When is Shapley Value Computation a Matter of Counting?

Meghyn Bienvenu
meghyn.bienvenu@cnrs.fr
Univ. Bordeaux, CNRS,
Bordeaux INP, LaBRI, UMR 5800
F-33400, Talence, France

Diego Figueira
diego.figueira@cnrs.fr
Univ. Bordeaux, CNRS,
Bordeaux INP, LaBRI, UMR 5800
F-33400, Talence, France

Pierre Lafourcade
pierre.lafourcade@u-bordeaux.fr
Univ. Bordeaux, CNRS,
Bordeaux INP, LaBRI, UMR 5800
F-33400, Talence, France

ABSTRACT

The Shapley value provides a natural means of quantifying the contributions of facts to database query answers. In this work, we seek to broaden our understanding of Shapley value computation (SVC) in the database setting by revealing how it relates to Fixed-size Generalized Model Counting (FGMC), which is the problem of computing the number of sub-databases of a given size and containing a given set of assumed facts that satisfy a fixed query. Our focus will be on explaining the difficulty of SVC via FGMC, and to this end, we identify general conditions on queries which enable reductions from FGMC to SVC. As a byproduct, we not only obtain alternative explanations for existing hardness results for SVC, but also new complexity results. In particular, we establish FP-#P complexity dichotomies for constant-free unions of connected CQs and connected homomorphism-closed graph queries. We also consider some variants of the SVC problem, by disallowing assumed facts or quantifying the contributions of constants rather than facts.

☞ This pdf contains internal links: clicking on a notion leads to its *definition*.

1 INTRODUCTION

The Shapley value [13] is a well-known measure for distributing wealth among players in cooperative games. It enjoys many desirable properties and has found application in numerous areas, including databases, where it provides a natural means of quantifying the contribution of a database element (typically a tuple) to a query result. The problem of Shapley value computation (SVC) for database queries has received considerable attention lately [5, 7, 9, 11]. It is known to be computationally challenging – #P-hard and $\mathrm{FP}^{\#\mathrm{P}}$-complete under polynomial-time reductions – which motivates the question of identifying those queries for which SVC is tractable (*i.e.*, in FP). It has been conjectured [5, 7] that a FP/#P-hard dichotomy holds for unions of conjunctive queries, as is the case for query evaluation in tuple-independent probabilistic databases [4] and for the generalized model counting problem [8]. At present, this conjecture has been only confirmed for the subclass of self-join-free conjunctive queries (sjf-CQs) [11]. There has however been progress on classifying the complexity of SVC for other query classes, with FP/#P-hard dichotomies proven for sjf-CQs with safe negations [12] and regular path queries [9].

The dichotomy established in [11] for SVC for sjf-CQs identifies precisely the same tractable queries (the hierarchical ones) as the dichotomies for probabilistic query evaluation (PQE) and generalized model counting (GMC), despite having been obtained through different means. This raised the intriguing question of whether the Shapley dichotomy was a consequence of these existing dichotomies, and more generally, what is the precise relationship holding between these problems. Subsequent work [5] provided a partial answer by exhibiting a general polynomial-time reduction (for all queries, not just sjf-CQs) of SVC to PQE, thereby showing how the tractability results for PQE can be transferred to SVC.

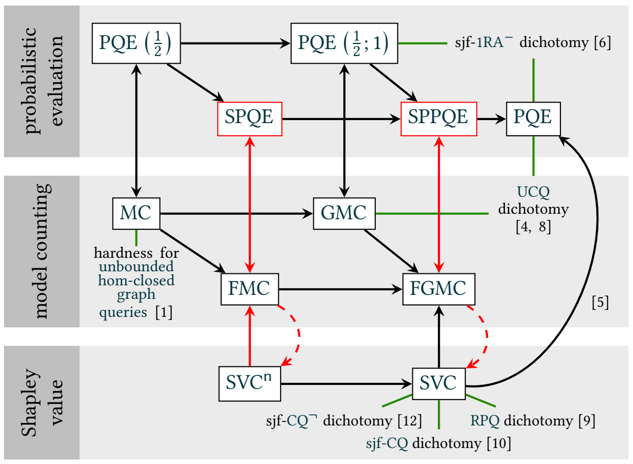
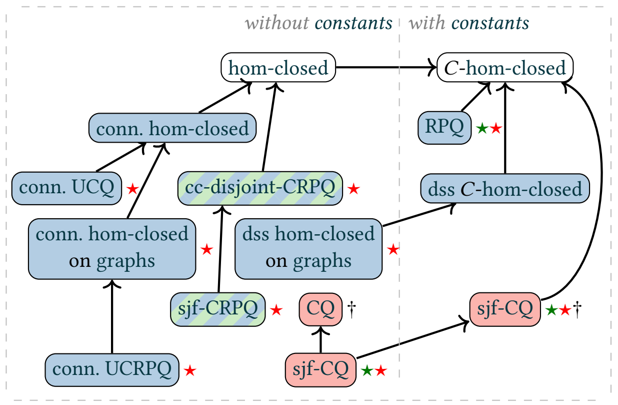


Figure 1: A (clickable) summary of the reductions. An arrow from A to B means a polynomial-time Turing-reduction from A to B. In red our contribution.

Can hardness results for SVC similarly be explained via PQE or GMC? A recent result [7] further demonstrated the closeness of the settings by reproving the hardness of SVC using the same model counting problem for Boolean functions as was originally used to show hardness of PQE for non-hierarchical sjf-CQs [3]. However, a reduction from PQE or GMC to SVC has yet to be exhibited, no doubt due to the fact that SVC outputs a complex weighted sum which cannot be obviously employed for probability or model counting computations. Our paper investigates this missing link and shows that the matching dichotomy conditions are no coincidence, as it is possible to exhibit reductions from PQE and GMC to SVC for broad classes of queries, which not only explain existing hardness results for SVC, but also yield some new complexity results.

Contributions

Our first conceptual contribution (Section 3) is to introduce suitable specializations of the PQE problem, namely SPQE and SPPQE, obtained by restricting the probability values of facts, and showing that they are polynomial-time equivalent to variants FMC and FGMC of the model counting MC and generalized model counting GMC problems (*cf.* “model counting” and “probabilistic evaluation” boxes of Figure 1). While some of these problems had been implicitly used in proofs, or defined for related settings, their formal



🟦 Lemma 4.1 ★/★ = dichotomy exists (prior work/this paper)
🟩 Lemma 4.2 dss = duplicable singleton support
🟥 Lemma 4.3 †= not preserving FGMC-equiv. with the same query

Figure 2: A (clickable) summary of the reductions from FGMC for the classes of queries captured by our results, and the FP/#P-hard dichotomies entailed.

introduction in the database setting serves to bring some clarity and order to the literature and provides us with suitable analogues of SVC in the probabilistic and counting domains. Indeed, the existing reduction of SVC to PQE can more precisely be expressed as a reduction to the more restricted FGMC and SPPQE problems.

