its super-roles (Rule 29 is instantiated for every $s \sqsubseteq t \in \mathcal{T}$ or $s^- \sqsubseteq t^- \in \mathcal{T}$):

$$\to \mathsf{sup}(r, \{r\}) \tag{28}$$

$$\mathsf{sup}(r, S) \wedge s \in S \to \mathsf{sup}(r, S \cup \{t\}) \tag{29}$$

$$\mathsf{role}(r, X, Y) \wedge \mathsf{sup}(r, S) \to \mathsf{roles}(S, X, Y) \tag{30}$$

$$\mathsf{role}(r^-, X, Y) \wedge \mathsf{sup}(r, S) \wedge \mathsf{inv}(S, S^-) \to \mathsf{roles}(S^-, X, Y) \tag{31}$$

Finally, axiom type (vii) can be handled like a concept assertion:

$$\to \mathsf{concept}(C, \{a\}) \wedge \mathsf{n}(\{a\}) \tag{32}$$

So far, the rules did not use negation and can be seen as the first stratum of the final rule set. This part is already a rewriting of any instance query over the TBox signature, i.e. $\mathsf{concept}(C, \{a\})$ is contained in the least Herbrand model of these rules and an ABox iff $C(a)$ is entailed by the original TBox over the ABox (Carral, Dragoste, and Krötzsch 2018, Lemma 4). To be able to answer arbitrary UCQs, we also need to encode the so-called *filtration phase* (Carral, Dragoste, and Krötzsch 2018).

Building a Canonical Model

The next stratum encodes Definition 7 from Carral, Dragoste, and Krötzsch (2018) by using negated atoms over the predicate n. The goal of this part is to derive a dependency relation $\mathsf{role}_\triangleright(r, X, Y)$ that encodes the order in which individuals are created during the construction of a model of the ontology. Extended individuals of the form $t^i_{R,X}$ are used in this relation, where $R \subseteq \mathbf{R}$, $X \subseteq \mathbf{C}$ and $i \in \{0, 1, 2\}$ (Carral, Dragoste, and Krötzsch 2018). Therefore, the sets X, Y in $\mathsf{role}_\triangleright(r, X, Y)$ are now considered to be subsets of $\mathbf{R} \cup \mathbf{C} \cup \mathbf{I} \cup \{0, 1, 2\}$, where again individuals can only occur in singleton sets $\{a\}$, and at most one index $0, 1, 2$ can be present in any given set. In addition to $\mathsf{role}_\triangleright$, the following rules also compute extensions role' and $\mathsf{concept}'$ of role and concept, respectively, to the new individuals. Together, these three predicates describe a kind of *canonical model* (called $\mathcal{C}_\mathcal{O}$) over which UCQs will be answered.

As a prerequisite, we need to introduce an intermediate stratum to define a total order $\preceq$ on sets of the form $R \cup X$. For this purpose, we consider an enumeration $r_1, \ldots, r_n, C_{n+1}, \ldots, C_{n+m}$ of all role and concept names and use the following rules to define the lexicographic order $\preceq$ based on the auxiliary relations $\prec_i$ and $\approx_i$, $i \in \{1, \ldots, n+m\}$ (all using infix notation):

$$\emptyset \approx_1 \emptyset \tag{33}$$

$$\emptyset \prec_1 \{r_1\} \tag{34}$$

$$\{r_1\} \approx_1 \{r_1\} \tag{35}$$

$$X \prec_1 Y \to X \prec_2 Y \tag{36}$$

$$X \approx_1 Y \to X \approx_2 Y \tag{37}$$

$$X \approx_1 Y \to X \prec_2 Y \cup \{r_2\} \tag{38}$$

$$X \approx_1 Y \to X \cup \{r_2\} \approx_2 Y \cup \{r_2\} \tag{39}$$

$$\vdots$$

$$X \approx_{n+m} Y \to X \preceq Y \tag{40}$$

$$X \prec_{n+m} Y \to X \preceq Y \tag{41}$$

The following rules, which use the negations of $\preceq$ and n from the previous strata, correspond to Definition 7 from Carral, Dragoste, and Krötzsch (2018), where $j = (i+1) \mod 3$:

