PhD Project Proposal: An Optimisation Procedure for Disjunctive Datalog Reasoning

David Carral
LIRMM, Inria, University of Montpellier
david.carral@inria.fr

May 17, 2022

1 Introduction

Hi! This is my best attempt to recruit some student that is motivated to work with me on an interesting research project! Before going ahead with the technical content (which I present in the following sections), I am going to present myself and discuss a bit the organisational aspects of this proposal:

Brief academic history. I did my PhD at Wright State University under the supervision of Prof. Pascal Hitzler. After my PhD, I worked as a postdoc at TU Dresden in the research group led by Prof. Markus Krötzsch. Since the beginning of 2021, I am a member of the GraphIK Inria team (soon to be called Boreal) at the University of Montpellier.

My research topics. Broadly speaking, I am interested in the study of logical languages (e.g., first-order logic, Description Logics [3], existential rules[1], etc), their theoretical properties, and the implementation of reasoning algorithms for such languages. To know a bit more about my work, you can check out my personal webpage or my Google Scholar page:

- Personal page: `https://www-sop.inria.fr/members/David.Carral/`

[1] To know more about existential rules read the "Dependencies" lecture of the course `https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2020)/en`

- Google: https://scholar.google.com/citations?user=5bEOMysAAAAJ

What I am looking for. I am looking for a student who is intrinsically motivated to work on some research topic that I also find interesting. I propose one such topic in this document but if you are interested in working on something else (and you think that I may also be interested in it), feel free to contact me about it.

What I offer. Long story short, I offer close and active supervision to help you solve some interesting research question. On top of that, I can also provide funding for 3 years!

Prerequisite knowledge. I assume that you are somewhat familiar with the syntax and semantics of first-order logic as well as with basic notions from computational theory such as (un)decidability, Turing machines, computational complexity, etc [2]. On top of that, it would be great if you knew a bit about logical languages such as Datalog [2] and/or existential rules (but this is not a strict requirement).

Co-encadrants. I have written this proposal using a personal template and hence, everything is written in the first-person of the singular. However, other researchers in my team will also participate in your supervision! Feel free to contact them if you have any questions about the project:

- Pierre Bisquert: Pierre.Bisquert@lirmm.fr
- Federico Ulliana: Federico.Ulliana@lirmm.fr

Contact. I am writing this document in a bit of a haste and hence, it is likely that it is not as clear as I would hope to. To make up for this, I encourage you to contact me if you have any questions about this proposal (my email is under my name in the header of this document). I would be happy to set up a meeting to discuss any further details (technical or otherwise) regarding this proposal.

[2] To know more about Datalog read the corresponding lectures of the course https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2020)/en

2 Preliminaries

The main goal of this research project is to develop an efficient implementation to solve fact entailment over disjunctive Datalog [1] knowledge bases (Definition 2.1). To define this decision problem, we first introduce some preliminary notions:

- We write $\vec{t}$ to denote a list of terms $t_1, \ldots, t_n$. We often identify a list of terms with the corresponding set.
- Given a first-order logic (FOL) formula υ and a list $\vec{x}$ of variables, we write $\upsilon[\vec{x}]$ to indicate that $\vec{x}$ is the set of all free variables that occur in υ. That is, the set of all variables in υ that are not explicitly quantified.
- A *disjunctive Datalog rule* is a FOL formula of the form

  $$\forall \vec{x}. \big(\mathsf{B}[\vec{x}] \rightarrow \mathsf{H}[\vec{z}]\big)$$

  where B is a conjunction of atoms, H is a disjunction of atoms, and $\vec{z} \subseteq \vec{x}$. We often omit universal quantifiers when writing disjunctive Datalog rules and simply refer to these as *rules*.
- A rule is *deterministic* if it features a single atom in its head. That is, a deterministic rule is a rule without disjunctions.
- A *fact* is a FOL formula of the form $P(\vec{c})$ where P is a predicate of arity $|\vec{c}|$ and $\vec{c}$ is a list of constants.
- In the context of this document, a *knowledge base* (KB) is a pair $\langle \mathcal{R}, \mathcal{F} \rangle$ where $\mathcal{R}$ is a rule set and $\mathcal{F}$ is a fact set.