Our main technical contributions (presented in Section 4 and proven in Section 5) are the first known reductions in the opposite direction: from FGMC (equivalently, SPPQE) to SVC. We in fact provide three such reductions (Lemmas 4.1 to 4.3), each employing somewhat different structural conditions and covering different classes of (C-)homomorphism-closed queries. Notably, in some cases, we can establish the polynomial-time equivalence of the SVC problem for query q and the FGMC problem for the same query.

As a consequence of our reductions, it is now possible to obtain the FP/#P-hard dichotomies for SVC for sjf-CQs and regular path queries as corollaries of existing dichotomies for PQE and GMC (intuitively, by following the arrows in Figure 1 from and to known results). Moreover, due to the general nature of our lemma hypotheses, additional new results can be obtained. In particular, we establish FP/#P dichotomies for constant-free unions of connected CQs and connected homomorphism-closed graph queries. These results are presented in Section 4 and summarized in Figure 2.

In Section 6, we further demonstrate the utility of our proof techniques by showing how they can be applied to other scenarios. First, we show that essentially the same reductions, modulo an additional hypothesis, can be used to show hardness of Shapley value computation over databases without assumed ("exogenous") facts (existing reductions make the crucial use of the presence of "exogenous" facts). Second, a simple adaptation of our techniques allows us to partially recapture existing results for CQs with safe negation [12]. Thirdly, we show that computing the maximum Shapley value is just as hard as SVC as far as our reductions go. Finally, we initiate the study of Shapley value of database constants (rather than facts) and employ our methods to show how this problem can be formally related to the analogous model counting problems.

2 PRELIMINARIES

We fix disjoint infinite sets Const, Var of *constants* and *variables*, respectively. For any syntactic object O (*e.g.* database, query), we will use $\mathit{vars}(O)$ and $\mathit{const}(O)$ to denote the sets of variables and constants contained in O, and let $\mathit{term}(O) \doteq \mathit{vars}(O) \cup \mathit{const}(O)$ denote its set of *terms*. A *(relational) schema* is a finite set of relation symbols, each associated with a (positive) arity. A *(relational) atom* over a schema Σ takes the form $R(\bar{t})$ where R is a *relation name* from Σ of some arity k, and $\bar{t} \in (\mathsf{Const} \cup \mathsf{Var})^k$. A *fact* is an atom which contains only constants. A *database* $\mathcal{D}$ over a schema Σ is a finite set of facts over Σ, and we call it a *graph database* if Σ is a binary schema (*i.e.*, consisting of binary relations only).

For sets of atoms S_1, S_2, a *homomorphism* from S_1 to S_2 is a function $h : \mathit{term}(S_1) \to \mathit{term}(S_2)$ such that $R(h(t_1), \dots, h(t_k)) \in S_2$ for every $R(t_1, \dots, t_k) \in S_1$. We write $S_1 \xrightarrow{hom} S_2$ to indicate the existence of such h. If further we have that $h(c) = c$ for every $c \in C \subseteq \mathsf{Const}$, we call h a *C-homomorphism* and write $S_1 \xrightarrow{C\text{-}hom} S_2$.

We say that a set of atoms S is *connected* if so is its associated undirected *incidence graph* $\mathbf{G}_S$, whose nodes are $S \cup \mathit{term}(S)$ and whose edges are $\{ \{t, \alpha\} \mid t \in \mathit{term}(\alpha), \alpha \in S\}$.

Query languages. To simplify the presentation, we shall only consider Boolean queries, which we will often refer to simply as "queries". A *Boolean query* q specifies a true-or-false property of databases. We write $\mathcal{D} \models q$ to denote that database $\mathcal{D}$ satisfies query q, in which case we may also call $\mathcal{D}$ a *support* for q.

A query q is *closed under homomorphisms*, or *hom-closed*, if for every pair $\mathcal{D}, \mathcal{D}'$ of databases such that $\mathcal{D} \xrightarrow{hom} \mathcal{D}'$ and $\mathcal{D} \models q$, we have $\mathcal{D}' \models q$ as well. We define similarly *closure under C-homomorphisms*, or *C-hom-closed*, to capture queries with constants. For (C-)hom-closed queries, it is natural to consider the smallest databases that satisfy the query. Formally, we call $\mathcal{D}$ a *minimal support* for q if $\mathcal{D} \models q$ and $\mathcal{D}' \not\models q$ for every $\mathcal{D}' \subsetneq \mathcal{D}$. We say a fact α is *relevant to* a query q if it appears in some minimal support of q. A query q is *connected* if every minimal support for q is connected.

We will consider queries formulated in (fragments of) standard query languages. A (Boolean) *conjunctive query* (henceforth just *CQ*) over Σ is a conjunction of atoms over Σ, all of whose variables are existentially quantified. We use $\mathit{atoms}(q)$ to denote the set of atoms of a CQ q. A database $\mathcal{D}$ *satisfies* a CQ q if $\mathit{atoms}(q) \xrightarrow{C\text{-}hom} \mathcal{D}$ where $C = \mathit{const}(q)$. We call a CQ q *self-join-free* if no two atoms of q have the same relation name, and we use *sjf-CQ* for the class of all self-join-free CQs. A *union of conjunctive queries* (*UCQ*) is a finite disjunction of CQs, with $\mathcal{D} \models q_1 \vee \dots \vee q_\ell$ iff $\mathcal{D} \models q_i$ for some $1 \leqslant i \leqslant \ell$. *Infinitary unions of CQs* (UCQ^∞) are defined similarly but the disjunction may be over a countably infinite set of CQs.

