Chasing Sets: How to Use Existential Rules for Expressive Reasoning (Extended Abstract) *

David Carral, Irina Dragoste, Markus Krötzsch, and Christian Lewe

Knowledge-Based Systems Group, TU Dresden, Dresden, Germany

Rules of inference play a major role in knowledge representation and reasoning, e.g., for consequence-based deduction calculi in DLs. Highly scalable rule engines are available for the popular rule language Datalog [2,3,4,8,12,15], and the ExpTime (combined) reasoning complexity of this logic suggests that these engines could be exploited for reasoning in DLs. However, solving ExpTime-complete problems by translation to Datalog requires polynomially large predicate arities [13] or exponentially many rules [6], neither of which is feasible.

To overcome this issue, we propose Datalog(S) – an extension of Datalog with terms that represent sets – and we show that it can be used to express common DL reasoning calculi by a simple translation of inference rules. Inspired by the approach in [9], we encode ontologies as facts, while the deductive calculus is a fixed Datalog(S) rule set. This extends and generalises an approach of Ortiz et al. [13] by making sets data-dependent, leading to ExpTime data complexity.

The main novelty of our work is the insight that Datalog(S)-reasoning can be encoded in theories of *existential rules* outside any known class, but for which the *standard chase* procedure is guaranteed to terminate, producing a finite model in exponential (i.e., worst-case optimal) time. This is surprising, as all previously known concrete rule languages for which the chase terminates feature PTime data complexity [5,11], which is strictly too weak for implementing DL reasoning in this way. Our insight enables us to use existing rule reasoners for Datalog(S), and therefore for DLs, and we show by empirical evaluation that this can lead to feasible practical implementations even without extensive optimisation.

Datalog(S) is defined as a logic with two sorts: an object sort for regular domain elements, and a set sort for sets over the elements of this domain. We impose certain syntactic restrictions that ensure finite object domains, and therefore finite (though exponentially larger) set domains. We provide built-in functions $\{\}$ and $\cup$ for constructing set terms, and built-in predicates $\in$ and $\subseteq$ that can be used in rule premises for accessing the contents of sets. For example, the Datalog(S) rules below define a unary predicate for all non-empty sets of books:

$$book(x) \rightarrow bookSet(\{x\}) \qquad bookSet(X) \wedge bookSet(Y) \rightarrow bookSet(X \cup Y)$$

The number of sets in a canonical model of these rules is exponential in the number of elements classified as books in the input data, and this is the source of the additional expressive power of Datalog(S).

We illustrate our approach by encoding the ExpTime-complete reasoning task of *classification* for a Horn-$\mathcal{ALC}$ TBox into a Datalog(S) program. We

* The full version of this paper has been accepted for publication at IJCAI'19 [7].

$(\mathbf{R_A})$ $\frac{}{H \sqsubseteq A} : H$ is active and $A \in H$		$Act(H) \wedge a \in H \rightarrow Sc(H, a)$
$(\mathbf{R_\sqcap})$ $\frac{\{H \sqsubseteq A_i\}_{i=1}^{n}}{H \sqsubseteq C}$:	$n = 0$, H active, $\top \sqsubseteq C \in \mathcal{T}$, or $n = 1$, $A_1 \sqsubseteq C \in \mathcal{T}$, or $n = 2$, $A_1 \sqcap A_2 \sqsubseteq C \in \mathcal{T}$	$Act(H) \wedge ax_{\sqsubseteq}(c_\top, c) \rightarrow Sc(H, c)$ $Sc(H, a_1) \wedge ax_{\sqsubseteq}(a_1, c) \rightarrow Sc(H, c)$ $Sc(H, a_1) \wedge Sc(H, a_2) \wedge ax_{\sqcap\sqsubseteq}(a_1, a_2, c) \rightarrow Sc(H, c)$
$(\mathbf{R}^{+}_{\exists})$ $\frac{H \sqsubseteq A}{H \sqsubseteq \exists R.B} : A \sqsubseteq \exists R.B \in \mathcal{T}$		$Sc(H, a) \wedge ax_{\sqsubseteq\exists}(a, r, b) \rightarrow Ex(H, r, \{b\})$
$(\mathbf{R}^{-}_{\exists})$ $\frac{H \sqsubseteq \exists R.K \quad K \sqsubseteq A}{H \sqsubseteq B} : \exists R.A \sqsubseteq B \in \mathcal{T}$		$Ex(H, r, K) \wedge Sc(K, a) \wedge ax_{\exists\sqsubseteq}(a, r, b) \rightarrow Sc(H, b)$
$(\mathbf{R}^{\perp}_{\exists})$ $\frac{H \sqsubseteq \exists R.K \quad K \sqsubseteq \perp}{H \sqsubseteq \perp}$		$Ex(H, r, K) \wedge Sc(K, c_\perp) \rightarrow Sc(H, c_\perp)$
$(\mathbf{R}_{\forall})$ $\frac{H \sqsubseteq \exists R.K \quad H \sqsubseteq A}{H \sqsubseteq \exists R.(K \sqcap B)} : A \sqsubseteq \forall R.B \in \mathcal{T}$		$Ex(H, r, K) \wedge Sc(H, a) \wedge ax_{\sqsubseteq\forall}(a, r, b) \rightarrow Ex(H, r, \{b\} \cup K) \wedge Act(\{b\} \cup K)$

