Reformulating Queries: Theory and Practice

Michael Benedikt
University of Oxford

Egor V. Kostylev
University of Oxford

Fabio Mogavero
University of Oxford

Efthymia Tsamoura
Alan Turing Institute &
University of Oxford

Abstract

We consider a setting where a user wants to pose a query against a dataset where background knowledge, expressed as logical sentences, is available, but only a subset of the information can be used to answer the query. We thus want to *reformulate* the user query against the subvocabulary, arriving at a query equivalent to the user's query assuming the background theory, but using only the restricted vocabulary. We consider two variations of the problem, one where we want any such reformulation and another where we restrict the size. We present a classification of the complexity of the problem, then provide algorithms for solving the problems in practice and evaluate their performance.

1 Introduction

A basic problem dealt with in databases and knowledge representation is *query reformulation with background knowledge*. The input to the problem includes a logical formula Q, the *input query*, and a set of sentences of first-order logic Σ, the *background theory* or *integrity constraints*. We also have a specification of an "interface restriction": what data can we actually access? For example, we can specify the allowed data as a sub-vocabulary $\mathcal{V}$ of the vocabulary of Σ and Q. We want to know if the answer to Q can be obtained using the restricted interface. It is well-known that this is the case if and only if there is another formula Q' that is logically equivalent Q over all instances satisfying Σ, and which uses only relations from $\mathcal{V}$. We call such Q' a *reformulation* of Q over Σ in $\mathcal{V}$. We can also restrict Q' to be in some specific logical fragment $\mathcal{L}$, talking about $\mathcal{L}$*-reformulation*.

Variants of the reformulation problem have been considered in several communities.

• Within the database community the emphasis has been on theories given as "dependencies"—that is, Horn sentences in predicate (first-order) logic. The bulk of the work has been on "view-based reformulation": the vocabulary is divided up into the "base relations", and the "view relations" which are derived from the base ones. Each view relation V_i in $\mathcal{V}$ is associated with a defining formula φ_i, and the sentences Σ state that V_i corresponds exactly to its definition: $V_i(\vec{x}) \leftrightarrow \varphi_i(\vec{x})$. The interface relations are then the V_i in $\mathcal{V}$. The problem has been extensively studied in the case where the "view definitions" φ_i are built up from $\wedge, \exists$ (that is, they are *conjunctive queries*, or *CQs*). One desires a target Q' as another CQ, perhaps adding a requirement that Q' contains no redundant atoms. The view-based reformulation problem has been extended to reformulation with dependencies, using the *chase and backchase (C&B) method* [Deutsch *et al.*, 2006; Popa, 2000]. The method proceeds by first generating consequences under the dependences (the *chasing phase*) and then searching for a minimal subset of the consequences that are equivalent to the original formula Q (the *backchasing phase*). The C&B exploits properties specific to dependencies: the full set of consequences can often be generated compactly; further the consequences can be instrumented in such a way as to make the backchasing phase efficient [Meier, 2014; Ileana *et al.*, 2014]. Reformulation over both sub-vocabularies and functional interfaces is considered in [Toman and Weddell, 2011; Benedikt *et al.*, 2016]. In [Toman and Weddell, 2011] the authors describe implementations of reformulation via interpolation and via the C&B method, but the implementation discussion focuses (as with [Meier, 2014; Ileana *et al.*, 2014]) on the case without disjunction. Extracting efficient reformulation from tableaux proofs is the topic of [Hudek *et al.*, 2015], but without complexity bounds or experimental evaluation. In contrast, [Benedikt *et al.*, 2016] deals strictly with theoretical issues, providing complexity bounds on the existence of a reformulation for richer languages such as the guarded fragment, but not for intermediate languages, and not for reformulation with a size bound.

• In the knowledge representation community both the background knowledge and the problem statement have been different. For background theories, the focus has been on *description logics*. The vocabulary is usually restricted to have only unary and binary predicates; but the sentences may allow more flexibility than the Horn fragments considered in the database community, including both disjunction and negation. The emphasis has not been on reformulation as defined above, where we desire a target formula that gives exactly the same results as the input query on an instance satisfying the background theory. Instead the emphasis is on converting theories in one vocabulary to theories in a smaller vocabulary, preserving consequences [Cate *et al.*, 2006; Konev *et al.*, 2010], and in getting the best upper approximation of

a formula that uses a given vocabulary [Lang *et al.*, 2003; Lutz and Wolter, 2011; Koopmann and Nikitina, 2017; Konev *et al.*, 2009; Lutz *et al.*, 2012; Nikitina and Rudolph, 2014].

In this paper our input will be a theory representing background knowledge and a conjunctive query (CQ). We look for a formula that is *exactly* equivalent relative to the background theory, as in the database work above. Thus the tools will be quite different from works considering the "best approximation" of a theory as in [Lutz and Wolter, 2011; Koopmann and Nikitina, 2017; Lutz and Wolter, 2011; Konev *et al.*, 2009; Lutz *et al.*, 2012; Nikitina and Rudolph, 2014]. However, unlike work in the database community, we consider theories that allow disjunction and negation; this is motivated by the usefulness of these operators in modelling the relationship of local and global schemas (e.g. in "global-as-view mappings") within data integration. We distinguish the problem of determining if *some* reformulation exists from the problem of finding one that is within a certain *size bound*; the latter problem has not been studied even in the view-based case. Both problems can be attacked by a reduction to entailment. For finding some reformulation there is a reduction using interpolation in theorem proving, dating back to the birth of interpolation [Craig, 1957b]. For size-bounded reformulation one can use a "guess-and-check" method that is analogous to the C&B, but accommodating disjunction and negation in the theories. Surprisingly, we give lower bounds showing that both these approaches provide optimal complexity for their respective problems, over a range of logics and targets for reformulation.