A *graph query* is any query over a binary schema. A *path atom* over a binary schema Σ takes the form $L(t, t')$, with $t, t' \in \mathsf{Const} \cup \mathsf{Var}$ and L a regular expression over the alphabet Σ. A (Boolean) *regular path query*, or *RPQ*, over Σ is a path atom $L(a, b)$ over Σ, with $a, b \in \mathsf{Const}$. Satisfaction of RPQs is as follows: $\mathcal{D} \models L(a, b)$ iff there exists a word $w = R_1 \dots R_\ell$ in the language of L and a sequence $c_0, c_1, \dots, c_{\ell-1}, c_\ell$ of constants such that $c_0 = a$, $c_\ell = b$, and $R_i(c_{i-1}, c_i) \in \mathcal{D}$ for every $1 \leqslant i \leqslant \ell$. A (Boolean) *conjunctive regular path query* (*CRPQ*) over Σ is an existentially quantified conjunction of path atoms over Σ. A graph database $\mathcal{D}$ satisfies a CRPQ q if there exists a mapping $h : \mathit{term}(q) \to \mathit{const}(\mathcal{D})$ such that $h(c) = c$ for every $c \in \mathit{const}(q)$ and $\mathcal{D} \models L(h(t), h(t'))$ for every path atom $L(t, t')$ of q. We shall also consider *unions of conjunctive regular path queries* (*UCRPQs*), defined as expected.

Reductions. We will use the standard notion of polynomial-time Turing-reductions between numeric problems Ψ_1, Ψ_2. Concretely, we write $\Psi_1 \leq_{poly} \Psi_2$, and say that there is a *polynomial-time reduction* from Ψ_1 to Ψ_2, if there exists a polynomial-time algorithm for computing Ψ_1 using unit-cost calls to Ψ_2. If $\Psi_1 \leq_{poly} \Psi_2$ and $\Psi_2 \leq_{poly} \Psi_1$, we write $\Psi_1 \equiv_{poly} \Psi_2$.

3 STUDIED PROBLEMS

In this section, we formally introduce Shapley value computation, which is the focus of the paper, as well as related model counting and probabilistic querying problems.

For the definition of Shapley values, as well as the task of generalized model counting, we shall require that the input database $\mathcal{D}$ has been partitioned into two sets $\mathcal{D}_{\mathsf{n}}$ and $\mathcal{D}_{\mathsf{x}}$ of *endogenous* and *exogenous* facts respectively. The notation $\mathcal{D} = (\mathcal{D}_{\mathsf{n}}, \mathcal{D}_{\mathsf{x}})$ will be used to refer to such *partitioned databases*. All databases will henceforth be assumed to be partitioned unless otherwise noted.

3.1 Shapley Value Computation

The Shapley value has been introduced by Lloyd Shapley in 1952 [13] to distribute the wealth of a cooperative game, and it follows three axioms of good behaviour. Intuitively, these axioms state that the value of two isomorphic games is the same, that the sum of all players values equals the total wealth of the game and that the value of a sum of two games equals the sum of the individual games' values. Remarkably, there is only one function satisfying these axioms: the Shapley value.

A *cooperative game* is given by a finite set of players P and a wealth function $\mathbf{v} : \wp(P) \to \mathbb{Q}$ that represents the wealth generated by every possible coalition of players, such that $\mathbf{v}(\varnothing) = 0$. To define the Shapley value, assume players arrive one by one in a random order, and each earns the difference between the coalition's wealth before she arrives and after. The *Shapley value* of a player $p \in P$ is her expected earnings in this scenario, which can be expressed as:

$$\mathrm{Sh}(P, \mathbf{v}, p) \doteq \frac{1}{|P|!} \sum_{\sigma \in \mathbb{P}(P)} \left(\mathbf{v}(\sigma_{<p} \cup \{p\}) - \mathbf{v}(\sigma_{<p})\right) \qquad (1)$$

where $\mathbb{P}(P)$ denotes the set of permutations of P and $\sigma_{<p}$ the set of players that appear strictly before p in the permutation σ. The following equivalent formula – more convenient for our proofs – can be obtained by grouping together the σ having the same $\sigma_{<p}$.

$$\mathrm{Sh}(P, \mathbf{v}, p) = \sum_{B \subseteq P \setminus \{p\}} \frac{|B|!(|P| - |B| - 1)!}{|P|!} \left(\mathbf{v}(B \cup \{p\}) - \mathbf{v}(B)\right) \quad (2)$$

The cooperative games that will interest us are those associated with a Boolean query q and a partitioned database $\mathcal{D} = (\mathcal{D}_{\mathsf{n}}, \mathcal{D}_{\mathsf{x}})$, where the set of players is the set $\mathcal{D}_{\mathsf{n}}$ of endogenous facts, and the *wealth function* $\mathbf{v}_q$ assigns $v_S - v_{\mathsf{x}}$ to each subset $S \subseteq \mathcal{D}_{\mathsf{n}}$, where $v_S = 1$ (resp. $v_{\mathsf{x}} = 1$) if $S \cup \mathcal{D}_{\mathsf{x}} \models q$ (resp. if $\mathcal{D}_{\mathsf{x}} \models q$), and 0 otherwise.

The main focus of this paper is to understand the complexity of computing the Shapley value of facts in such games. For a fixed Boolean query q, this is the task of computing, for a given input database $\mathcal{D}$ and fact $\alpha \in \mathcal{D}_n$, the value $\mathrm{Sh}(\mathcal{D}_{\mathsf{n}}, \mathbf{v}_q, \alpha)$. We will use SVC_q to refer to this computational task, and $\mathrm{SVC}_q^{\mathsf{n}}$ to the task when restricted to partitioned databases with only endogenous facts (*i.e.*, of the form $\mathcal{D} = (\mathcal{D}_{\mathsf{n}}, \mathcal{D}_{\mathsf{x}})$ with $\mathcal{D}_{\mathsf{x}} = \varnothing$).

Remark 3.1 (Non-Boolean case). *Shapley value computation could be defined for non-Boolean queries by asking for the contribution of a fact to a given tuple being a query answer. This variant straightforwardly reduces to our Boolean version, by substituting the query's free variables with constants from the answer tuple. Observe however that this reduction yields Boolean queries* with constants, *thus motivating the interest of obtaining results for queries with constants.*

3.2 Model Counting

We now introduce several versions of the model counting problem. For any Boolean query q, the *generalized model counting problem* on q, or GMC_q problem, is the task of computing, for a given input database $\mathcal{D} = (\mathcal{D}_{\mathsf{n}}, \mathcal{D}_{\mathsf{x}})$, the number of subsets $S \subseteq \mathcal{D}_{\mathsf{n}}$ such that $S \uplus \mathcal{D}_{\mathsf{x}} \models q$ (such subsets will be called *generalized supports for q in $\mathcal{D}$*). On the other hand, the *fixed-size GMC problem*, or FGMC_q, is the task of computing, for a given $n \in \mathbb{N}$ and database $\mathcal{D} = (\mathcal{D}_{\mathsf{n}}, \mathcal{D}_{\mathsf{x}})$, the number of subsets $S \subseteq \mathcal{D}_{\mathsf{n}}$ of size exactly n such that $S \uplus \mathcal{D}_{\mathsf{x}} \models q$. The *model counting problem* MC_q and *fixed-size model counting problem* FMC_q correspond to the previous problems when restricted so that the input database contains no exogenous facts (*i.e.*, $\mathcal{D}_{\mathsf{x}} = \varnothing$).