$$\mathsf{n}(X) \wedge \mathsf{role}(r, X, Y) \wedge \mathsf{n}(Y) \to \mathsf{role}_\triangleright(r, X, Y) \wedge \mathsf{role}_\triangleright(r^-, Y, X) \tag{42}$$

$$\mathsf{n}(X) \wedge \mathsf{roles}(R, X, Y) \wedge \mathsf{anon}(Y) \wedge \neg\mathsf{n}(Y) \wedge r \in R \to \mathsf{role}_\triangleright(r, X, R \cup Y \cup \{0\}) \tag{43}$$

$$\mathsf{role}_\triangleright(r, X, R \cup Y \cup \{i\}) \wedge \mathsf{roles}(S, Y, Z) \wedge \mathsf{anon}(Z) \wedge \neg\mathsf{n}(Z) \wedge s \in S \wedge R \cup Y \preceq S \cup Z \to \mathsf{role}_\triangleright(s, R \cup Y \cup \{i\}, S \cup Z \cup \{j\}) \tag{44}$$

$$\mathsf{role}_\triangleright(r, X, R \cup Y \cup \{i\}) \wedge \mathsf{roles}(S, Y, Z) \wedge \mathsf{anon}(Z) \wedge \neg\mathsf{n}(Z) \wedge s \in S \wedge R \cup Y \not\preceq S \cup Z \to \mathsf{role}_\triangleright(s, R \cup Y \cup \{i\}, S \cup Z \cup \{i\}) \tag{45}$$

$$\mathsf{role}_\triangleright(s, X, R \cup Y \cup \{i\}) \wedge \mathsf{role}(r, Y, Z) \wedge \mathsf{n}(Z) \to \mathsf{role}_\triangleright(r, R \cup Y \cup \{i\}, Z) \wedge \mathsf{role}_\triangleright(r^-, Z, R \cup Y \cup \{i\}) \tag{46, 47}$$

$$\mathsf{role}_{\triangleright}(s, X, Y) \wedge \mathsf{concept}(C, Y) \rightarrow \mathsf{concept}'(C, Y) \tag{48}$$

$$\mathsf{role}_{\triangleright}(s, X, R \cup Y \cup \{i\}) \wedge \mathsf{concept}(C, Y) \rightarrow \mathsf{concept}'(C, R \cup Y \cup \{i\}) \tag{49}$$

$$\mathsf{role}_{\triangleright}(r, X, Y) \rightarrow \mathsf{role}'(r, X, Y) \wedge \mathsf{role}'(r^-, Y, X) \tag{50}$$

The least Herbrand model of these rules (restricted to $\mathsf{role}_{\triangleright}$, $\mathsf{concept}'$, and role') corresponds to the set $\mathcal{C}_{\mathcal{O}}$ (Carral, Dragoste, and Krötzsch 2018). Conditions of the type "$t^i_{R,Y}$ is in $\mathcal{C}_{\mathcal{O}}$" are translated into body atoms like $\mathsf{role}_{\triangleright}(r, X, R \cup Y \cup \{i\})$ since these new individuals are only introduced into $\mathcal{C}_{\mathcal{O}}$ inside of $\mathsf{role}_{\triangleright}$-facts.

Encoding the Filtration

Finally, we encode Definition 8 from Carral, Dragoste, and Krötzsch (2018), which constructs a family of graphs that use the variables of a given CQ as vertices, and their edges encode possible matches of the CQ into $\mathcal{C}_{\mathcal{O}}$. Some of these matches have to be filtered out since they lead to spurious answers. We assume that the input CQ q is of the form $\exists v_1, \ldots, v_\ell . \phi(v_1, \ldots, v_k)$, i.e. $v_{\ell+1}, \ldots, v_k$ are the free variables (UCQs can be treated by encoding each component CQ individually and then merging the results in a single predicate). By $\phi(V_1, \ldots, V_k)$ we denote the result of replacing each concept atom $C(v_i)$ in ϕ by $\mathsf{concept}'(C, V_i)$ and similarly $r(v_i, v_j)$ by $\mathsf{role}'(r, V_i, V_j)$, where each V_i is viewed as a set variable over $\mathbf{R} \cup \mathbf{C} \cup \{i\}$ and denotes a possible mapping of v_i into $\mathcal{C}_{\mathcal{O}}$. We use the indices $1, \ldots, k$ to refer to the vertices in the graphs that are constructed, and atoms of the form $\mathsf{edge}(i, j, V_1, \ldots, V_k)$ to denote an edge from i to j (which depends on a specific instantiation of all variables by individuals in $\mathcal{C}_{\mathcal{O}}$).

