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 from that were missing in our submitted version.

2 Proof for Complexity of finding 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". We remind the reader that a graph problem called *critical 3-colorability* is a DP-complete problem: *For a given graph is it true that the graph is not 3-colorable but any subgraph obtained by removing one of the edges is 3-colorable [1].*

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) 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 again with one call to an CO-NP oracle.

(Hardness) We show hardness by reducing 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 database D that contains six correct coloring for *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 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 proposition $test$ is complete iff G is 3-colorable. Next, query Q' is the same as Q except it does not contain $test$ 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, 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. *For a 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 Finite Number of MCSs for Acycling TC statements

Theorem 1. *Let Q be a CQ 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 need 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 effect 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 TCSs are 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 the condition of some TC statement with R_j as a head does not have R_i-atom as 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 additional $|\mathcal{C}|$ in Q. This is because each TC statement $\mathcal{C}$ is of size at most $|\mathcal{C}|$. Then, such 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 $|\mathcal{C}|$ R_i-atoms where $i \geq 3$. And so on. For R_n-atoms in Q we cannot introduce any new atom since they their TC have empty condition (otherwise there will 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 do 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 Proof of the Soundness of Complete Unifier

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: $\mathcal{C} \models \mathit{Compl}(Q)$ 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 TC $C = \mathit{Compl}(A'; G) \in \mathcal{C}$ 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.

From the construction of γQ 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 atoms $B_i \subseteq B$ such that $\gamma A_i = \gamma A'_i$ and $\gamma G_i = \gamma B'_i$. If in the equation (2) we set $\beta = \theta\gamma$ we have that (2) holds for γQ, i.e., that is γQ is complete. □

5 Proof of the Sufficiency of Complete Unifier

Theorem 2. *Let $\mathcal{C}$ be a set of TCSs, Q a CQ, 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 again restate the characterization 2 listed the proof above. For a minimal query Q, it holds that $\mathcal{C} \models \mathit{Compl}(Q)$ iff for each $A \in B$ there exists a TC $C = \mathit{Compl}(A'; G) \in \mathcal{C}$ 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 that is as good as Q'' (or better) and that is returned by the algorithm. We do so by constructing a substitution α such that *(i)* αQ is returned by the algorithm, *(ii)* $\alpha' Q \sqsubseteq \alpha Q \sqsubseteq Q$ and *(iii)* αQ is complete.

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', i.e., $\gamma' \preceq \gamma$). Then either *(i)* γQ is returned by the algorithm, or the algorithm computes even more general specialization $\alpha'' Q$ for some γ'' for which it holds $\gamma Q \sqsubseteq \gamma'' Q$. Since $\gamma' \preceq \gamma$ then *(ii)* $\gamma Q \sqsubseteq \gamma' Q$. It is also not hard to see that *(iii)* γQ is complete. For each A_i we have that $\gamma A_i = \gamma A_i'$ and $\gamma G_i = \gamma B_i'$. If in the equation (2) we set $\beta = \theta\gamma$ we have that (2) holds for γQ, i.e., γQ is complete.

□

6 Proof of the k-MCS Algorithm

First we recall the Algorithm 1 that returns all k-MCSs for a given query Q and a set of TCSs $\mathcal{C}$. Then we can prove.

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

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

Proof. (Soundness) Each extension Q' of Q is a specialization of Q of size $|Q| + k$. Next, each call $\mathit{MCI}(Q', \mathcal{C})$ returns complete specializations Q'' of size $\leq |Q| + k$. Thus, Q'' is a complete specialization of Q.

(Completeness) Assume $Q'(\bar{u}') \leftarrow B'$ is a complete specialization of $Q(\bar{u}) \leftarrow B$ of size l, $n \leq l \leq n + k$. In the following we show there is a query Q'' that is constructed

Algorithm 1: Computes k-MCSs

Input : a query $Q(\overline{X}) \leftarrow B$, 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 | Let $Q''(\overline{X}) \leftarrow B, B'$
4 | $\mathcal{S} := \mathcal{S} \cup MCI_{\leq n+k}(Q'', \mathcal{C})$
5 **foreach** Q'' *in* $\mathcal{S}$ **do**
6 | **if** *exists* $Q''' \in \mathcal{S}$ *such that* $Q'' \sqsubseteq Q'''$ **then** $\mathcal{S} \leftarrow \mathcal{S} \setminus \{Q''\}$
7 **return** $\mathcal{S}$

by our algorithm in line 3, such that there must be a complete specialization Q''' of Q'' created in line 4, such that Q''' is is at least as general as Q' Such Q''' are returned by the subroutine $\texttt{MCI}_{\leq n+k}(Q'', \mathcal{C})$ that picks maximal CI of size $\leq n + k$ (this extends our MCI algorithm by first eliminating those which size is $> n + k$ and then among them eliminating non-maximal ones).

Since, we picked Q' as an arbitrary complete specialization then we are guaranteed that our algorithm considers all maximal complete specializations. Finally, our algorithm eliminated non-maximal specializations in line 5-6, thus it indeed returns all k-MCSs

Now we prove the main claim stated above: *that for any given Q' the subroutine $MCI_{\leq n+k}$ returns a query that is a CS of Q and that is at least general as Q'.*

Since Q' is a specialization of Q then there exists a homomorphism $\delta \colon Q \rightarrow Q'$ such that $\delta B \subseteq B'$. Let $B'' = B' \setminus \delta B$. We observe that $|B''| < n + k - 1$ since δ has to mapped B to at least one atom in B'. Now let B''' be a fresh version of B'', i.e., each atom in B' contains only fresh variables. Also $|B'''| < n + k - 1$. Our algorithm directly guesses fresh extension of size k hence to match our algorithm we do the following. Let A be an atom in B then we construct B'''' by repeating fresh version of A such that $|B| + |B'''| + |B''''| = n + n + k - 1$. We construct a query $Q''(\bar{u}) \leftarrow B, B''', B''''$. We observe that B''' is "redundant" in Q'' (i.e.,) dropping B''' one gets an equivalent query but our algorithm creates exactly such queries in line 2.

Now we observe that $Q' \sqsubseteq Q''$, because starting from δ one can construct a homomorphism δ' from Q'' to Q' such that δ' and δ match on the variables from Q and then extend it cover the variables in B'' and B''' so the image of B'' is exactly B' and of B''' is A.

In other words, Q' is a complete instantiation of Q''. Hence, when we call $\texttt{MCI}_{\leq n+k}(Q'', \mathcal{C})$ it will return all MCIs of Q'', and at least one of those is either Q' itself or a query is more general than Q'. □

References

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