3.3 Probabilistic Query Evaluation

A *tuple-independent probabilistic database* is a pair $\mathcal{D} = (S, \pi)$ where S is a finite set of facts and $\pi : S \to (0, 1]$ is a probability assignment. We *associate* to every such database a partitioned database with $\mathcal{D}_{\mathsf{x}} \doteq \{\alpha \in S \mid \pi(\alpha) = 1\}$. For a Boolean query q, $\Pr(\mathcal{D} \models q)$ is the probability of q being true, where each fact α has independent probability $\pi(\alpha)$ of being in the database. For a fixed Boolean query q, the problem of computing, for a given a tuple-independent probabilistic database $\mathcal{D}$, the probability $\Pr(\mathcal{D} \models q)$ is known as the *probabilistic query evaluation* problem, or PQE_q. We consider also restrictions of PQE_q where the input probabilistic database (S, π) is such that the image $Im(\pi)$ has certain characteristics.

(1) If $Im(\pi) = \{\frac{1}{2}\}$ we obtain $\mathrm{PQE}_q\left(\frac{1}{2}\right)$.
(2) If $Im(\pi) \subseteq \{\frac{1}{2}, 1\}$ we obtain $\mathrm{PQE}_q\left(\frac{1}{2}; 1\right)$.
(3) If $Im(\pi) = \{p\}$ for some $p \in (0, 1]$, we obtain SPQE_q: the *single probability query evaluation* problem.
(4) If $Im(\pi) = \{p, 1\}$ for some $p \in (0, 1]$, we obtain SPPQE_q: the *single proper probability query evaluation* problem.

3.4 Prior Work and Relations Between Problems

PQE was first introduced in [3], and the most important result about it is the FP/#P-hard dichotomy for UCQs, established in [4]. The UCQs for which PQE is tractable are known as "*safe*". MC, GMC and their probabilistic counterparts have been considered in [8] where the same dichotomy has been extended to the latter:

Proposition 3.1 ([4, Theorem 4.21] and [8, Theorem 2.2]). *Let q be a Boolean UCQ. If q is safe, then both PQE_q and GMC_q are in FP, otherwise both are #P-hard.*

Another notable result is the hardness of MC for hom-closed graph queries which are *unbounded*, that is, queries which are not equivalent to a UCQ:

Proposition 3.2 ([1, Theorem 1.3]). *For every unbounded hom-closed graph query q, MC_q is #P-hard.*

This result immediately yields a FP/#P dichotomy for GMC over hom-closed graph queries. To the best of our knowledge, FGMC, FMC, SPPQE, and SPQE have not been explicitly introduced before in the context of databases. However, a notion of fixed-size model counting over Boolean functions has been used in [7], which is analogous to FGMC.[1]

SVC was first introduced in [10], where a FP/#P-hard dichotomy was established for sjf-CQs. Interestingly, the latter result identifies the same tractable queries as for PQE, namely, the safe sjf-CQs, which admit a simple syntactic characterization as the *hierarchical*[2] sjf-CQs. Recent work has clarified the relation between the two dichotomies by reducing SVC to PQE [5] and reproving the hardness of SVC [7] by reduction from the same model counting problem for Boolean functions that had been used to show hardness of PQE for non-hierarchical sjf-CQs [3].

The problems presented in this section are closely related. Figure 1 summarizes the reductions between these problems.

Proposition 3.3. *For any Boolean query q, all the polynomial-time reductions denoted by the solid arrows of Figure 1 hold. In particular:*

(1) $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SPPQE}_q$;
(2) $\mathrm{FMC}_q \equiv_{poly} \mathrm{SPQE}_q$;
(3) $\mathrm{SVC}_q \leq_{poly} \mathrm{FGMC}_q$.

Moreover, the reductions for (1) and (2) only use the oracle on inputs with the same associated partitioned database.

Proof sketch. The fact that $\mathrm{SVC}_q \leq_{poly} \mathrm{PQE}_q$ is known from [5, Proposition 3.1], and the proofs of items 1 to 3 are adaptations of this result (details in the appendix). The proof of $\mathrm{SVC}^{\mathrm{n}}_q \leq_{poly} \mathrm{FMC}_q$ is relegated to Section 6.1. The remaining reductions are trivial. ◻

4 MAIN RESULTS

Our main results are reductions from FGMC_q to SVC_q for large classes of C-hom-closed queries, which shall enable us to obtain known and new dichotomy results for SVC for several classes of queries. At a very high-level we show that:

(i) $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_q$ when q is (almost) connected (Lemma 4.1),
(ii) $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_{q\wedge q'}$ if q is connected and the queries q and q' do not have an undesirable interaction (Lemma 4.2),
(iii) $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_q$ whenever q can be decomposed into two queries $q = q_1 \wedge q_2$ with no undesirable interaction (Lemma 4.3).

However, the concrete notions of what we mean by "connected" and "undesirable interaction" are somewhat technical and differ between the items. The rationale for such definitions is to make the reduction statements as general as possible, which will become evident when we present the proofs in Section 5. The involved hypotheses are the price to pay for the ability to derive complexity results for several query languages at once.

In the remainder of the section, we make precise these hypotheses, provide formal statements of our key results, and present corollaries for concrete query classes (summarized in Figure 2).

4.1 Almost Connected Queries

Consider a finite set $C \subseteq \mathrm{Const}$ and a C-hom-closed query q. An *island support* for q is a support S for q such that for every set of facts S' with $const(S) \cap const(S') \subseteq C$, every minimal support for q in $S \cup S'$ is contained either in S or in S'. Intuitively, the facts of S cannot participate in minimal supports outside S, except perhaps for those over C. We say that q is *pseudo-connected* if it has a minimal support S that is an island support and such that $const(S) \nsubseteq C$.

Lemma 4.1. *For every pseudo-connected C-hom-closed query q, we have $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_q$.*

The most notable class of pseudo-connected queries is the one of connected hom-closed queries (*cf.* Lemma B.1). Observe that these are queries which are equivalent to infinitary unions of connected constant-free CQs.

Corollary 4.1. *For every connected hom-closed query q, we have $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SVC}_q$.*

As a consequence of Corollary 4.1, we obtain three FP/#P-complete dichotomies (also depicted in Figure 2) by leveraging the known UCQ dichotomy for GMC and PQE [4, 8], as well as the hardness for MC over hom-closed graph queries [1]:

Corollary 4.2.