The following are the rules corresponding to the first part of Definition 8 from Carral, Dragoste, and Krötzsch (2018), for all role atoms $r(v_i, v_j)$ in ϕ:

$$\phi(V_1, \ldots, V_k) \wedge \mathsf{role}_{\triangleright}(r, V_i, V_j) \wedge \neg \mathsf{role}_{\triangleright}(r^-, V_j, V_i) \rightarrow \mathsf{edge}(i, j, V_1, \ldots, V_k) \tag{51}$$

$$\phi(V_1, \ldots, V_k) \wedge \mathsf{role}_{\triangleright}(r^-, V_j, V_i) \wedge \neg \mathsf{role}_{\triangleright}(r, V_i, V_j) \rightarrow \mathsf{edge}(j, i, V_1, \ldots, V_k) \tag{52}$$

Now, for all $i, j \in \{1, \ldots, k\}$, we apply the following rules to collapse this graph according to Definition 8 from Carral, Dragoste, and Krötzsch (2018):

$$\mathsf{edge}(i, j, V_1, \ldots, V_k) \wedge \mathsf{edge}(m, j, V_1, \ldots, V_k) \rightarrow \mathsf{equal}(i, m, V_1, \ldots, V_k) \tag{53}$$

$$\mathsf{edge}(i, j, V_1, \ldots, V_k) \wedge \mathsf{edge}(m, n, V_1, \ldots, V_k) \wedge \mathsf{equal}(j, n, V_1, \ldots, V_k) \rightarrow \mathsf{equal}(i, m, V_1, \ldots, V_k) \tag{54}$$

We now need to check whether the resulting graph is a rooted directed forest, i.e. contains no cycles and no "diamonds" that reconnect different branches:

$$\mathsf{edge}(i, j, V_1, \ldots, V_k) \rightarrow \mathsf{reach}(i, j, V_1, \ldots, V_k) \tag{55}$$

$$\mathsf{reach}(i, j, V_1, \ldots, V_k) \wedge \mathsf{equal}(j, m, V_1, \ldots, V_k) \rightarrow \mathsf{reach}(i, m, V_1, \ldots, V_k) \tag{56}$$

$$\mathsf{reach}(i, j, V_1, \ldots, V_k) \wedge \mathsf{equal}(i, m, V_1, \ldots, V_k) \rightarrow \mathsf{reach}(m, j, V_1, \ldots, V_k) \tag{57}$$

$$\mathsf{reach}(i, j, V_1, \ldots, V_k) \wedge \mathsf{edge}(j, m, V_1, \ldots, V_k) \rightarrow \mathsf{reach}(i, m, V_1, \ldots, V_k) \tag{58}$$

$$\mathsf{reach}(i, i, V_1, \ldots, V_k) \rightarrow \mathsf{bad}(V_1, \ldots, V_k) \tag{59}$$

$$\mathsf{edge}(i, j, V_1, \ldots, V_k) \wedge \mathsf{edge}(i, m, V_1, \ldots, V_k) \wedge \neg \mathsf{equal}(j, m, V_1, \ldots, V_k) \wedge \mathsf{reach}(j, n, V_1, \ldots, V_k) \wedge \mathsf{reach}(m, n, V_1, \ldots, V_k) \rightarrow \mathsf{bad}(V_1, \ldots, V_k) \tag{60}$$

Finally, we can use the predicate bad to filter out spurious matches and return the actual answers to q in the predicate P_q:

$$\phi(V_1, \ldots, V_k) \wedge \neg \mathsf{bad}(V_1, \ldots, V_k) \rightarrow P'_q(V_{\ell+1}, \ldots, V_k) \tag{61}$$

$$P'_q(V_{\ell+1}, \ldots, V_k) \wedge \mathsf{ind}(V_{\ell+1}) \wedge \cdots \wedge \mathsf{ind}(V_k) \wedge a_{\ell+1} \in V_{\ell+1} \wedge \cdots \wedge a_k \in V_k \rightarrow P_q(a_{\ell+1}, \ldots, a_k) \tag{62}$$

