Using HOL LIGHT to Reason over Higher-Order Meaning Representation Languages

Michael Minock

TCS/CSC
KTH Royal Institute of Technology
minock@kth.se

Abstract

This extended abstract presents some very preliminary work exploring higher-order *meaning representation languages* (MRLs) for natural language interfaces over databases. Specifically the HOL LIGHT theorem prover is being applied to test query containment for a language that uses sub-queries to compute counts, a key step toward covering MRLs as powerfull as SQL. While the deduction method is, by definition, sound, it can not be complete. Still, on a containment corpus derived from GEOQUERY, the prover is managing to deduce or refute many query containments automatically without interaction. Work is underway to supply additional theorems, that when added to the stock of available theorems, will automatically solve a broader and broader class of containment problems. It will be interesting to see how practical this approach can be made.

1. Introduction

In reviewing natural language interface efforts of the 1980s, (Copestake and Sparck Jones, 1990) observed that even simple domains require highly expressive MRLs. At roughly the same time the argument was put forth on the need to be able to decide logical equivalence between arbitrary MRL expressions (Shieber, 1993) [1]. This continues to presents something of a quandary.

Our position is to pursue expressive semantics rather than decidable inference, although we acknowledge the clear desirability of the later (Minock, 2014). As for how expressive, ultimately we desire an MRL roughly as expressive as SQL with its aggregation and grouping operators and its flexibility to use sub-query results in conditions. However since we aim to perform logical analysis (e.g. determine query containment, equivalence, etc.), we prefer the clean syntax and semantics of higher-order logic (Enderton, 2015) over the arguably grubby syntax and informal semantics of SQL.

The overarching question we are exploring is *'to what extent can higher-order reasoners be used practically in inference problems over relational databases?'* This is a broad and ambitions undertaking, thus, for now, we limit ourselves to a queries and sub-queries computing just counts. If we meet with success on this class of queries we will try to extend solutions to more complex aggregation functions (e.g. averages, sums, etc). Ultimately we aim to compute containment, equivalence disjointness problems for general SQL.

2. Queries with cardinalities over sets

The syntax and semantics we use is standard. The logic looks exactly like first order logic with the addition of 1-place predicate variables which range over sets of objects. We also use a special cardinality function $|X|$ indicating the cardinality of sets. Here we present[2] some example queries over the GEOQUERY corpus database (figure 1) paired with expressions in our higher-order query language:

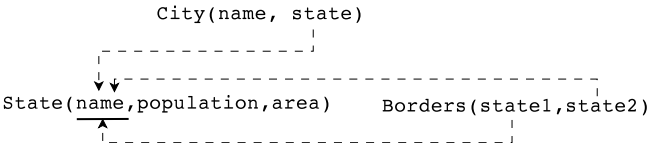

Figure 1: Part of GEOQUERY schema.

1. *"give me the cities in virginia"*:
   $\{(x)|City(x, \text{'Virginia'})\}$

   This query is a simple first-order expression which builds a set of 1-tuples for bindings of variable x.

2. *"how many cities are in montana"*:
   $\{(|X|)|(\forall x)(X(x) \leftrightarrow City(x, \text{'Montana'}))\}$

   This query returns the single tuple giving the size of the set X which is exactly the cities in Montana.

3. *"what states have more cities than ohio"*
   $\{(x)|State(x, \_, \_) \wedge (\exists Y)(\exists Z)$
   $((\forall y)(Y(y) \leftrightarrow City(y, x) \wedge$
   $(\forall z)(Z(z) \leftrightarrow City(z, \text{'Ohio'})) \wedge |Y| > |Z|)\}$

   This query, by introducing sets of cities in Ohio and sets of cities in the free variable of the query, can, via $>$ on the sizes of the sets, determine which states have more cities than Ohio.

4. *"What state has the most cites"*

[1] In short, this gives a principled way to resolve spurious ambiguity during analysis and also gives the capability of generating equivalent natural language paraphrases of equivalent logical expressions.

[2] To shorten the expressions we use *don't care* existential variables (_).

$\{(x)|State(x,\_,\_)\wedge$
$(\exists Y)(\forall y)(Y(y)\leftrightarrow City(y,x)\wedge$
$\neg(\exists w)(\exists Z)(\forall z)(Z(z)\leftrightarrow City(z,w)\wedge|Y|<|Z|))\}$

This query introduces a not exists over a set variable.

5. *"States where the majority of cities are less than 10,000 people."*

$\{(x)|State(x,\_,\_)\wedge$
$(\exists Y)(\exists Z)($
$(\forall y)(Y(y)\leftrightarrow(\exists p)(City(y,x,p)\wedge p<10000))\wedge$
$(\forall z)(Z(z)\leftrightarrow(\exists p)(City(z,x,p)\wedge p\geq 10000))\wedge$
$|Y|>|Z|)\}$

This query shows that generalized quantifiers(Barwise and Cooper, 1981) like majority are expressible within our MRL[3].

Given a database state D and a query Q of the above form, answers $Q(D)$ are computable; it is straight forward to map such queries to SQL with sub-queries computing counts. What is difficult is deciding things like query containment and thus by extension equivalence. Our focus is to automatically determine containment for as large a class of formulas as possible.

3. Using HOL LIGHT to decide query containment

The theorem prover we use is HOL LIGHT (Harrison, 2009), one of the descendants of the original HOL system of Gordon (Gordon, 1991). We picked HOL LIGHT because it seemed to be the easiest interactive higher-order logic theorem prover to install, comprehend and use.