Definition 2.1 (Fact Entailment) *Given a KB $\mathcal{K}$ and a fact φ, we want to determine if $\mathcal{K}$ entails φ under standard first-order logic semantics.*

In order to solve this problem efficiently, we intend to develop a filtration procedure that is used to remove formulas from the knowledge base $\mathcal{K}$ that are not relevant for the considered entailment φ. In the definition of this filtration procedure we make use the *chase algorithm* (Definition 2.2), which is a bottom up (or forward chaining) materialisation procedure used to solve reasoning tasks over deterministic Datalog KBs. To define the chase[3], we must first introduce some preliminary notions:

- A *homomorphism* σ is a function that maps variables into constants.
- Given a FOL formula υ and a homomorphism σ, we write $\sigma(\upsilon)$ to denote the formula that results from replacing each occurrence of a variable x in υ with $\sigma(x)$ if the latter is defined.
- Given a homomorphism σ, a conjunction $\mathcal{A}$ of atoms, and a fact set $\mathcal{F}$, we write $\sigma : \mathcal{A} \to \mathcal{F}$ to indicate that every atom occurring in the conjunction $\sigma(\mathcal{A})$ is in $\mathcal{F}$.
- Given a deterministic rule $R = \mathsf{B} \to \mathsf{H}$ and a fact set $\mathcal{F}$, let $\mathsf{Appl}(R, \mathcal{F})$ denote the (minimal) set of facts that contains all of the facts in $\sigma(\mathsf{H})$ for every homomorphism $\sigma : \mathsf{B} \to \mathcal{F}$.
- For a set $\mathcal{R}$ of deterministic rules and a fact set $\mathcal{F}$, let $\mathsf{Appl}(\mathcal{R}, \mathcal{F}) = \bigcup_{R \in \mathcal{R}} \mathsf{Appl}(R, \mathcal{F})$.

Definition 2.2 (The Chase) *Consider a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{F} \rangle$ where $\mathcal{R}$ is a deterministic rule set (and $\mathcal{F}$ is a fact set). Then, let $Ch_1(\mathcal{K}), Ch_2(\mathcal{K}), \ldots$ be the sequence of fact sets such that*

- $Ch_1(\mathcal{K}) = \mathcal{F}$ *and*
- $Ch_i(\mathcal{K}) = \mathsf{Appl}(\mathcal{R}, Ch_{i-1}(\mathcal{K}))$ *for every* $i \geq 1$.

Moreover, let $\mathsf{Ch}(\mathcal{K}) = \bigcup_{i \geq 1} Ch_i(\mathcal{K})$.[4]

Another notion used in the definition of our filtration procedure is that of the *chase graph*, which is a structure that encodes how facts can be derived during the computation of the chase.

Definition 2.3 (The Chase Graph) *Consider a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{F} \rangle$ where $\mathcal{R}$ is a deterministic rule set, in which all rules have a head restricted to a single atom (and $\mathcal{F}$ is a fact set). Then, $\mathsf{ChGraph}(\mathcal{K}) = \langle V, E \rangle$ is the directed graph such that:*

[3] To know more about this reasoning procedure read the ”The Chase” lecture of the course `https://iccl.inf.tu-dresden.de/web/Database_Theory_(SS2020)/en`

[4] Note that the fact set $\mathsf{Ch}(\mathcal{K})$ is finite since we can only define finitely many facts using the predicates and the constants that occur in $\mathcal{K}$.

- *The set V of vertices is the fact set $\mathsf{Ch}(\mathcal{K})$.*
- *The set E contains an edge $\varphi \rightarrow \phi$ for some $\varphi, \phi \in V$ if there is some rule $R = \mathsf{B} \rightarrow \mathsf{H} \in \mathcal{R}$ and some homomorphism σ such that*
  - *$\sigma : \mathsf{B} \rightarrow \mathsf{Ch}(\mathcal{K})$,*
  - *φ occurs in $\sigma(\mathsf{B})$, and*
  - *ϕ is the only atom in $\sigma(\mathsf{H})$.*

Consider some facts φ and ϕ that occur in the result of the chase of some KB $\mathcal{K}$ (i.e., in $\mathsf{Ch}(\mathcal{K})$). Intuitively, if the edge $\varphi \rightarrow \phi$ is in the chase graph of $\mathcal{K}$ (i.e., in $\mathsf{ChGraph}(\mathcal{K})$), then we know that φ could potentially be used to derive ϕ during the computation of the chase. In turn, if there is no path from φ to ϕ in $\mathsf{ChGraph}(\mathcal{K})$, then φ is irrelevant to determine if ϕ is entailed.