After completing an examination of the complexity of reformulation in Section 3, we turn to practice in Section 4. We discuss several ways of implementing the approaches of Section 3, focusing on an approach using a new variant of Huang's interpolation algorithm for resolution [Huang, 1995]. We perform an extensive experimental comparison of the "optimized Huang" approach on top of an existing theorem prover with several alternatives, including both an extension of C&B to disjunction (based on [Deutsch *et al.*, 2008]) and an approach using an interpolation algorithm that is integrated with a theorem prover. *Our work is the first look at query reformulation over a sub-vocabulary for logics that include disjunction in arbitrary arity, either either from the point of view of complexity or experimental evaluation.*

Organization. Section 3 studies the complexity of reformulation, considering both propositional and predicate logic theories. Section 4 turns to reformulation in practice, outlining our implementation strategy, experimental testbed, and experimental results. Conclusions are in Section 5.

2 Reformulation Problems

Given a fragment of first-order logic $\mathcal{L}$, called *target* fragment, a Boolean conjunctive query (CQ) Q, the *input query*, a finite set Σ of first-order sentences, *the background theory*, and a sub-vocabulary $\mathcal{V}$ of the vocabulary of Q and Σ, an *$\mathcal{L}$-reformulation* of Q over Σ in $\mathcal{V}$ is a sentence in $\mathcal{L}$ over $\mathcal{V}$ that is logically equivalent Q over all instances satisfying Σ.

The first, most basic problem we consider is the *reformulation existence problem* for a target fragment $\mathcal{L}$, denoted $\exists\mathsf{REF}_{\mathcal{L}}(Q, \Sigma, \mathcal{V})$, which is true for an input query Q, background theory Σ, and sub-vocabulary $\mathcal{V}$ if and only if there exists an $\mathcal{L}$-reformulation of Q over Σ in $\mathcal{V}$.

The *size-restricted reformulation problem* for a target fragment $\mathcal{L}$, $\mathsf{REF}^{\leq}_{\mathcal{L}}(Q, \Sigma, \mathcal{V}, k)$, has an additional integer argument k, written in unary; it is true whenever there is a $\mathcal{L}$-reformulation of Q over Σ in $\mathcal{V}$ of size at most k, where the *size* is the length of a binary representation.

We consider several variants of these two problems, which are parametrised along the following dimensions.

First, besides arbitrary vocabularies, we consider vocabularies with bounded arity and propositional vocabularies—that is, the setting when all relations are nullary. Second, target fragment $\mathcal{L}$ can be one of the following: (Boolean) *conjunctive queries* CQ—that is, existentially quantified conjunctions of atoms (i.e., propositions in the propositional setting), unions of conjunctive queries UCQ—that is, disjunctions of Boolean CQs, positive existentially quantified sentences POS, and all (first-order or propositional) sentences ALL. Note that in the propositional case quantification is irrelevant. For $\exists\mathsf{REF}$ there is no difference between POS and UCQ, because these have the same expressiveness; we thus talk of *monotone reformulation* in both cases.

Finally, our basic language for Σ is the language of *tuple generating dependencies* (*TGDs*)—that is, (implicitly universally quantified) sentences of the form $\bigwedge_i R_i(\vec{x}) \rightarrow \exists\vec{y}\ S(\vec{x}, \vec{y})$. Most reasoning problems concerning arbitrary TGDs are undecidable, so in the general setting we consider the following two restrictions: *full* TGDs—that is TGDs without existentially quantified variables $\vec{y}$ on the right, and *weakly-acyclic* (sets of) TGDs—that is, TGDs such that none of them can fire unboundedly often (see [Fagin *et al.*, 2005] for a precise definition). Besides this, we also investigate the theories consisting of *full disjunctive TGDs*—that is, sentences of the form $\bigwedge_i R_i(\vec{x}) \rightarrow \bigvee_j S_j(\vec{x})$. These sentences arise naturally in data integration, both to model bidirectional relationships between a local and global schema and to capture datatype restrictions. Note that although the syntax of full disjunctive TGDs matches disjunctive Datalog [Eiter *et al.*, 1997], the semantics is classical entailment, different from the one in [Eiter *et al.*, 1997]. In the propositional setting, (full) TGDs are just Horn formulas; hence, we also refer to full disjunctive TGDs as *disjunctive Horn* formulas. Finally, in this setting we also consider the general case where Σ ranges over all sets of propositional formulas.

3 Complexity of Reformulation

We now investigate the reformulation problems in theory. All upper bounds follow from reduction to entailment for the theory in question. The lower bounds require more effort.

3.1 Complexity Of $\exists\mathsf{REF}$

The fact that the $\exists\mathsf{REF}$ problem for both general reformulation and monotone reformulation can be reduced to entailments from interpolation, dates back to work of Craig [Craig, 1957b]. For any formula φ, let φ' be formed by replacing any relation R by a fresh copy R'. Then, given a vocabulary $\mathcal{V}$, let $\mathsf{ForwAx}_{\mathcal{V}} = \bigwedge_{R\in\mathcal{V}} \forall\vec{x}\, R(\vec{x}) \rightarrow R'(\vec{x})$ and

background theory	vocabulary: any	vocabulary: fixed arity	vocabulary: prop.
full TGDs	ExpTime	NP	P
w.-a. TGDs	2ExpTime	2ExpTime	P
full disj. TGDs	coNExpTime / undecidable	Π_2^p / undecidable	coNP
prop. sentences	–	–	coNP