Our method of testing if query Q_2 contains query Q_1 is to prepare the sentence $\Sigma\Rightarrow(Q_1\Rightarrow Q_2)$ where Σ expresses the relevant database constraints and the *unique names assumption* for the constants in Q_1 and Q_2. If HOL LIGHT can prove the validity of this sentence, then containment holds[4]. If HOL LIGHT does not return a theorem expressing the input sentence within a certain time span, then we conclude that the containment does not hold.

4. Initial Observations

To test our containment checker, we are constructing a corpus of containment problems over GEOQUERY augmented with the number of 2016 electoral college votes per state. This consists of two files. The first is an SQL schema and set of SQL INSERT statements to build a plain SQL database state. The second file, encoded in XML, represents a set of problems. Each problem consists of a representation of the assumptions, antecedent and consequent and a determination of whether the consequent relationship holds or does not hold. These 'representations' are encoded in natural language, higher-order logic and SQL. A part of this corpus is drawn from traces of our replication and extension of PRECISE (Popescu, et. al., 2003) as it attempted to simplify returned query sets. We are also adding additional problems that more extensively exercise the higher-order capabilities of our MRL. It is not difficult to come up with very challenging problems. Just a slight extension of our language to allow for constant multiplication in cardinality expressions would let us state things like, "there are two times as many rivers in Ohio than in Nebraska". Given such an extension we can easily pose problems on the cusp of being in or outside of Simpson's paradox – hard problems indeed. Our goal is to systematically add a range of problems from easy to practically impossible to our corpus.

As it stands now, our prover correctly solves all queries in our corpus that only require first order reasoning. We are still trying to find counter examples, but it seems that the model elimination prover in HOL LIGHT is very much up to the task. We add simple numerical constraints to the proofs via HOL LIGHT's `ARITH_RULE` function. Determinations of answer set size constraints in count queries (e.g. query 2 above) are also being correctly solved; using the theorem `CARD_SUBSET` we can determine that the number of cities in the Western States is always greater than the answer to the query 2 above.

Our current focus now is getting the reasoner to dig into set definitions. Currently we are manually constructing simple higher order proofs for individual examples to get a better insight into how to develop an automatic method.

5. Discussion

Recent work (Chatzikyriakidis and Luo, 2014; Mineshima, et. al., 2015) uses the Coq proof assistant in natural language inference – entailments, such as those given in the FraCaS corpus (Cooper, et. al., 1994), are computed between natural language statements. Our work is focused on determining formal query containment using a complex MRL. Ultimately we would like for our MRL to be as expressive as full SQL.

Because one may reduce arbitrary Boolean first-order logic (FOL) expressions to SQL, query containment and equivalence of SQL expressions must, in general, be undecidable. While large *query classes* (what we refer to as MRLs) are decidable for containment and thus equivalence[5], we see such MRLs as too limited for natural language interfaces. We thus pursue the sound, though incomplete approach proposed in this extended abstract and encode problems in HOL LIGHT. Since humans can reason over these types of problems, a person should be able to prove such lines of reasoning and encode them in HOL LIGHT theorems to further patch the system. Such theorems need to be defined over general predicates so that the patterns of reasoning developed in one domain are useful to other domains. It will be interesting to see how practical this approach can be made.

[3] Note that we extended the vocabulary in figure 1 to include city populations just to support this example.

[4] If the arities of the answer tuples of Q_1 and Q_2 are not equal, then we simply determine that query containment does not hold.

[5] For example MRLs limited to the select-project-join queries have long been known to be decidable for containment (Abiteboul, et. al., 1995). Many extensions preserved this, for example, unions of conjunctive queries, conjunctive queries with built-in predicates, conjunctive queries with inequalities, conjunctive queries under constraints, etc.

References

S. Abiteboul, R. Viannu, and V. Hull. *Foundations of Database Systems*. Addison Wesley, 1995.

J. Barwise and R. Cooper "Generalized Quantifiers and Natural Language". *Language and Philosophy*, Vol. 4. No.2,pp 159–219, 1981

S. Chatzikyriakidis and Z. Luo. Natural language inference in Coq. *Journal of Logic, Language and Information*, 23(4):441480. 2014.

R. Cooper, R. Crouch, J. van Eijck, C. Fox, J. van Genabith, J. Jaspers, H. Kamp, M. Pinkal, M. Poesio, S. Pulman. FraCaS: A Framework for Computational Semantics. Deliverable, D6. 1994

A. Copestake and K. Sparck Jones. Natural language interfaces to databases. *The Natural Language Review*, 5(4):225–249, 1990.

H. Enderton "Second-order and Higher-order Logic". *The Stanford Encyclopedia of Philosophy (Fall 2015 Edition)*, 2015

M. Gordon. Introduction to the HOL System. *Proceedings of the 1991 International Workshop on the HOL Theorem Proving System and its Applications*, Davis, California 1991

J. Harrison. HOL Light: An Overview. In *22nd International Conference on Theorem Proving in Higher Order Logics* , Munich 2009.

K. Mineshima, P. Martínez-Gómez, Y. Miyao and D. Bekki Higher-order logical inference with compositional semantics In *Empirical Methods in Natural Language Processing* Lisbon, 2015.

M. Minock. In pursuit of decidable 'logical form'. In *Swedish Language Technology Conference (SLTC)*, Uppsala, 2014.

A. Popescu, O. Etzioni, and H. Kautz. Towards a theory of natural language interfaces to databases. In *Intelligent User Interfaces*, 2003.

S. Shieber. The problem of logical-form equivalence. *Computational Linguistics*, 19(1):179–190, 1993.