Supplementary Material for the Paper: Complete Approximations of Incomplete Queries

Julien Corman, Werner Nutt, and Ognjen Savković

Free University of Bozen-Bolzano, Italy

Table of Contents

1 Overview

This document contains the proofs omitted from the submitted version due to space limitations. As in the paper, we assume that all queries are conjunctive queries.

2 The Complexity of Identifying MCGs

Recall that DP consists of problems that can be decided by an algorithm that simultaneously performs two calls to an NP-oracle. The algorithm responds "yes" if the first oracle call returns "yes" and the second one returns "no". Our proof is based on a reduction of a DP-complete graph problem, known as *Critical 3-colorability* [1]:

> Given a graph, is it true that the graph is not 3-colorable but any subgraph obtained by removing one of the edges is 3-colorable?

Proposition 1. *It is DP-complete to decide, given queries Q, Q', and a set of TC statements $\mathcal{C}$, whether $Q' \equiv G_{\mathcal{C}}(Q)$.*

Proof. (Membership) That the problem is in DP follows because one can verify that $Q' \equiv G_{\mathcal{C}}(Q)$ in two steps. For every atom in $A' \in D_{Q'}$ one has to find a statement $C \in \mathcal{C}$ such that $A' = T_C(D_Q)$. This can be done with one NP-oracle call. Similarly, one has to show for every $A \in D_Q \setminus D_{Q'}$ that such a rule does not exist. This can be done again with one call to an CO-NP oracle.

(Hardness) We show hardness by reducing the Critical 3-colorability problem. Assume we are given a graph $G = \langle V, E \rangle$ with vertices V and edges E. It is known that starting from G one can construct a Boolean query $Q_G \leftarrow B_G$ where:

$$B_G = \bigwedge_{(v_i, v_j) \in E} Eg(X_i, X_j)$$

such that G is 3-colorable iff Q_G evaluates to *true* over the database D that contains six correct colorings for the relation Eg: $\{Eg(red, blue), Eg(red, blue), \dots\}$. Similarly, for each edge $(v_i, v_j) \in E$ we construct a query $Q_G^{(i,j)} \leftarrow B_G^{(i,j)}$ where

$$B_G^{(i,j)} = \bigwedge_{(v_l,v_k)\in E\setminus(v_i,v_j)} Eg(X_l, X_k),$$

such that $Q_G^{(i,j)}(D) = \mathit{true}$ iff the subgraph $G^{(i,j)} = \langle V, E \setminus (v_i, v_j)\rangle$ is 3-colorable. We use this idea to construct a set of TCSs $\mathcal{C}$:

$$\begin{aligned}\mathcal{C} \quad = \quad & \Big(\bigcup_{(v_i,v_j)\in E} \mathit{Compl}(test_{(i,j)}; B_G^{(i,j)})\Big) \cup \mathit{Compl}(test_G; B_G) \\ & \cup \mathit{Compl}(Eg(X, Y); \mathit{true}),\end{aligned}$$

where $test_{(i,j)}$ are propositions. Then we construct two Boolean queries:

$$Q \leftarrow \Big(\bigwedge_{(v_i,v_j)\in E} test_{(i,j)}\Big) \wedge test \wedge D \quad \text{and} \quad Q' \leftarrow \Big(\bigwedge_{(v_i,v_j)\in E} test_{(i,j)}\Big) \wedge D.$$

From the characterizing condition for completeness "$\theta\bar{X} \in Q(T_{\mathcal{C}}(D_Q))$," we observe that each proposition $test_{(i,j)}$ is complete iff $G^{(i,j)}$ subgraph is 3-colorable, and similarly the proposition $test$ is complete iff G is 3-colorable. Next, query Q' is the same as Q except it does not contain the $test$ proposition. Thus, $Q' = G_{\mathcal{C}}(Q)$ iff each $G^{(i,j)}$ is 3-colorable but G is not. □

The fixpoint iteration algorithm computes a MCG (if it exists) in a number of steps linear in the size of a given query, using a DP-oracle. A call to a DP-oracle can be realized by two calls to an NP-oracle. Thus, the MCG decision problem belongs to the class of problems P^{NP}. This is a class of problems that can be decided in polynomial time by a Turing machine that uses an NP-oracle.[1]

Hence we have the following proposition.