Table 1: Summary of results for $\exists$REF: all bounds are tight and all hold for any target fragment $\mathcal{L}$ among ALL, POS, UCQ and CQ, except those with '/' in which case the second bound is for CQ and the first is for the others.

background theory	vocabulary: any	vocabulary: fixed arity	vocabulary: prop.
full TGDs	ExpTime	NP	NP
w.-a. TGDs	2ExpTime	2ExpTime	NP
full disj. TGDs	coNExpTime	Σ_3^p	Σ_2^p
prop. sentences	–	–	Σ_2^p

Table 2: Summary of results for $\mathsf{REF}^{\leq}$: all bounds are tight and all hold for any target fragment $\mathcal{L}$ among ALL, POS, UCQ and CQ, except Σ_3^p, which is tight for POS, UCQ and CQ, but only known to be an upper bound for ALL.

$\mathsf{BackAx}_{\mathcal{V}} = \bigwedge_{R\in\mathcal{V}} \forall\vec{x}\, R'(\vec{x}) \rightarrow R(\vec{x})$. From [Craig, 1957b] and [Lyndon, 1959] we have the following proposition.

Proposition 1. *Given an input query Q, theory Σ and sub-vocabulary $\mathcal{V}$, $\exists\mathsf{REF}_{\mathsf{ALL}}(Q,\Sigma,\mathcal{V})$ is true if and only if $\Sigma \wedge Q \wedge \mathsf{ForwAx}_{\mathcal{V}} \wedge \mathsf{BackAx}_{\mathcal{V}} \wedge \Sigma' \wedge \neg Q'$ is not satisfiable; $\exists\mathsf{REF}_{\mathsf{POS}}(Q,\Sigma,\mathcal{V})$ is true if and only if $\Sigma \wedge Q \wedge \mathsf{ForwAx}_{\mathcal{V}} \wedge \Sigma' \wedge \neg Q'$ is not satisfiable.*

In these works it is also shown that reformulations can be read off from a proof of unsatisfiability in a suitable proof system, using an interpolation algorithm. We discuss this further in Section 4. For now we note that this immediately provides upper bounds when Σ is restricted to range over theories with decidable entailment. For example, in the propositional setting the following bounds for general and monotone reformulations follow from the same bounds for entailment.

Corollary 1. *In the propositional setting, problems $\exists\mathsf{REF}_{\mathsf{ALL}}(Q,\Sigma,\mathcal{V})$ and $\exists\mathsf{REF}_{\mathsf{POS}}(Q,\Sigma,\mathcal{V})$ are in* coNP *as Σ ranges over either all (propositional) or disjunctive Horn formulas. They are in* P *as Σ ranges over Horn formulas.*

The same approach can be applied to first-order predicate logic. For full TGDs, it is well-known that entailment is in ExpTime, and becomes NP if we fix the arity of the vocabulary. If we consider weakly-acyclic TGDs, then entailment is 2ExpTime-complete [Calì *et al.*, 2010], no matter whether the arity is bounded or not. For full disjunctive TGDs, we make use of the following folklore fact: entailment problem for full disjunctive TGDs is coNExpTime-complete; it is Π_2^p-complete when the arity of the vocabulary is bounded.

Thus, Craig's reduction above gives the following bounds.

Corollary 2. *Problems $\exists\mathsf{REF}_{\mathsf{ALL}}(Q,\Sigma,\mathcal{V})$ and $\exists\mathsf{REF}_{\mathsf{POS}}(Q,\Sigma,\mathcal{V})$ are in* ExpTime *and in* NP *for bounded arity as Σ ranges over full TGDs. They are in* 2ExpTime *for weakly-acyclic TGDs. They are* coNExpTime *and in Π_2^p for bounded arity for full disjunctive TGDs.*

Craig's reduction does not provide any information about reformulating when the target does not allow for disjunction—that is, when $\mathcal{L}$ is CQ. However, in the propositional setting, there is a simpler reduction of $\exists\mathsf{REF}_{\mathsf{CQ}}$ to entailment: letting S be the propositions in $\mathcal{V}$ that are entailed by $Q \wedge \Sigma$, $\exists\mathsf{REF}_{\mathsf{CQ}}(Q,\Sigma,\mathcal{V})$ holds exactly if $S \wedge \Sigma$ entails Q.

Proposition 2. *The bounds in Corollary 1 hold also for $\exists\mathsf{REF}_{\mathsf{CQ}}(Q,\Sigma,\mathcal{V})$.*

In the non-propositional setting, it is known that if a CQ Q has a UCQ-reformulation over TGDs Σ in a sub-vocabulary $\mathcal{V}$ then it has a CQ-reformulation as well. Indeed, one disjunct in the UCQ-reformulation must be a CQ-reformulation.

Proposition 3. *The bounds in Corollary 2 on full and weakly-acyclic TGDs hold also for $\exists\mathsf{REF}_{\mathsf{CQ}}(Q,\Sigma,\mathcal{V})$.*

For full disjunctive TGDs, disjunction in the target fragment may be essential in the reformulation; thus in this case we cannot make use of either Craig's reduction or a reduction that considers each atom at a time. In fact, we can show the following.

Theorem 1. *$\exists\mathsf{REF}_{\mathsf{CQ}}(Q,\Sigma,\mathcal{V})$ is undecidable when Σ ranges over full disjunctive TGDs even if the arity is bounded.*

The proof is by reduction of the Datalog containment problem. Intuitively, solving $\exists$REF requires us to be able to decide if a union Q of CQs is equivalent to a CQ Q' with respect to a set of full TGDs Σ, and this is the same as checking equivalence of the Datalog programs $\Sigma \cup \{Q \rightarrow \mathsf{Goal}\}$ and $\Sigma \cup \{Q' \rightarrow \mathsf{Goal}\}$, for a fresh nullary predicate Goal.