(1) For any connected hom-closed UCQ q, SVC_q is either in FP or #P-hard.
(2) For any connected hom-closed graph query q, SVC_q is either in FP or #P-hard. —in particular, this holds for any connected UCRPQ without constants.

Another class of pseudo-connected queries is the class of RPQs (*cf.* Lemma B.2). We can then obtain the dichotomy for RPQs established in [9] now as corollary of Lemma 4.1.

Corollary 4.3 (Originally established in [9]). *Let q be an RPQ of language L. If L contains a word of length at least 3, then SVC_q is #P-hard, otherwise it is in FP.*

The pseudo-connected class contains also queries that are *not connected* in any real sense, for instance any C-hom-closed query with a "*duplicable singleton support*", that is, a support of size 1 that contains a constant outside of C. The name comes from the fact that we can make homomorphic copies by renaming a constant outside of C. The most basic example of such a query would be $A(x) \vee q$ for any C-hom-closed q. More interesting examples would include a CRPQ containing just one path atom $\exists x\, L(a, x)$, where a is a constant and L any language containing a word of length 1,

[1]The reason why it is analogous to FGMC rather than to FMC is because the exogenous facts get abstracted out in the transformation between databases and Boolean functions.
[2]We recall that a CQ q is *not hierarchical* iff there exist $\alpha_1, \alpha_2, \alpha_3 \in atoms(q)$ such that $(vars(\alpha_1) \cap vars(\alpha_2)) \nsubseteq vars(\alpha_3)$ and $(vars(\alpha_3) \cap vars(\alpha_2)) \nsubseteq vars(\alpha_1)$.

such as $L = A^*B$. Other examples may include disconnected atoms, such as $\exists x, y, u, z\ R(a, x, y) \wedge R(u, b, z)$ (with a, b constants).

Corollary 4.4. *For every C-hom-closed query q that admits a duplicable singleton support,* $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SVC}_q$.

As for the known dichotomy for SVC over sjf-CQs [10], it turns out that, although not all sjf-CQs are pseudo-connected, the intractable queries always have an intractable component which is "connected via variables". This motivates the following definition and variant of Lemma 4.1 above.

A Boolean CQ q is *variable-connected* if the incidence graph $\mathbf{G}_{atoms(q)}$ remains connected after removal of the nodes $const(q)$. We say that a C-hom-closed query is *variable-connected* if it is equivalent to an infinitary union of variable-connected C-hom-closed Boolean CQs. Observe that any hom-closed query is connected iff it is variable-connected.

A fact α is called a q-*leak* if there exists some fact α' from some minimal support of q and a C-homomorphism $h : \{\alpha'\} \to \{\alpha\}$ such that $h(c) \in C$ for some $c \in const(\alpha') \setminus C$. Intuitively, a q-leak is a structure that allows a minimal support of a variable-connected q to intersect two databases that share no constant outside of C, by instantiating a variable with a constant in C. For instance, consider the ($\{a\}$-hom-closed) CRPQ $q \mathrel{\hat{=}} \exists x\ [AB + BA](x, a) \equiv \exists x, y\ (A(x, y) \wedge B(y, a)) \vee (B(x, y) \wedge A(y, a))$. The fact $A(b, a)$ is a q-leak because of the first fact of the minimal support $\{A(b, d), B(d, a)\}$ and the mapping $\{b \mapsto b, d \mapsto a\}$.

Lemma 4.2. *Let q_1, q_2 be C_1-hom-closed and C_2-hom-closed queries, respectively, such that*

(1) q_1 is variable-connected;
(2) there is a minimal support S_2 of q_2 such that
    (a) $S_2 \not\models q_1$,
    (b) S_2 has no q_1-leak,
    (c) if a fact $\alpha \in S_2$ is relevant to q_1, then $const(\alpha) \nsubseteq C_1$;
(3) q_1 has a minimal support S_1 with no q_1-leak.

Then, $\mathrm{FGMC}_{q_1} \leq_{poly} \mathrm{SVC}_{q_1 \wedge q_2}$.

Corollary 4.5. *Let q be a non-hierarchical sjf-CQ or a non-hierarchical constant-free CQ. Then,* SVC_q *is #P-hard.*

The preceding lemma allows us, in particular, to recapture the dichotomy over sjf-CQs established in [10]. It does not prove a full dichotomy over constant-free CQs however because there could potentially be hierarchical queries which are #P-hard.

4.2 Disconnected Queries

Another approach for reducing FGMC to SVC is to have a query whose answers on any database can be computed as two queries over disjoint subsets of the data. We can in fact view the database as being probabilistic due to the FGMC $\equiv_{poly}$ SPPQE equivalence. In this setting, the intuition is that we compute the probability of each subquery by replacing the part of the query that does not affect it by some minimal support S of the other subquery, apply the same construction as before to compute the probability of this subquery, and finally multiply the two probabilities together.

Formally, we say that a C-hom-closed query q is *decomposable into* $q_1 \wedge q_2$ if it is equivalent to $q_1 \wedge q_2$, where q_1, q_2 are Boolean queries such that

(1) there are minimal supports S_1, S_2 of q_1, q_2 respectively, with $const(S_1) \nsubseteq C$ and $const(S_2) \nsubseteq C$,
(2) for all minimal supports S_1, S_2 of q_1, q_2 respectively, we have $S_1 \cap S_2 = \varnothing$.

Lemma 4.3. *For every decomposable C-hom-closed query q, we have* $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_q$.

The most obviously decomposable queries are those that can be written as the conjunction of two independent subqueries over distinct vocabularies. In fact, in the constant-free case, this syntactic class covers all decomposable queries.

Lemma 4.4. *A hom-closed Boolean query q is decomposable iff it is equivalent to $q_1 \wedge q_2$, where q_1 and q_2 are Boolean UCQ^∞ queries that do not share any relation name.*

This characterization does not apply to C-hom-closed queries, as witnessed by the query $\exists x, y\, R(a, x) \wedge R(b, y)$ which is decomposable into $(\exists x\, R(a, x)) \wedge (\exists y\, R(b, y))$ but admits no disjoint-vocabulary decomposition.

Lemma 4.3 allows us to establish a dichotomy for the constant-free *sjf-CRPQ*, that is CRPQs whose path atoms have pairwise distinct vocabularies, and more generally constant-free *cc-disjoint-CRPQ*, that is constant-free CRPQs whose connected components are over pairwise disjoint vocabularies. Of course the connected cases are not decomposable, but they are pseudo-connected hence Lemma 4.1 applies instead.