Proposition 2. *Given queries Q and Q', and set of TC statements $\mathcal{C}$, the problem of checking whether Q' is the MCG of Q wrt $\mathcal{C}$ is in P^{NP}.*

It is unknown if the problem is also P^{NP}-hard.

3 The Number of MCSs for Acyclic Sets of TC Statements

Theorem 1. *Let Q be a query and $\mathcal{C}$ an acyclic set of TCSs such that exactly s relations names appear in $\mathcal{C}$. Then the number of atoms in a MCS of Q wrt $\mathcal{C}$ is in $\mathcal{O}(|Q| \times |C|^{s+1})$.*

Proof. We analyze the number of atoms that one needs to add to Q (in the worst case) in order to make it complete. We observe that some atoms in Q may need to be also instantiated but that does not affect the maximal number of atoms that one may need to

[1] An example of a complete problem for class P^{NP} provided by Krentel (1988) is the lexicographically last satisfying assignment of a Boolean formula: Given a Boolean formula $\phi(X_1, \dots, X_n)$ the question whether in the in the lexicographically largest satisfying assignment of ϕ variable X_n takes value 1.

consider. We also observe that adding more atoms may only create queries that are more special thus they wont be maximal any more.

Since the set of TCSs is acyclic we can create a total order on the relation names $R_1, \ldots, R_n$ in $\mathcal{C}$ such that if $i < j$ then there is no TC statement with R_j in the head and an R_i-atom in the condition. Further, for any relation that occurs in the condition of some TC statement with R_j as a head does not have and R_i-atom in the condition, and so on recursively. In other words, completeness of R_j-atoms does not depend on the completeness of R_i-atoms.

Now we can reason in the following way analyzing the atoms in Q. We start with some R_1-atom in Q (or the smallest i for which R_i-atom exists in Q), and we observe that to make each such atom complete (that is to match it with some TCS) we may need (in the worst case) to introduce an additional $|\mathcal{C}|$ atoms in Q. This is because each TC statement $\mathcal{C}$ is of size at most $|\mathcal{C}|$. These introduced atoms may be already complete or not, but again, in the worst case we may need for each of them to introduce again $|\mathcal{C}|$ new atoms from the conditions of TC statements. However, those new atoms are now R_i-atoms where $i \geq 2$. Hence, repeating the same logic, for R_2 we may need to introduce at most $|\mathcal{C}|$ many R_i-atoms, where $i \geq 3$, and so on. For R_n-atoms in Q we cannot introduce any new atom since their TCSs have an empty condition (otherwise there would be a loop in the dependency graph).

Hence, the total number of such steps in bounded by the number of relations in $|\mathcal{C}|$ which is s. Since we may need to perform this procedure for every atom in Q, in the worst case the resulting query can have $|Q| \times (|\mathcal{C}| + |\mathcal{C}|^2 + \cdots + |\mathcal{C}|^s)$ of total atoms, and so $|Q| \times (|\mathcal{C}| + |\mathcal{C}|^2 + \cdots + |\mathcal{C}|^s) \in \mathcal{O}(|Q| \times |\mathcal{C}|^{s+1})$.

□

4 Complete Unifiers Make Queries Complete

Applying a complete unifier to Q yields a complete query:

Proposition 3. *If γ is a complete unifier for a CQ Q and set $\mathcal{C}$ of TCSs, then $\mathcal{C} \models \mathit{Compl}(\gamma Q)$.*

Proof. First let us recall the completeness characterization for minimal queries. Let $\mathcal{C}$ be a set of TCSs and $Q(\bar{u}) \leftarrow B$ be a minimal query then it holds that $\mathcal{C} \models \mathit{Compl}(Q)$ iff $\theta B \subseteq T_{\mathcal{C}}(\theta B)$. In other words, $\mathcal{C} \models \mathit{Compl}(Q)$ iff for each $A \in B$ there exists a TCS $C = \mathit{Compl}(A'; G) \in \mathcal{C}$ such that

$$\theta A \in T_C(\theta B) = \{\beta A' \mid \beta \text{ is a substitution and } \beta G \subseteq \theta B\}. \tag{1}$$