Table 1 summarizes our bounds for $\exists$REF. The matching lower bounds can be easily shown by a reduction from entailment for the corresponding background theories.

3.2 Complexity Of $\mathsf{REF}^{\leq}$

Problem $\mathsf{REF}^{\leq}$ does not have a similar direct reduction to entailment. However, one can reduce it to *"guessing plus entailment"*: the machine first guesses Q' and then checks if Q' is a reformulation using an oracle for entailment. For example, from this technique we get the following upper bounds for propositional setting.

Proposition 4. *In the propositional setting, problems $\mathsf{REF}^{\leq}_{\mathcal{L}}(Q,\Sigma,\mathcal{V},k)$, for $\mathcal{L} \in \{\mathsf{ALL},\mathsf{POS},\mathsf{UCQ},\mathsf{CQ}\}$, are all in Σ_2^p as Σ ranges over either all (propositional) or disjunctive Horn formulas. They are in* NP *for Horn formulas.*

Outside of the propositional setting, the guessing phase is often negligible on top of entailment; the only exception is the case of vocabulary of bounded arity and theories consisting of full disjunctive TGDs, in which the phase adds a level in the polynomial hierarchy.

Proposition 5. *Problems $\mathsf{REF}^{\leq}_{\mathcal{L}}(Q,\Sigma,\mathcal{V},k)$, for $\mathcal{L} \in \{\mathsf{ALL},\mathsf{POS},\mathsf{UCQ},\mathsf{CQ}\}$, are in* ExpTime *and in* NP *for bounded arity as Σ ranges over full TGDs. They are in* 2ExpTime *for*

weakly-acyclic TGDs. They are CONEXPTIME *and in* Σ_3^p *for bounded arity for full disjunctive TGDs.*

The upper bounds provided by the guess-and-check technique turn out to be almost always tight, but unlike for $\exists$REF, this is not obvious for the Σ_2^p bound in Proposition 4 and the Σ_3^p bound in Proposition 5 —that is, when the complexity of $\mathsf{REF}^{\leq}$ differs from the complexity of entailment.

In the propositional setting, for $\mathsf{REF}^{\leq}_{\mathsf{ALL}}(Q, \Sigma, \mathcal{V}, k)$ where Σ ranges over arbitrary propositional formulas, the Σ_2^p lower bound follows from results on minimization of Boolean formulas in [Umans, 2001]. Indeed, using related results [Buchfuhrer and Umans, 2011] one can show hardness for the variant of the problem where we fix the number of alternations of $\vee$ and $\wedge$. However, the technique of [Buchfuhrer and Umans, 2011] requires formulas with hard satisfiability problems, so it is not applicable when formulas in Σ are not arbitrary and always satisfiable, such as disjunctive Horn formulas. Thus in this case we need a "native" reduction for Σ_2^p-hardness.

Theorem 2. *In the propositional setting,* $\mathsf{REF}^{\leq}_{\mathcal{L}}(Q, \Sigma, \mathcal{V}, k)$*, for* $\mathcal{L} \in \{\mathsf{POS}, \mathsf{UCQ}, \mathsf{CQ}\}$*, is* Σ_2^p*-hard when* Σ *ranges over sets of disjunctive Horn formulas.*

Sketch. The proof is by reduction of $\exists\forall$3SAT problem. Suppose $\varphi = \exists\vec{u}\,\forall\vec{v}\,\neg\psi$ is formula with $\vec{u}$ and $\vec{v}$ tuples of propositional variables, and ψ a conjunction of clauses $\ell^1 \vee \ell^2 \vee \ell^3$ with each ℓ^j a variable from $\vec{u} \cup \vec{v}$ or its negation.

We construct a set of propositional disjunctive Horn formulas Σ and a set of propositional variables $\mathcal{V}$ such that φ holds if and only if there exists a positive formula Q' over $\mathcal{V}$ with $k = |\vec{u}|$ occurrences of variables such that Q' is equivalent to a propositional variable Q over Σ (for simplicity, we assume here that the size of Q' is the number of occurrences). The vocabulary $\mathcal{V}$ consists of variables TrueX_i and FalseX_i, for each u_i in $\vec{u}$, while Σ guarantees the following:

1. Q implies all the variables in $\mathcal{V}$, so the equivalence of Q and Q' over Σ boils down to the containment of Q' in Q;
2. containment of Q' in Q over Σ is only possible if Q' implies either TrueX_i or FalseX_i for each u_i; since the size of Q' is bounded by k, it means that Q' can only be of the form $\mathsf{AsX}_1 \wedge \cdots \wedge \mathsf{AsX}_k$, for $\mathsf{AsX}_i \in \{\mathsf{TrueX}_i, \mathsf{FalseX}_i\}$— that is, Q' corresponds to an assignment of variables u_i;
3. if a set of propositions and Σ imply TrueX_i or FalseX_i for each u_i, then they imply either TrueY_i or FalseY_i for each universally quantified v_i in $\vec{v}$, which corresponds to an assignment of $\vec{v}$;
4. if one of the minimal extensions of a set of propositions satisfying Σ ("disjunctive chase models") *falsifies* a clause in ψ under this encoding, which corresponds to falsifying the whole of ψ, then this extension implies Q.