Corollary 4.6. *Let q be a constant-free cc-disjoint-CRPQ. Then* SVC_q *is in FP if it can be written as a safe UCQ, or #P-hard otherwise.*

5 PROOFS OF MAIN RESULTS

Our main results are general-purpose reductions from FGMC_q to SVC_q for C-hom-closed queries q verifying certain properties. Let us first give a general idea of the reductions we will use.

Given a partitioned database $\mathcal{D} = \mathcal{D}_\mathsf{n} \uplus \mathcal{D}_\mathsf{x}$ and a query q, our goal is to compute FGMC_q on $\mathcal{D}$ in polynomial time with the help of an oracle for SVC_q. The basic idea for doing this is to add a minimal support S of q, and make every fact exogenous except for one, denoted by μ. That is, $S = (S_\mathsf{n}, S_\mathsf{x})$ where $S_\mathsf{n} = \{\mu\}$. If the query has the right properties, the arrival of μ will have no impact on the satisfaction of q if, and only if, the set of players before μ forms a generalized support for q in $\mathcal{D}$. In such case, the Shapley value $\mathrm{Sh}(\mathcal{D}_\mathsf{n} \uplus S_\mathsf{n}, \mathbf{v}_q, \mu)$ on $(\mathcal{D}_\mathsf{n} \uplus S_\mathsf{n}, \mathcal{D}_\mathsf{x} \uplus S_\mathsf{x})$ will be an affine combination of the different counts of FGMC_q on $\mathcal{D}$, and by building variants we obtain a system of linear equations that can be inverted to compute the values in FGMC_q on $\mathcal{D}$.

However, for this technique to be successful, one must ensure that no minimal support of q can intersect both $\mathcal{D}$ and S, otherwise there could be a subset of $\mathcal{D}_\mathsf{n} \cup S_\mathsf{n}$ which is a generalized support for q in $(\mathcal{D}_\mathsf{n} \uplus S_\mathsf{n}, \mathcal{D}_\mathsf{x} \uplus S_\mathsf{x})$, but not in $\mathcal{D}$.

5.1 Proofs for Lemmas 4.1 and 4.2

The proof is organized as follows. We first consider how to exploit the hypotheses of Lemma 4.2 to build the reduction, as the construction is slightly more involved due to the fact that use of different queries for the two problems (FGMC_{q_1} and $\mathrm{SVC}_{q_1 \wedge q_2}$). We then show that a very similar construction yields the same desired properties if we start instead from the hypotheses of Lemma 4.1. Finally we use said properties to establish correctness of the reduction.

Let q_1, q_2 satisfy the hypotheses of Lemma 4.2, and let $\mathcal{D}$ be the database on which we wish to compute FGMC_{q_1}. Let S'' be the minimal support of q_2 given by the hypotheses (named as "S_2" in the statement). If $\mathcal{D}_{\mathsf{x}} \models q_1$, then every subset of $\mathcal{D}_{\mathsf{n}}$ is a generalized support for q, hence counting is trivial. We shall therefore assume $\mathcal{D}_{\mathsf{x}} \not\models q_1$ from now on. We also assume *w.l.o.g.* that $\mathit{const}(\mathcal{D}) \cap \mathit{const}(S'') \subseteq C_1$ —we can achieve this by simply renaming the constants $\mathit{const}(S'') \setminus C_1$ by fresh ones in S''. Observe that all facts of $\mathcal{D} \cap S''$ are irrelevant to q_1 due to item (2c). We may therefore assume *w.l.o.g.* that there are no such facts. To see why, let $\alpha \in \mathcal{D} \cap S''$. Then from $\mathrm{FGMC}_{q_1}(\mathcal{D} \setminus \{\alpha\})$, one may use Proposition 3.3-(1) to compute $\mathrm{SPPQE}_{q_1}((\mathcal{D} \setminus \{\alpha\})^p)$ for any $p \in (0,1)$, where $(\mathcal{D} \setminus \{\alpha\})^p$ denotes the probabilistic version of $\mathcal{D} \setminus \{\alpha\}$ where all endogenous (resp. exogenous) facts have probability p (resp. probability 1). Since α is irrelevant to q_1, $\mathrm{SPPQE}_{q_1}((\mathcal{D} \setminus \{\alpha\})^p) = \mathrm{SPPQE}_{q_1}(\mathcal{D}^p)$, and from there we can use again the equivalence of Proposition 3.3-(1) to obtain $\mathrm{FGMC}_{q_1}(\mathcal{D})$.

To build our reduction, we shall first "complete" $\mathcal{D}$ into $\mathcal{D}'$ adding facts so that $\mathrm{FGMC}_{q_1}(\mathcal{D}) = \mathrm{FGMC}_{q_1 \wedge q_2}(\mathcal{D}')$. Then we "duplicate" part of a minimal support of q_1 in order to compute $\mathrm{FGMC}_{q_1 \wedge q_2}(\mathcal{D}')$ using calls to an $\mathrm{SVC}_{q_1 \wedge q_2}$ oracle.

Completion. Take $\mathcal{D}' \mathrel{\hat{=}} \mathcal{D} \uplus S''$, where $\mathcal{D}'_{\mathsf{x}} = \mathcal{D}_{\mathsf{x}} \uplus S''$ (*i.e.*, all facts of S'' are exogenous). Observe that $\mathcal{D}$ and $\mathcal{D}'$ have the same endogenous facts. We next show that $\mathrm{FGMC}_{q_1}(\mathcal{D}) = \mathrm{FGMC}_{q_1 \wedge q_2}(\mathcal{D}')$.

For the left-to-right inclusion, let $B \subseteq \mathcal{D}_{\mathsf{n}}$ be a generalized support for q_1 in $\mathcal{D}$ (*i.e.*, $B \cup \mathcal{D}_{\mathsf{x}} \models q_1$). Since $S'' \models q_2$ and q_2 is C_2-hom-closed, it follows that $B \cup \mathcal{D}'_{\mathsf{x}} = B \cup \mathcal{D}_{\mathsf{x}} \cup S'' \models q_1 \wedge q_2$.