From Datalog$^{S,\neg}$ to Datalog$^{\neg}$

To simulate set terms in plain Datalog$^{\neg}$, we adapt an existing construction, which did not deal with negation or ABox-independent rule sets (Ortiz, Rudolph, and Šimkus 2010). First, each set union $t_1 \cup t_2$ over the domain S is replaced with a fresh variable X and the ternary atom $\mathsf{U}_S(t_1, t_2, X)$ is added to the body of the rule in which this term occurs. New rules are added to simulate the set union with this predicate (see below). Then, sets $X \subseteq S$ are represented as bit vectors of length $|S|$ and set variables as vectors of variables, and all atoms are replaced accordingly. This gets rid of all set expressions while increasing the arity of the predicates polynomially.

The main problem we face here is that we used singleton sets $\{a\}$ to refer to individual names a from the ABox, which means that the set sort $2^{\mathbf{C} \cup \mathbf{I}}$ was treated as if it contained all these individual names, although our translation cannot depend on the ABox (see Definition 1). To avoid this issue, we modify the bit vector encoding above to directly represent constants by themselves. For this, recall that individual names a can only occur in singleton sets $\{a\}$, because sets X are only extended if they belong to anon (see, e.g. Rules (21), (23), or (43)). Thus, we can represent each instantiated set X, which is either of

the form $\{a\}$ or a subset of $\mathbf{C}$, as a vector $(a, 0, \ldots, 0)$ or $(0, b_1, \ldots, b_m)$, respectively, where $m = |\mathbf{C}|$ and $b_i = 1$ iff the i-th concept name of $\mathbf{C}$ is in X (according to some fixed enumeration of $\mathbf{C}$). The new constants 0 and 1 are used to represent bit values. Correspondingly, set variables X are split into vectors $(x_0, x_1, \ldots, x_m)$, where x_0 holds the individual name (if any) and $x_1, \ldots, x_m$ represent a subset of $\mathbf{C}$. The encoding works similarly for sets of the sorts $2^{\mathbf{R}}$, $2^{\mathbf{R} \cup \mathbf{C}}$, and $2^{\mathbf{R} \cup \mathbf{C} \cup \mathbf{I} \cup \{0,1,2\}}$ that are employed in the rewriting above. For example, the Datalog$^\neg$ versions of Rules (9) and (19) are

$$C(x) \rightarrow \mathsf{concept}(C, x, 0, \ldots, 0) \wedge \mathsf{ind}(x, 0, \ldots, 0), \text{ and} \tag{63}$$

$$\mathsf{concept}(C, x_0, \ldots, x_m) \rightarrow \mathsf{role}(r, x_0, \ldots, x_m, 0, b_1, \ldots, b_m), \tag{64}$$

respectively, where $b_i = 1$ iff D is the i-th concept name in $\mathbf{C}$.

Apart from the new predicates like $\mathsf{U}_{\mathbf{C} \cup \mathbf{I}}$, this encoding clearly preserves the stratification of the original rule set. The following additional rules are used to define $\mathsf{U}_{\mathbf{C} \cup \mathbf{I}}$, and similarly for the other set sorts:

$$\rightarrow \mathsf{max}(0, 0, 0) \tag{65}$$

$$\rightarrow \mathsf{max}(1, 0, 1) \tag{66}$$

$$\rightarrow \mathsf{max}(0, 1, 1) \tag{67}$$

$$\rightarrow \mathsf{max}(1, 1, 1) \tag{68}$$

$$\mathsf{max}(x_1, y_1, z_1) \wedge \cdots \wedge \mathsf{max}(x_m, y_m, z_m) \rightarrow \mathsf{U}_{\mathbf{C} \cup \mathbf{I}}(0, x_1, \ldots, x_m, 0, y_1, \ldots, y_m, 0, z_1, \ldots, z_m), \tag{69}$$

This suffices to define the set union since we never need to compute unions involving singleton sets $\{a\}$ with $a \in \mathbf{I}$. These additional rules can be included in the first stratum since max and U_S do not occur in any other rule heads.