These guarantees imply the statement of the theorem. □

Moving outside of propositional logic, there is a mismatch between the Σ_3^p upper bound from Proposition 4 and Π_2^p lower bound inherited from the entailment in the case when the arity of the vocabulary is bounded and Σ ranges over full disjunctive TGDs. For positive $\mathcal{L}$, we show Σ_3^p-hardness via a direct reduction of the $\exists\forall\exists$3SAT problem (and leave the case of ALL open). The reduction uses a similar idea to the Σ_2^p-reduction in the propositional case, but it is more involved: the inner existential quantification is encoded by a homomorphism-existence check from a candidate Q' to Q.

Theorem 3. *Problem* $\mathsf{REF}^{\leq}_{\mathcal{L}}(Q, \Sigma, \mathcal{V}, k)$*, for* $\mathcal{L} \in \{\mathsf{POS}, \mathsf{UCQ}, \mathsf{CQ}\}$*, is* Σ_3^p*-hard if* Σ *ranges over sets of full disjunctive TGDs and the arity is bounded by* 2*.*

Table 2 summarizes our bounds for $\mathsf{REF}^{\leq}$. The matching lower bounds for the cases not considered in Theorems 2 and 3 can be easily shown by a reduction of entailment for the corresponding background theories.

4 Implementing Reformulation

In this section, we develop and evaluate three approaches for finding a reformulation. We concentrate on target fragments with disjunction—that is, leaving CQ aside. We start with a description of the approaches in Section 4.1 and then proceed to their evaluation in Section 4.2.

4.1 Reformulation Algorithms

Two of our three approaches are based on Proposition 1, reducing reformulation to Craig and Lyndon interpolations.

A *Craig interpolant* for an entailment $\varphi_1 \models \varphi_2$ of first-order formulas φ_1 and φ_2 is a formula ψ over relations common to φ_1 and φ_2 such that $\varphi_1 \models \psi$ and $\psi \models \varphi_2$. A *Lyndon interpolant* further has the property that a relation appears positively in ψ only if it appears positively in both φ_1 and φ_2, and similarly for appearing negatively.

It is shown in [Craig, 1957a] that first-order Craig interpolants exist for all first-order entailments, while [Lyndon, 1959] showed the same for Lyndon interpolants. By the argument of [Craig, 1957b], a Craig interpolant for the entailment

$$\mathsf{BackAx}_{\mathcal{V}} \wedge \Sigma \wedge Q \models \mathsf{ForwAx}_{\mathcal{V}} \wedge \Sigma' \rightarrow Q',$$

where $\mathsf{BackAx}_{\mathcal{V}}$, $\mathsf{ForwAx}_{\mathcal{V}}$, Σ' and Q' are as in Section 3.1, can be converted to a reformulation of a CQ Q over a theory Σ in a sub-vocabulary $\mathcal{V}$; conversely the entailment is also a necessary condition for such a reformulation to exist. A variation of the argument shows that a Lyndon interpolant for

$$\Sigma \wedge Q \models \mathsf{ForwAx} \wedge \Sigma' \rightarrow Q'$$

is itself a monotone reformulation of Q, and the entailment is likewise necessary. Thus reformulation using Craig's technique requires finding proofs witnessing the appropriate entailment, and then applying an interpolation procedure.

We implemented two approaches for finding monotone reformulation following this high-level strategy, both using existing state-of-the-art theorem provers. The first approach is based on an interpolation module native to a prover, which is used as a "black box" to get reformulations. In our implementation we use Vampire, a prover with the most robust built-in support for interpolation [Hoder *et al.*, 2010; 2012]. Unfortunately, it is not clear if these interpolants are Lyndon, so resulting reformulations might not be monotone.

The second approach is based on our own implementation of interpolation, on top of resolution theorem proving. This approach has several variations, as described next.

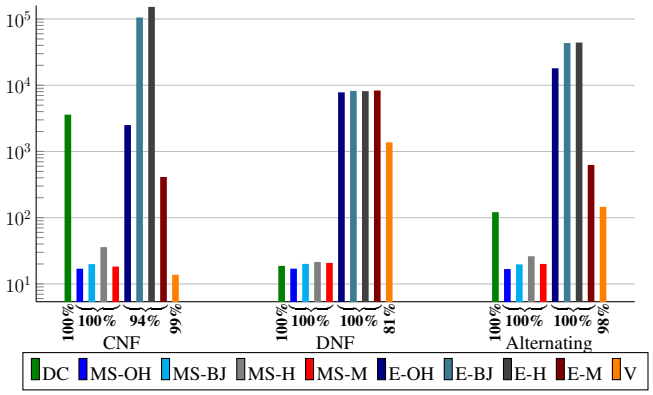

Figure 1: Evaluation of reformulation approaches.

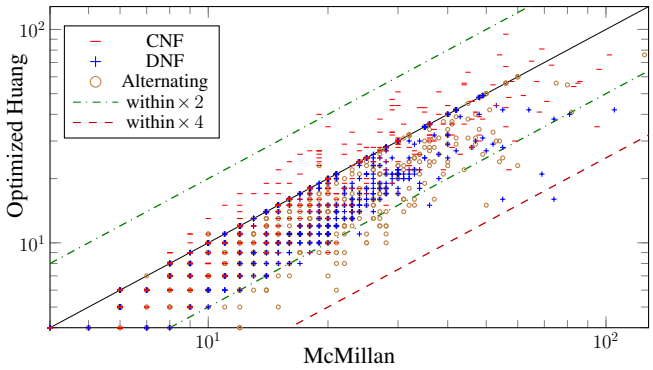

Figure 2: Comparison between optimized Huang and McMillan interpolation algorithms on top of MathSat.