To show the right-to-left inclusion, suppose towards a contradiction that we have a set $B \subseteq \mathcal{D}_{\mathsf{n}}$ such that (i) $B \cup \mathcal{D}'_{\mathsf{x}} \models q_1 \wedge q_2$, and (ii) $B \cup \mathcal{D}_{\mathsf{x}} \not\models q_1$. In particular, (i) implies that there exists a minimal support S of q_1 in $B \cup \mathcal{D}'_{\mathsf{x}}$. Since we have assumed that $B \cup \mathcal{D}_{\mathsf{x}} \not\models q_1$ and $S'' \not\models q_1$, this means that S must intersect both $\mathcal{D}$ and S''. We can thus pick facts $\alpha_1 \in S \cap \mathcal{D}$ and $\alpha_2 \in S \cap S''$. Since q_1 is variable-connected, S is the C_1-homomorphic image of some minimal support $S^\dagger$ in which every atom is connected to every other via constants outside of C_1. In other words, there is a path P in the incidence graph $\mathbf{G}_S$ linking α_1 and α_2 only via atoms and constants that are the homomorphic image of a constant from $\mathit{const}(S^\dagger) \setminus C_1$. However, since the only vertices that $\mathbf{G}_{\mathcal{D}}$ and $\mathbf{G}_{S''}$ have in common are the constants in C_1, the path P must go through one of them, which is a contradiction because it would witness a q_1-leak in S''.

Duplication. Take S to be the q_1-leak-free support of q_1 given by the lemma hypotheses. Since q_1 is variable-connected, S is the C_1-homomorphic image of some minimal support $S^\dagger$ in which every atom is connected to every other by some constant outside of C_1. There cannot be any q_1-leak in $S^\dagger$ either, otherwise the homomorphism would transfer it to S. By replacing S with $S^\dagger$ we can therefore assume *w.l.o.g.* that it satisfies this extra property, and by C_1-isomorphically renaming it we can further assume that it shares no constant with $\mathcal{D}'$ or $C_1 \cup C_2$, except for those in C_1.

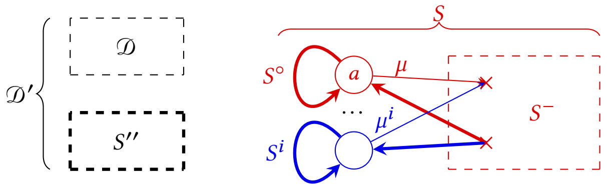


Figure 3: Illustration of the construction of $\mathcal{A}^i$. Parts that are graphically disconnected do not share any constant except for those that appear in C. New exogenous facts are in bold.

At this stage, we can get some information about $\mathrm{FGMC}_{q_1}(\mathcal{D})$ by computing a Shapley value in $\mathcal{D}' \cup S$, but in order to actually compute the value we will need a way to "duplicate" some facts of S.

To this end, we take any constant $a \in \mathit{const}(S) \setminus (C_1 \cup C_2)$ and partition S into the set S° of facts that contain a and the set S^- of facts that do not. We then build, for some parameter i, a family $S^1, \ldots, S^i$ of isomorphic copies of S° by replacing a with fresh constants. Fix a distinguished fact μ of S° and for every k denote by μ^k the corresponding fact in S^k. Finally, consider the database

$$\mathcal{A}^i \mathrel{\hat{=}} \mathcal{D}' \cup S^\circ \cup S^1 \cup \cdots \cup S^i \cup S^-$$

depicted in Figure 3. We partition $\mathcal{A}^i$ by letting the set of endogenous facts consist of:

- the endogenous facts in $\mathcal{D}'$, *i.e.* $\mathcal{D}_{\mathsf{n}}$ (recall that $S'' \subseteq \mathcal{D}_{\mathsf{x}}'$)
- the fact μ and each of its copies μ^k,
- all facts in S^-,

and having all remaining facts as exogenous.

Note that we deviate slightly from the proof idea by only duplicating a part of S. This is so that, aside from S'', the only exogenous facts that we add in our construction are those in $S^\circ \setminus \{\mu\}$ and their copies. In particular none is added if $S'' = \varnothing$ and S° is a singleton. This will prove important later in Section 6.1 when we consider purely endogenous databases.

Putting everything together. We now use a $\mathrm{SVC}_{q_1 \wedge q_2}$ oracle to compute the Shapley value of μ in the partitioned database $\mathcal{A}^i$. This value is given by the formula:

$$\mathrm{Sh}(\mathcal{A}^i_{\mathsf{n}}, \mathbf{v}, \mu) = \sum_{B \subseteq \mathcal{A}^i_{\mathsf{n}} \setminus \{\mu\}} \frac{|B|!(|\mathcal{A}^i_{\mathsf{n}}| - |B| - 1)!}{|\mathcal{A}^i_{\mathsf{n}}|!} (\mathbf{v}(B \cup \{\mu\}) - \mathbf{v}(B)),$$

where we abbreviate $\mathbf{v}_{q_1 \wedge q_2}$ as $\mathbf{v}$.

We aim to express $\mathrm{Sh}(\mathcal{A}^i_{\mathsf{n}}, \mathbf{v}, \mu)$ as a linear combination (plus constant) of the different numbers GSMC_j of generalized supports for $q_1 \wedge q_2$ in $\mathcal{D}$ of size j. To do so, we observe that our construction satisfies the following lemma.

Lemma 5.1. $(\mathbf{v}(B \cup \{\mu\}) - \mathbf{v}(B)) = 0$ *iff one of the following three mutually exclusive cases holds:*

(1) $\exists k \in [i]. \mu^k \in B$
(2) $\forall k \in [i]. \mu^k \notin B \ \wedge\ \exists \alpha \in S^-.\ \alpha \notin B$
(3) $\forall k \in [i]. \mu^k \notin B \ \wedge\ \forall \alpha \in S^-.\ \alpha \in B \ \wedge\ (B \cap \mathcal{D}_{\mathsf{n}}) \cup \mathcal{D}_{\mathsf{x}} \models q_1$

Case (1) is when there is already a copied fact in B that makes μ irrelevant. Case (2) is when, although μ has not been made redundant, there is some other fact of S^- missing to satisfy q_1. Case (3) is when the arrival of μ could have made an impact but q_1 was already satisfied by $(B \cap \mathcal{D}_\mathsf{n}) \cup \mathcal{D}_\mathsf{x}$. Recall that we only need to consider q_1 because the q_2 component has been dealt with by the addition of S''. In all other cases: (i) $B \cup \{\mu\} \cup \mathcal{A}^i_\mathsf{x} \models q_1$ because the presence of $S^- \cup \{\mu\}$ suffices to satisfy the query, and (ii) $B \cup \mathcal{A}^i_\mathsf{x} \not\models q_1$ because of the following observations. Assume for a contradiction that there is a minimal support of q_1 in $B \cup \mathcal{A}^i_\mathsf{x}$. It cannot be S because $\mu \notin B$, nor one of its copies otherwise case (1) would be met, nor can it be contained in S'' because $S'' \not\models q_1$, nor in $\mathcal{D}$ otherwise case (3) would be met. The only remaining possibility is that the minimal support intersects several of the three parts, but since they have no constant in common outside of C_1 and q_1 is variable-connected, this witnesses a q_1-leak, hence a contradiction.