Now we return to our queries Q and γQ from the theorem. Wlog we assume that γQ is minimal as well. From the definition of the unifier γ over γQ and $\mathcal{C}$, we have that for each atom A_i of the query there exists a TC statement $C_i = \mathit{Compl}(A_i'; G_i)$ and there are atoms $B_i \subseteq B$ such that $\gamma A_i = \gamma A_i'$ and $\gamma G_i = \gamma B_i'$. On the other hand, if in equation (1) we set $\beta = \theta\gamma$ we have that

$$\theta\gamma A \in T_C(\theta\gamma B) = \{\theta\gamma A' \mid \theta\gamma \text{ is a substitution and } \theta\gamma G \subseteq \theta\gamma B\}.$$

That is, equation (1) holds for γQ, so γQ is complete. □

5 Maximal Instantiations are Produced by Complete Unifiers

Theorem 2. *Let $\mathcal{C}$ be a set of TCSs, Q a query, and Q' a complete instantiation of Q wrt $\mathcal{C}$. Then there is a complete unifier γ for Q and $\mathcal{C}$ such that $Q' \sqsubseteq \gamma Q$.*

Proof. We restate again characterization (1) that we established in the preceding proof. Let $\mathcal{C}$ be a set of TCSs and $Q(\bar{u}) \leftarrow B$ be a minimal query then it holds that $\mathcal{C} \models \mathit{Compl}(Q)$ iff $\theta B \subseteq T_{\mathcal{C}}(\theta B)$. For a minimal query Q, it holds that $\mathcal{C} \models \mathit{Compl}(Q)$ iff for each $A \in B$ there exists a TC statement $C = \mathit{Compl}(A'; G) \in \mathcal{C}$ such that

$$\theta A \in T_C(\theta B) = \{\beta A' \mid \beta \text{ is a substitution and } \beta G \subseteq \theta B\}. \tag{2}$$

Let Q'' be a complete specialization of $Q(\overline{X}) \leftarrow A_1, \ldots, A_n$ and let α' be a specializing substitution such that $\alpha' Q \equiv Q''$. In the following we construct a complete specialization of Q that is at least as general as Q and that is obtained by applying some unifier γ on Q.

Wlog we assume that $\alpha' Q$ is a minimal query. According to characterization criteria for each atom $\alpha' A_i$ in $\alpha' Q$ there exists a TC statement $C_i = \mathit{Compl}(A_i'; G_i) \in \mathcal{C}$ such that

$$\theta \alpha' A_i \in T_{C_i}(\theta \alpha' B) = \{\beta A_i' \mid \beta \text{ is a substitution and } \beta G_i \subseteq \theta \alpha' B\}.$$

From there it follows that A_i and A_i' are unifiable and that G_i unifies with some set of atoms from B, say $B_i \subseteq B$. In fact, assuming that TCSs and query have different variables there exist substitutions β_i such that $\theta \alpha' A_i = \beta_i A_i'$ and $\beta_i G_i = \theta \alpha' B_i$, and thus $\theta \alpha' \beta_1 \cdots \beta_n$ is a unifier for all pairs.

We set that $\gamma' = \theta \alpha' \beta_1 \cdots \beta_n$ Since, pairs of are unifiable, there exists a mgu γ that unifies all pairs (and that is more general than γ', that is, $\gamma' \preceq \gamma$). Since $\gamma' \preceq \gamma$ then $Q' = \gamma' Q \sqsubseteq \gamma Q$. It is also not hard to see that γQ is complete. Since we have that $\gamma A_i = \gamma A_i'$ and $\gamma G_i = \gamma B_i'$, if in equation (2) we set $\beta = \theta\gamma$ we have that

$$\theta\gamma A_i \in T_{C_i}(\theta\gamma B)\{\theta\gamma A_i' \mid \theta\gamma \text{ is a substitution and } \theta\gamma G_i \subseteq \theta\gamma B\}.$$

That is (2) holds for γQ, that is, γQ is complete.

□

6 The k-MCS Algorithm is Correct

First we recall the algorithm (in this document listed as Algorithm 1) that returns all k-MCSs for a given query Q and set of TCSs $\mathcal{C}$.