There exist a number of algorithms that extract an interpolation from a resolution-based proof. They proceed in a top-down manner, creating a "partial interpolant" on each step of resolution, and the complete interpolant is the formula associated with the final contradiction in the resolution proof. The Huang ([Huang, 1995]) and Bonacina-Johansson ([Bonacina and Johansson, 2011]) algorithms are similar; the base cases assign either True or False to input clauses, while the inductive step for resolution may introduce new atoms. In contrast, the McMillan algorithm ([McMillan, 2003]) has a more complex base case, while the propagation is purely compositional.

It is desirable to produce a monotone reformulation, for which we need an implementation of Lyndon interpolation. The Bonacina-Johansson algorithm has this property, but Huang and McMillan have not. We developed a variant of Huang algorithm, called *optimized Huang*, that addresses this, also performing optimizations that we found crucial to produce good-quality reformulations. Next we briefly describe our algorithm, leaving details for the full version.

Consider an entailment $\varphi_1 \models \varphi_2$, where φ_1 and φ_2 are first-order formulas. In a resolution proof of this entailment, we search for a contradiction from the CNF of $\varphi_1 \wedge \neg\varphi_2$ by means of resolving of clauses. For brevity, we assume that φ_1 and $\neg\varphi_2$ are already in CNF—that is, conjunctions of clauses (the general case can be handled via skolemization, as described in [Huang, 1995]). A resolution proof is a DAG, with nodes representing clauses derived within the proof. In particular, source nodes are clauses of φ_1 and $\neg\varphi_2$, each intermediate node γ has two clauses that resolve to γ as parents, and the drain is the contradiction (we assume that factoring is incorporated into the resolution steps).

For purposes of constructing interpolants, the literals in the nodes are first annotated with a *provenance*, which is either LEFT or RIGHT. The literals in the source clauses of φ_1 have provenance LEFT and the literals of $\neg\varphi_2$ have RIGHT; inductively, in the clause resulting from a resolution step, each literal inherits the provenance of the corresponding literal in one of the parents. After the provenance, we inductively assign partial interpolants to each clause. Following [Huang, 1995], we assign False to the root nodes corresponding to the clauses in φ_1, and True to the root nodes corresponding to $\neg\varphi_2$. For a non-root node γ with parents γ_1 and γ_2, we form the partial interpolant ψ as follows. Suppose the partial interpolants assigned to γ_1 and γ_2 are ψ_1 and ψ_2, respectively. Suppose also the literals involved in the resolution are $P(\vec{t})$ in γ_1 and $\neg P(\vec{s})$ in γ_2, and the provenances of these literals are DIR_1 and DIR_2, respectively. Let the unifier be σ. Then

- if $\text{DIR}_1 = \text{DIR}_2 = \text{LEFT}$ then $\psi = (\psi_1 \vee \psi_2)\sigma$;
- if $\text{DIR}_1 = \text{DIR}_2 = \text{RIGHT}$ then $\psi = (\psi_1 \wedge \psi_2)\sigma$;
- if $\text{DIR}_1 = \text{LEFT}$ and $\text{DIR}_2 = \text{RIGHT}$ then $\psi = (\psi_1 \vee (\psi_2 \wedge P(\vec{t})))\sigma$; and
- if $\text{DIR}_1 = \text{RIGHT}$ and $\text{DIR}_2 = \text{LEFT}$ then $\psi = ((\psi_1 \wedge \neg P(\vec{s})) \vee \psi_2)\sigma$.

The distinction from [Huang, 1995] is that [Huang, 1995] collapses the last two cases into one, producing $((\psi_1 \wedge \neg P(\vec{s})) \vee (\psi_2 \wedge P(\vec{t})))\sigma$ as the partial interpolant in both cases. This is the reason why the original algorithm lacks the Lyndon property. Our modification is inspired by [Bonacina and Johansson, 2011].

In the above algorithm, we assume that factoring and absorption rules are applied silently at each inductive step. In our implementation we further apply several optimisations. For example, if P is a proposition, then we simplify the second-to-last case by producing $(\psi_1 \vee (\psi_2' \wedge P))\sigma$, where ψ_2' substitutes True for occurrences of P and then simplifies by absorption; in the last case, False substitutes P in ψ_1'.

Proposition 6. *The interpolant computed by the optimised Huang algorithm is a Lyndon interpolant.*

We implemented the Huang, optimized Huang, Bonacina-Johansson, and McMillan algorithms for finding interpolation. All of these procedures require resolution proofs as inputs. For the propositional case, there are several tools producing such proofs, and we focused on one of them, namely MathSat [Cimatti *et al.*, 2013]. To the best of our knowledge, the only first-order theorem prover that produces sufficiently detailed resolution proofs is E [Schulz, 2013], so we integrated it into our system as well.

Our third approach, which is complete only for the propositional case and when the background theories are disjunctive Horn formulas, does not rely on interpolation to find an reformulation. Instead, it relies on a variant of the "disjunctive chase" technique of [Deutsch *et al.*, 2008], making use of the disjunctive logic programming system DLV [Leone *et al.*,

2006]. Given an input CQ Q, disjunctive Horn formulas Σ and sub-vocabulary $\mathcal{V}$, we generate, using DLV, all minimal models of $Q \wedge \Sigma$ by chasing Q with Σ. For each such model we consider the conjunction of all its propositional letters that are in $\mathcal{V}$, and our candidate reformulation Q' is the disjunction of these conjunctions. Then, we check that Q' implies Q under Σ, again using DLV.