Fig. 1. Classification Horn-$\mathcal{ALC}$ inference rules from [14] (left) and corresponding Datalog(S) rule set $\mathcal{R}$ (right), where $c_\top$ and $c_\perp$ are constants, lower-case letters are object variables, and upper-case letters are set variables

define a Horn-$\mathcal{ALC}$ TBox $\mathcal{T}$ as a set of axioms in the following normal form: $\top \sqsubseteq C$, $A \sqsubseteq C$, $A \sqcap B \sqsubseteq C$, $A \sqsubseteq \exists R.C$, $\exists R.A \sqsubseteq C$, $A \sqsubseteq \perp$, $A \sqsubseteq \forall R.C$, where A, B, C are concept names, R is a role name, and $\top, \perp$ are the top and bottom concepts. *Classification* is the task of computing all axioms of the form $A \sqsubseteq C$ that are entailed by $\mathcal{T}$. We consider the *consequence-driven* classification method for Horn-$\mathcal{ALC}$ proposed in [14], which is shown in Figure 1 (left) as a set of inference rules. For an input TBox $\mathcal{T}$, the rules produce inferences of the form $H \sqsubseteq B$ and $H \sqsubseteq \exists R.K$, with H, K representing conjunctions of concepts. Rules $(\mathbf{R_A})$ and $(\mathbf{R_\sqcap})$ are restricted to the set of *active* conjunctions, which is initialised with all singleton conjunctions (i.e. concept names), and is further extended with the new conjunctions derived by rule $(\mathbf{R_\forall})$.

This classification calculus can be encoded as a Datalog(S) program consisting of the rule set $\mathcal{R}$ shown Figure 1 (right), and a set of facts $\mathcal{F}(\mathcal{T})$ encoding the axioms in the input TBox: for example, an axiom $A_1 \sqcap A_2 \sqsubseteq C \in \mathcal{T}$ will be encoded as a fact $ax_{\sqcap\sqsubseteq}(A_1, A_2, C)$. Each of the Datalog(S) rules in the right of Figure 1 corresponds to the inference rule on its left, where conjunctions are represented as sets, predicates Sc and Ex encode inferences, and predicate Act signals active conjunctions. These can be initialised by adding $cn(a)$ to $\mathcal{F}(\mathcal{T})$ for each concept name a in $\mathcal{T}$, and adding the rule $cn(a) \rightarrow Act(\{a\})$ to $\mathcal{R}$. Rule $(\mathbf{R_\forall})$ shows how adding a concept name B to a conjunction of concept names K can be encoded as the set union $\{B\} \cup K$, which results in a new active set. The semantics of Datalog(S) rule set $\mathcal{R}$ can be captured by a set of existential rules for which a standard chase variant that prioritises non-generating rules [10] is guaranteed to terminate for all inputs. This approach leads to an ExpTime-complete (hence worst-case optimal) chase-based classification algorithm for Horn-$\mathcal{ALC}$.