Theorem 3 (k-MCS Algorithm). *Let Q be a CQ, $\mathcal{C}$ a set of TCSs, k a nonnegative integer and k-MCS$(Q, \mathcal{C})$ be the set of queries returned by Algorithm 1. Then for every query Q':*

$$Q' \in \text{k-MCS}(Q, \mathcal{C}) \quad \textit{iff} \quad Q' \textit{ is a k-MCS of } Q \textit{ wrt } \mathcal{C}.$$

Proof. Let b be the size of Q, i.e. the number of atoms in Q.

(Soundness). We show that the algorithm only returns complete specializations of Q of size $\leq n + k$.
An extension of Q is a specialization of Q. So each constructed query Q'' (Line 3) is a specialization of Q. Next, the call $\texttt{MCI}_{len+k}(Q'', \mathcal{C})$ returns complete specializations of

Algorithm 1: Computes k-MCSs

Input : a query $Q(\bar{u}) \leftarrow B$, a set $\mathcal{C}$ of TCSs , $k \in \mathbb{N}_0$
Output : the set $\mathcal{S}$ of all k-MCSs of Q wrt $\mathcal{C}$

1 $\mathcal{S} := \emptyset$
2 **foreach** *set* B' *of fresh atoms of size* $n + k - 1$ *over* $\Sigma_{\mathcal{C}}$ **do**
3 | *construct* $Q''(\bar{u}) \leftarrow B, B'$
4 | $\mathcal{S} := \mathcal{S} \cup \mathtt{MCI}_{\leq n+k}(Q'', \mathcal{C})$
5 **foreach** Q'' *in* $\mathcal{S}$ **do**
6 | **if** *exists* $Q''' \in \mathcal{S}$ *such that* $Q'' \sqsubseteq Q'''$ **then** $\mathcal{S} := \mathcal{S} \setminus \{Q''\}$
7 **return** $\mathcal{S}$

Q'' (therefore also of Q) of size $\leq n + k$. Thus $\mathcal{S}$ can only contain such queries, and since the algorithm returns a subset of $\mathcal{S}$, it can only return such queries.

(Completeness). Let us assume that our input query Q is of the form $Q(\bar{u}) \leftarrow B$, where B is a set of n atoms. And let $Q'(\bar{u}') \leftarrow B'$ be a complete specialization of Q of size $\leq n + k$. We show that our algorithm returns a complete specialization of Q of size $\leq n + k$ that is more general than Q'.

To do so, we construct a query Q'' that is one of the candidate queries generated by our algorithm (Line 2) for Q as input, in such a way that Q'' is more general than Q' (i.e. $Q' \sqsubseteq Q''$). In other words, Q' is a complete instantiation of Q''. Therefore, after the algorithm (Line 4) calls $\mathtt{MCI}_{\leq n+k}(Q'', \mathcal{C})$, the set $\mathcal{S}$ will contain either Q' itself or a query that is more general than Q'. Together with the fact that the algorithm returns the maximal elements of $\mathcal{S}$ (Lines 5 to 7), this proves our claim.

The query Q'' is of the form $Q''(\bar{u}) \leftarrow B, B''', B''''$, where the sets of atoms B''' and B'''' are defined as follows. Since Q' is a specialization of Q, there exists a homomorphism $\delta \colon Q \to Q'$ such that $\delta B \subseteq B'$. Let $B'' = B' \setminus \delta B$. We observe that $|B''| < n + k - 1$, since δ has to map B to at least one atom in B'. We define B''' as a fresh version of B'', that is, a set of atoms identical to B'', but where each variable is replaced with a fresh one (note that B''' also has size $< n + k - 1$). To define B'''', we select an arbitrary atom A in B, and construct B'''' with enough repeated fresh version of A so that $|B| + |B'''| + |B''''| = n + n + k - 1$.

To conclude the proof, we show that $Q' \sqsubseteq Q''$. First, we observe that B'''' is "redundant" in Q'', meaning that dropping B'''' yields an equivalent query. So in order to show that $Q' \sqsubseteq Q''$, it is sufficient to show that there is a homomorphism from (B, B''') to B'. Such a homomorphism can be constructed by extending δ (which, as a reminder, maps B to B') with a substitution δ' that maps B''' to B' and is compatible with δ. Trivially, from the construction of B''', such a substitution exists: since B''' is constructed out of $B'' \subseteq B'$ by renaming variables with fresh ones, δ' can be defined as the inverse of this renaming.

□

References

1. Christos M. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.