4.2 Evaluation Of Reformulation Approaches

We tested the approaches for finding reformulation described in the previous section. We concentrate on the propositional case with disjunctive Horn formulas in background theories. For the target language, we consider general and monotone reformulations—that is, classes ALL and POS. However, some approaches (e.g., Vampire-based) cannot guarantee a monotone reformulation even if it exists, so we always allow these approaches to return reformulations with negation.

Our testing infrastructure consists of two components: the generator of reformulation problems and the evaluator of the implemented approaches on the outputs of the generator.

The generator has two parts. The first part randomly generates a canonical target reformulation Q' in one of the following six classes: arbitrary or positive propositional formulas either in CNF, or in DNF, or which are alternations of conjunctions and disjunctions of literals with alternation depth varying from 2 to 5. The second part takes reformulation Q' as input and outputs a CQ Q, disjunctive Horn formulas Σ, and sub-vocabulary $\mathcal{V}$ such that Q' is a reformulation of Q over Σ in $\mathcal{V}$. The tool generates sentences that follow the structure of Q' to witness that Q implies Q'; for example, if Q' is $B \wedge (A_1 \vee A_2)$, then Σ contains clauses equivalent to $Q \to C$, $C \to B$, $C \to D$, $D \to A_1 \vee A_2$, and additionally some "noise clauses" (controlled by an input parameter).

The second component of the testing infrastructure runs our implementations of the reformulation approaches on the output of the generator and measures their running times and the sizes of the output reformulations. In particular, we tested ten variants of the approaches, based on one of the following:

- native interpolation of Vampire 4.1 for CASC J8;
- one of the Huang, optimized Huang, Bonacina-Johansson, and McMillan algorithms applied to the proofs constructed by either MathSat or E resolution-based theorem provers;
- disjunctive logic programming system DLV.

Note that in all settings the existence of the reformulation is guaranteed (because the canonical Q' is such), but the tested procedures do not know this and do not have an access to Q'.

We report the results of our experiments with more than 3600 runs in our infrastructure, more than 600 for each of the six classes for the canonical target reformulation.

We first evaluate the running time of the procedures. Since the interpolation algorithms themselves are linear, the performance of the interpolation-based approaches is dominated by the time of theorem proving. The MathSat-based procedure was fastest at proving, requiring only few milliseconds. The DLV-based procedure was next, always terminating within 86 seconds. We timed out any other computation within 5 times of the running time of the DLV-based procedure, since the latter gives an idea of the complexity of brute-force search. The percentage of terminating runs is given at bottom of Fig. 1.

Next we briefly summarise the results on the size of reformulations, deferring details for the full version, and the github repository of [Benedikt *et al.*, 2017]. The bars in Figure 1 show the average size of output reformulations for all 10 approaches when the canonical target reformulation is in CNF, in DNF, or an alternation of conjunctions and disjunctions. Note that only terminating runs are taken into account.

The DLV-based approach always generates reformulations in DNF, so it is not surprising that it performs badly in cases where the canonical reformulation is not in DNF. Even in the case of DNF, it is slightly inferior to the optimised Huang algorithm on top of MathSat; intuitively the disjunctive chase fails to identify succinct representations achieved by leveraging repetition of atoms across different minimal models. The Vampire-based approach failed to find a reformulation within the timeout in a significant percentage of the cases, and further the size of reformulation was, in average, poor. Due to the "black-box" nature of this approach we can only speculate on the reason. However, we note that the interpolation procedures of Vampire are geared towards examples from verification [Hoder *et al.*, 2010; 2012], and our experiments can be seen as indicating a strong difference between interpolation for reformulation and verification.

We can also observe the superiority of MathSat over E, as well as of the optimized Huang and McMillan algorithms over the others. Figure 2 contains a finer comparison of these two algorithms on top of MathSat, which shows that the optimized Huang is superior on most examples. Our experiments also show that the McMillan algorithm is much more robust to inefficiencies of theorem provers—that is, even with long proofs it can produce relatively smaller reformulations. This may be explained by the compositionality of the McMillan algorithm, while the optimized Huang introduces literals at resolution steps, inducing redundancy in the resulting formula.

5 Conclusion

Our theoretical results shows that the most straightforward algorithms for $\exists$REF and REF$^{\leq}$ provide the optimal complexity. While $\exists$REF can be polynomially reduced to entailment over the corresponding logic, our lower bounds show that REF$^{\leq}$ requires an additional layer of non-determinism. There is no obvious implementation other than brute-force for this. In our experiments we focused on the interpolation-based approach, showing that for propositional theories good performance can be achieved with an optimized resolution-based interpolation algorithm on top of a high-performance theorem-prover. Limitations in the proofs exposed by first-order theorem-provers currently represent a bottleneck to application of this approach to richer logics. An interesting direction is to allow the search heuristics of a theorem-prover to be guided by the size of partial-interpolants, thus giving "interpolant-driven theorem-proving". We are examining the implementation of this over open-source provers such as E.

Acknowledgments

This work is funded by the EPSRC grants EP/M005852/1, EP/N014359/1, EP/L012138/1, and EP/N510129/1.

References

[Benedikt *et al.*, 2016] Michael Benedikt, Balder Ten Cate, Julien Leblay, and Efthymia Tsamoura. *Generating plans from proofs: the interpolation-based approach to query reformulation*. Morgan Claypool, 2016.

[Benedikt *et al.*, 2017] Michael Benedikt, Egor V. Kostylev, Fabio Mogavero, and Efthymia Tsamoura. Query reformulation: Theory and practice, 2017. Available at `www.github.com/qreform/ijcai17qreform`.