In summary, our proposed language Datalog(S) can be used to translate algorithms of ExpTime-complete data complexity into programs with a fixed set of existential rules. Besides providing an elegant, fully-declarative implementation for DL reasoning, our approach can take advantage of rule engines capabilities.

Acknolwedgements. This work is supported by Deutsche Forschungsgemeinschaft in project number 389792660 (TRR 248, Center for Perspicuous Systems) and Emmy Noether grant KR 4381/1-1 (DIAMOND).

References

1. Proc. 22nd Int. Joint Conf. on Artif. Intell. (IJCAI'11). AAAI Press/IJCAI (2011)
2. Aref, M., ten Cate, B., Green, T.J., Kimelfeld, B., Olteanu, D., Pasalic, E., Veldhuizen, T.L., Washburn, G.: Design and implementation of the LogicBlox system. In: Proc. 2015 ACM SIGMOD Int. Conf. on Management of Data. pp. 1371–1382. ACM (2015)
3. Baget, J., Leclère, M., Mugnier, M., Rocher, S., Sipieter, C.: Graal: A toolkit for query answering with existential rules. In: Bassiliades, N., Gottlob, G., Sadri, F., Paschke, A., Roman, D. (eds.) Proc. 9th Int. Web Rule Symposium (RuleML'15). LNCS, vol. 9202, pp. 328–344. Springer (2015)
4. Benedikt, M., Leblay, J., Tsamoura, E.: PDQ: proof-driven query answering over web-based data. PVLDB **7**(13), 1553–1556 (2014)
5. Carral, D., Dragoste, I., Krötzsch, M.: Restricted chase (non)termination for existential rules with disjunctions. In: Sierra, C. (ed.) Proc. 26th Int. Joint Conf. on Artificial Intelligence (IJCAI'17). pp. 922–928. ijcai.org (2017)
6. Carral, D., Dragoste, I., Krötzsch, M.: The combined approach to query answering in Horn-$\mathcal{ALCHOIQ}$. In: Proc. 16th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'16). pp. 339–348. AAAI Press (2018)
7. Carral, D., Dragoste, I., Krötzsch, M., Lewe, C.: Chasing sets: How to use existential rules for expressive reasoning. In: Sierra, C. (ed.) Proc. 28th Int. Joint Conf. on Artificial Intelligence (IJCAI'19). ijcai.org (2019)
8. Geerts, F., Mecca, G., Papotti, P., Santoro, D.: That's all folks! LLUNATIC goes open source. PVLDB **7**(13), 1565–1568 (2014)
9. Krötzsch, M.: Efficient rule-based inferencing for OWL EL. In: Proc. 22nd Int. Joint Conf. on Artif. Intell. (IJCAI'11) [1], pp. 2668–2673
10. Krötzsch, M., Marx, M., Rudolph, S.: The power of the terminating chase. In: Proc. 22st Int. Conf. on Database Theory (ICDT'19). LIPIcs, vol. 127, pp. 3:1–3:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2019)
11. Marnette, B.: Generalized schema-mappings: from termination to tractability. In: Proc. 28th Symposium on Principles of Database Systems (PODS'09). pp. 13–22. ACM (2009)
12. Nenov, Y., Piro, R., Motik, B., Horrocks, I., Wu, Z., Banerjee, J.: RDFox: A highly-scalable RDF store. In: et al., M.A. (ed.) Proc. 14th Int. Semantic Web Conf. (ISWC'15), Part II. LNCS, vol. 9367, pp. 3–20. Springer (2015)
13. Ortiz, M., Rudolph, S., Simkus, M.: Worst-case optimal reasoning for the Horn-DL fragments of OWL 1 and 2. In: Proc. 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'10). pp. 269–279. AAAI Press (2010)
14. Simančík, F., Kazakov, Y., Horrocks, I.: Consequence-based reasoning beyond Horn ontologies. In: Proc. 22nd Int. Joint Conf. on Artif. Intell. (IJCAI'11) [1], pp. 1093–1098
15. Urbani, J., Jacobs, C., Krötzsch, M.: Column-oriented Datalog materialization for large knowledge graphs. In: Proc. 30th AAAI Conf. on Artificial Intelligence (AAAI'16). pp. 258–264. AAAI Press (2016)