This finishes the presentation of the Datalog$^\neg$ rewriting for any UCQ over a Horn-$\mathcal{ALCHOIQ}$ TBox. Its correctness follows mainly from an existing result (Carral, Dragoste, and Krötzsch 2018, Theorem 3) since we only translated the relevant definitions into Datalog$^{S,\neg}$ rules. It can also be verified that the resulting set of Datalog$^\neg$ rules is of polynomial size.

B Proof of Theorem 5

We show how to construct the eKAB task $(\Delta_n, \mathcal{O}, I_w, g)$ such that M accepts a word w of length n iff there is a plan of length 1. The construction is based on the 2ExpTime-hardness proof for Horn-$\mathcal{SROIQ}$ (Ortiz, Rudolph, and Šimkus 2010) and uses only a single action with precondition $[\exists x.B(x)]$ and unconditional effect g. We do not repeat all details of the original construction here, but only adapt the relevant parts. The proof encodes the Turing machine M and an input word w into a TBox $\mathcal{T}$ using the two objects o and e such that M accepts w iff at least one unary predicate H_{q_f} (representing a final state of the TM) is empty in every model of $\mathcal{T}$. The original proof then goes on to add axioms $H_{q_f} \sqsubseteq \bot$ to force $\mathcal{T}$ to become unsatisfiable in such a case. Since we require the initial state to be consistent with the TBox, we instead use the axioms $H_{q_f} \sqsubseteq B$, which allows us to query for $\exists x.B(x)$ instead of checking unsatisfiability.

We further adapt the original reduction by extracting from $\mathcal{T}$ the description of the input word w into a state I_w. For $w = w_0 \ldots w_{n-1}$, $\mathcal{T}$ contains the following axioms to encode the input tape (notation is slightly adjusted to avoid clashes):

$$\{o\} \sqsubseteq I_1 \sqcap H_{q_0} \tag{70}$$

$$I_j \sqsubseteq A_{w_j} \sqcap \forall h.I_{j+1} \qquad (0 \leq j < n) \tag{71}$$

$$I_j \sqsubseteq \overline{H_r} \qquad (1 \leq j < n) \tag{72}$$

$$I_n \sqsubseteq A_\Box \tag{73}$$

$$I_n \sqsubseteq \forall h.I_n \tag{74}$$

The object o indicates the starting point of the tape, and the unary predicates I_j identify the first n cells, which are connected via the binary predicate h. The predicates A_{w_j} indicate the presence of the input symbols in these cells. The predicates H_{q_0} and $\overline{H_r}$ indicate the presence and absence of the head, respectively. Finally, all cells to the right of the input word are labeled with a blank symbol $\Box$ by using the auxiliary predicate I_n. We leave the axioms (72)–(74) in the TBox, but replace (70)–(71) by the following axioms (75)–(76) and assertions for I_w (77)–(78):

$$S_{j,a} \sqsubseteq \forall h^j.I_j \sqcap A_a \qquad (0 \leq j < n,\ a \in \Sigma) \tag{75}$$

$$I_{n-1} \sqsubseteq \forall h_n.I_n \tag{76}$$

$$H_{q_0}(o) \tag{77}$$

$$S_{j,w_j}(o) \qquad (0 \leq j < n) \tag{78}$$

Here, $\forall h^j$ stands for j nested restrictions of the form $\forall h$, and $S_{j,a}$ indicates the presence of the symbol a of the TM alphabet Σ at cell j. In this way, the final TBox $\mathcal{T}_n$ only depends on the length n of the input word w, but not on w itself. The final domain description is $\Delta_n = (\mathcal{P}_n, \mathcal{A}_n, \mathcal{T}_n)$, where $\mathcal{P}_n$ contains all symbols from $\mathcal{T}_n$ as well as g, and $\mathcal{A}_n$ consists of the single action described above.