[Bonacina and Johansson, 2011] Maria Paola Bonacina and Moa Johansson. On interpolation in decision procedures. In *Automated Reasoning with Analytic Tableaux and Related Methods*. 2011.

[Buchfuhrer and Umans, 2011] David Buchfuhrer and Christopher Umans. The complexity of boolean formula minimization. *J. Comput. Syst. Sci.*, 77(1):142–153, 2011.

[Calì *et al.*, 2010] Andrea Calì, Georg Gottlob, and Andreas Pieris. Query Answering under Non-guarded Rules in Datalog+/-. In *Web Reasoning and Rule Systems*, 2010.

[Cate *et al.*, 2006] Balder ten Cate, Willem Conradie, Maarten Marx, and Yde Venema. Definitorially complete description logics. In *KR*, 2006.

[Cimatti *et al.*, 2013] Alessandro Cimatti, Alberto Griggio, Bastiaan Joost Schaafsma, and Roberto Sebastiani. The MathSAT5 SMT solver. In *TACAS*, 2013.

[Craig, 1957a] William Craig. Linear reasoning. A new form of the Herbrand-Gentzen theorem. *The Journal of Symbolic Logic*, 22(03):250–268, 1957.

[Craig, 1957b] William Craig. Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *Journal of Symbolic Logic*, 22(3):269–285, 1957.

[Deutsch *et al.*, 2006] Alin Deutsch, Lucian Popa, and Val Tannen. Query reformulation with constraints. *SIGMOD Record*, 35(1):65–73, 2006.

[Deutsch *et al.*, 2008] Alin Deutsch, Alan Nash, and Jeff Remmel. The chase revisited. In *PODS*, 2008.

[Eiter *et al.*, 1997] Thomas Eiter, Georg Gottlob, and Heikki Mannila. Disjunctive Datalog. *ACM Trans. Database Syst.*, 22(3):364–418, 1997.

[Fagin *et al.*, 2005] Ronald Fagin, Phokion G. Kolaitis, Renee J. Miller, and Lucian Popa. Data exchange: Semantics and query answering. *Theoretical Computer Science*, 336(1):89–124, 2005.

[Hoder *et al.*, 2010] Kryštof Hoder, Laura Kovács, and Andrei Voronkov. *Interpolation and Symbol Elimination in Vampire*. 2010.

[Hoder *et al.*, 2012] Krystof Hoder, Andreas Holzer, Laura Kovács, and Andrei Voronkov. Vinter: A Vampire-Based tool for interpolation. In *APLAS*, 2012.

[Huang, 1995] Guoxiang Huang. Constructing Craig interpolation formulas. In *Computing and Combinatorics*. 1995.

[Hudek *et al.*, 2015] Alexander K. Hudek, David Toman, and Grant E. Weddell. On enumerating query plans using analytic tableau. In *TABLEAUX*, 2015.

[Ileana *et al.*, 2014] Ioana Ileana, Bogdan Cautis, Alin Deutsch, and Yannis Katsis. Complete yet practical search for minimal query reformulations under constraints. In *SIGMOD*, 2014.

[Konev *et al.*, 2009] Boris Konev, Dirk Walther, and Frank Wolter. Forgetting and uniform interpolation in large-scale description logic terminologies. In *IJCAI*, 2009.

[Konev *et al.*, 2010] Boris Konev, Carsten Lutz, Denis K. Ponomaryov, and Frank Wolter. Decomposing description logic ontologies. In *KR*, 2010.

[Koopmann and Nikitina, 2017] Patrick Koopmann and Nadeschda Nikitina. Small is beautiful: Computing minimal equivalent EL concepts. In *AAAI*, 2017.

[Lang *et al.*, 2003] Jérôme Lang, Paolo Liberatore, and Pierre Marquis. Propositional independence: Formula-variable independence and forgetting. *J. Artif. Intell. Res. (JAIR)*, 18:391–443, 2003.

[Leone *et al.*, 2006] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. The DLV system for knowledge representation and reasoning. *TOCL*, 7(3):499–562, 2006.

[Lutz and Wolter, 2011] Carsten Lutz and Frank Wolter. Foundations for uniform interpolation and forgetting in expressive description logics. In *IJCAI*, 2011.

[Lutz *et al.*, 2012] Carsten Lutz, Inanç Seylan, and Frank Wolter. An automata-theoretic approach to uniform interpolation and approximation in the description logic EL. In *KR*, 2012.

[Lyndon, 1959] Roger C. Lyndon. An interpolation theorem in the predicate calculus. *Pacific Journal of Mathemantics*, 9:129–142, 1959.

[McMillan, 2003] K. L. McMillan. Interpolation and sat-based model checking. In *CAV*, 2003.

[Meier, 2014] Michael Meier. The backchase revisited. *VLDB Journal*, 23(3):495–516, 2014.

[Nikitina and Rudolph, 2014] Nadeschda Nikitina and Sebastian Rudolph. (Non-)Succinctness of uniform interpolants of general terminologies in the description logic EL. *Artif. Intell.*, 215:120–140, 2014.

[Popa, 2000] Lucian Popa. *Object/Relational Query Optimization with Chase and Backchase*. PhD thesis, U. Penn., 2000.

[Schulz, 2013] Stephan Schulz. System Description: E 1.8. In *LPAR*, 2013.

[Toman and Weddell, 2011] D. Toman and G. Weddell. *Fundamentals of Physical Design and Query Compilation*. Morgan Claypool, 2011.

[Umans, 2001] Christopher Umans. The minimum equivalent DNF problem and shortest implicants. *J. Comput. Syst. Sci.*, 63(4):597–611, 2001.