3 The Research Goal: Formal Description

We are finally ready now to describe our filtration procedure! The idea is to, on input $\mathcal{K} = \langle \mathcal{R}, \mathcal{F} \rangle$ a KB and φ a fact, define the fact set $\mathsf{Relevant}(\mathcal{K}, \varphi)$, which is a subset of $\mathcal{F}$ and contains all of the facts that are necessary to determine whether $\mathcal{K}$ entails φ (see Theorem 3.1).

Definition 3.1 *Consider a KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{F} \rangle$ and a fact φ:*

- *For a rule $R = \mathsf{B} \rightarrow \mathsf{H}$, let $R^{\wedge} = \{\mathsf{B} \rightarrow \alpha \mid \alpha \in \mathsf{H}\}$.*
- *For a rule set $\mathcal{R}$, let $\mathcal{R}^{\wedge} = \bigcup_{R \in \mathcal{R}} R^{\wedge}$.*
- *Let $\mathsf{Relevant}(\mathcal{K}, \varphi)$ be the (minimal) set of facts that contains some $\phi \in \mathcal{F}$ if there is a path from ϕ to φ in $\mathsf{ChGraph}(\langle \mathcal{R}^{\wedge}, \mathcal{F} \rangle)$.*

Theorem 3.1 *For a knowledge base $\mathcal{K} = \langle \mathcal{R}, \mathcal{F} \rangle$ and a fact φ,*

$$\mathcal{K} \models \varphi \iff \langle \mathcal{R}, \mathsf{Relevant}(\mathcal{K}, \varphi) \rangle \models \varphi.$$

The above result implies that we can verify if $\mathcal{K} \models \varphi$ by instead checking if $\langle \mathcal{R}, \mathsf{Relevant}(\mathcal{K}, \varphi) \rangle \models \varphi$. The advantage here is that $\mathsf{Relevant}(\mathcal{K}, \varphi)$ may be (in practice) much smaller than the input set of facts $\mathcal{F}$!

Tasks. Having clarified our research goal, we can now discuss the three research tasks that we would like you to solve:

1. Show Theorem 3.1.[5]
2. Produce an implementation that, on input $\mathcal{K}$ a KB and φ a fact, computes the set $\mathsf{Relevant}(\mathcal{K}, \varphi)$.
3. Evaluate our optimisation. That is, empirically study which of the following two methods is more efficient to determine if an input KB $\mathcal{K} = \langle \mathcal{R}, \mathcal{F} \rangle$ entails a given fact φ:
   - Use a reasoner to directly check if $\mathcal{K}$ entails φ.
   - Compute the fact set $\mathsf{Relevant}(\mathcal{K}, \varphi)$ and then use a reasoner to check if $\langle \mathcal{R}, \mathsf{Relevant}(\mathcal{K}, \varphi) \rangle$ entails φ.

4 What now?

If, by this point, you are interested in knowing more about this proposal, just send me an email and let's arrange a meeting. I would like to encourage you to do so even if you could not completely understand everything! Later on, we will determine whether this project is for you or not.

Also, let me finish by stating that the work described in this proposal is only the first step towards a much more ambitious reserach project that involves computing explanations for entailments and probabilistic reasoning. Again, if you are interested, contact me to know more about it!

References

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases.* Addison-Wesley, 1995.

[2] S. Arora and B. Barak. *Computational Complexity: A Modern Approach.* Cambridge University Press, USA, 1st edition, 2009.

[3] M. Krötzsch, F. Simancik, and I. Horrocks. A description logic primer. *CoRR*, 2012.

[5] I will not discuss how to do this here; if you're interested, come see me!