C Proof of Theorem 6

The proof follows the same arguments as for Theorem 5, the only difference being how the domain description Δ_n and state I_w are obtained. For CQ entailment in $\mathcal{ALCI}$, we adapt a reduction from a (universal) AEXPSPACE Turing machine (Lutz 2007). The single action we use in the reduction will have a precondition $[q_w]$, where q_w is the CQ from that reduction, which, despite its name, does not depend on the input word $w = w_0 \dots w_{n-1}$, but only on the length n. Again, the only adaption we have to do is to extract the part of the TBox encoding the input word into a state I_w. Consider the relevant axioms (again with slightly adapted notation), where $0 \leq j < n$ (Lutz 2007):

$$\big(R \sqcap I\big)(o) \tag{79}$$

$$I \sqsubseteq \forall s^{n+2}.\big((G_h \sqcap (\mathsf{pos} = j)) \to w_j\big) \tag{80}$$

$$I \sqsubseteq \forall s^{n+2}.\big((G_h \sqcap (\mathsf{pos} = 0)) \to q_0\big) \tag{81}$$

$$I \sqsubseteq \forall s^{n+2}.\big((G_h \sqcap (\mathsf{pos} \geq n)) \to b\big) \tag{82}$$

Here, $R \sqcap I$ describes the starting point of the initial configuration and its s^{n+2}-successors marked with G_h identify the exponentially many tape cells. The counter pos identifies particular cells, w_j describes the tape content, q_0 the initial state, and b the blank symbol. We use a similar trick as before to split (80) into ABox facts and TBox axioms that do not depend on the input word w, but only on its length (for all $0 \leq j < n$, $a \in \Sigma$):

$$S_{j,a} \sqsubseteq \forall s^{n+2}.\big((G_h \sqcap (\mathsf{pos} = j)) \to a\big) \tag{83}$$

$$S_{j,w_j}(o) \tag{84}$$

The state I_w now consists of all facts (84) as well as (79), and all other axioms are part of $\mathcal{T}_n$. The remaining arguments are the same as in the proof of Theorem 5.

The reduction for CQ entailment over $\mathcal{SH}$ TBoxes (Eiter et al. 2009) is very similar to the previous case, except that the predicates R and G_h are not used, s^{n+2} is replaced by r^{n+1}, w_j is replaced by $\forall r.(E_h \to w_j)$, and similarly for q_0 and b (many other details not relevant here are different as well). Hence, we can use very similar adaptations.

D 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, $n \in \{5, \dots, 10\}$, and queens, $m \in \{n-4, \dots, n\}$. 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.

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

- 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

$$\begin{array}{r}\text{(when (and (AboveOf1 ?x) (BelowOf2 ?x))}\\ \text{(Row0 ?x)),}\end{array} \tag{85}$$

  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

$$\begin{array}{r}\text{(when (Row1 ?x))}\\ \text{(Row0 ?x)),}\end{array} \tag{86}$$

  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 $\mathsf{AboveOf1} \sqcap \mathsf{BelowOf2} \sqsubseteq \mathsf{Row1}$, which allows us to get rid of (85).

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. https://lat.inf.tu-dresden.de/research/papers/2021/BHKS-KR21.pdf.

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

Carral, D.; Dragoste, I.; and Krötzsch, M. 2018. The Combined Approach to Query Answering in Horn-$\mathcal{ALCHOIQ}$. In Thielscher, M.; Toni, F.; and Wolter, F., eds., *Proc. of the 16th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'18)*, 339–348. AAAI Press.

Eiter, T.; Lutz, C.; Ortiz, M.; and Šimkus, M. 2009. Query Answering in Description Logics with Transitive Roles. INFSYS Research Report 1843-09-02, Institut für Informationssysteme, TU Wien.

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)*.

Lutz, C. 2007. Inverse Roles Make Conjunctive Queries Hard. In Calvanese, D.; Franconi, E.; Haarslev, V.; Lembo, D.; Motik, B.; Turhan, A.-Y.; and Tessaris, S., eds., *Proc. of the 20th Int. Workshop on Description Logics (DL'07)*, volume 250 of *CEUR Workshop Proceedings*, 100–111.

Ortiz, M.; Rudolph, S.; and Šimkus, M. 2010. Worst-case Optimal Reasoning for the Horn-DL Fragments of OWL 1 and 2. In Lin, F.; Sattler, U.; and Truszczynski, M., eds., *Proc. of the 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'10)*, 269–279. AAAI Press.