Adapting the Construction for Lemma 4.1. Before finishing the correctness argument, we shall show that almost the same construction can be employed if we start with the hypotheses of Lemma 4.1. Letting q be the input pseudo-connected query, we shall follow the preceding construction, but using $q_1 \doteq q$ (and C for C_1), $q_2 \doteq \top$ (i.e. the trivial, always-true query), and S an island minimal support for q containing some $a \in \mathrm{Const}(S) \setminus C$. An isomorphic renaming ensures that no constant is shared between S and $\mathcal{D}$ besides those in C. We then observe that if any of the conditions of Lemma 5.1 is met, then μ is redundant for the exact same reason as before, and otherwise the only way of having $(\mathbf{v}(B \cup \{\mu\}) - \mathbf{v}(B)) = 0$ is to have a minimal support of q that intersects both B and S or one of its copies. However, this is precisely forbidden by the fact that S is an island support for q.

Concluding. We finally show that Lemma 5.1 suffices to conclude. Consider the complement of the formula above, that is:

$$1 - \mathrm{Sh}(\mathcal{A}^i_\mathsf{n}, \mathbf{v}, \mu) = \sum_{\substack{B \subseteq \mathcal{A}^i_\mathsf{n} \setminus \{\mu\},\\ B \text{ verifies (1), (2) or (3)}}} \frac{|B|!(|\mathcal{A}^i_\mathsf{n}| - |B| - 1)!}{|\mathcal{A}^i_\mathsf{n}|!}$$

It suffices to enumerate these cases since in all others $(\mathbf{v}(B \cup \{\mu\}) - \mathbf{v}(B)) = 1$ by Lemma 5.1. Moreover, the contributions of the subsets in case (1) and (2) are constants that can be calculated and subtracted out. We are thus left with the problem of computing

$$Sh^i \doteq \sum_{B \subseteq \mathcal{A}^i_\mathsf{n} \setminus \{\mu\}, B \text{ verifies (3)}} \frac{|B|!(|\mathcal{A}^i_\mathsf{n}| - |B| - 1)!}{|\mathcal{A}^i_\mathsf{n}|!}.$$

Since the B sets in the summation do not contain μ nor any μ^k and contain S^-, they are exactly the sets of the form $G \cup S^-$ where G is a generalized support for q_1 in $\mathcal{D}$. Grouping the G's by size and simplifying, one obtains the following formula.

$$Sh^i = \sum_{j=0}^{|\mathcal{D}_\mathsf{n}|} \frac{(j + |S^-|)!(|\mathcal{D}_\mathsf{n}| + i - j)!}{(|\mathcal{D}_\mathsf{n}| + i + |S^-| + 1)!} \cdot GSMC_j$$

Finally, by performing this computation for i ranging from 0 to $|\mathcal{D}_\mathsf{n}|$, we get a system of equations linking the Sh^i to the $GSMC_j$ whose underlying matrix can be reduced (multiplying every line by $(|\mathcal{D}_\mathsf{n}| + i + |S^-| + 1)!$, dividing every column by $(j + |S^-|)!$ then reversing the column order) to the matrix of general term $(i + j)!$ which by [2, proof of Theorem 1.1] is invertible.

5.2 Proof for Lemma 4.3

Let q be decomposable into $q_1 \wedge q_2$ and $\mathcal{D}$ be an input partitioned database. Recall that $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SPPQE}_q$ by Proposition 3.3. By definition, no fact can be relevant to both q_1 and q_2, hence $\mathcal{D}$ can be partitioned into $\mathcal{D}^1 \uplus \mathcal{D}^2$ such that no fact of $\mathcal{D}^1$ (resp. $\mathcal{D}^2$) is relevant to q_2 (resp. q_1). Therefore, the SPPQE_q problem over $\mathcal{D}$ can be solved by computing separately the SPPQE_{q_i} over $\mathcal{D}_i$, then multiplying the probabilities together. Using Proposition 3.3 in the other direction, we can obtain the latter values by computing FGMC_{q_1} (resp. FGMC_{q_2}) over $\mathcal{D}_1$ (resp. $\mathcal{D}_2$). Due to symmetry, it suffices to show how to compute FGMC_{q_1} over $\mathcal{D}_1$.

To this end, we build the same reduction as in Section 5.1 with $S'' \doteq \varnothing$ except that we use any minimal support of q_2 as S. The reduction will be successful because μ can only contribute if $B \cup \mathcal{D}_\mathsf{x} \models q_1$, and the facts in S'' are irrelevant to q_1 since they are relevant to q_2. A detailed proof can be found in Appendix C.

6 EXTENSIONS AND VARIANTS

This section explores variants of the considered problems, namely: computing the Shapley value on databases without exogenous facts (§6.1), extending the reductions to queries with negation (§6.2), computing the maximum Shapley value (§6.3), and initiating the study of the Shapley value of constants rather than facts (§6.4).

6.1 Purely Endogenous Databases

Having the possibility of fixing some exogenous facts makes the SVC problem more general and flexible. However, often we will be presented with an unpartitioned database, in which case all facts should be treated as endogenous and taken as players in the Shapley game. Unfortunately, SVC hardness results crucially rely on the existence of exogenous facts, and whether similar hardness results hold for purely endogenous databases is unclear. While we do not resolve this challenging question here, we make steps towards understanding how $\mathrm{SVC}^\mathsf{n}_q$, the restriction of SVC_q to partitioned databases of the form $\mathcal{D} = (\mathcal{D}_\mathsf{n}, \varnothing)$, is related to other problems.

Let us start by observing that the proof of $\mathrm{SVC}_q \leq_{poly} \mathrm{FGMC}_q$ in Proposition 3.3 adds an exogenous fact to the input database, meaning it cannot prove $\mathrm{SVC}^\mathsf{n}_q \leq_{poly} \mathrm{FMC}_q$ immediately. However, the following lemma shows that a constant number of exogenous facts is not a problem under polynomial-time Turing-reductions.

Lemma 6.1. *Fix $k \geqslant 1$ and consider a query q and instance $\mathcal{D} = (\mathcal{D}_\mathsf{n}, \mathcal{D}_\mathsf{x})$ of FGMC_q with $|\mathcal{D}_\mathsf{x}| = k$. Then FGMC_q on $\mathcal{D}$ can be solved by a polynomial-time algorithm that performs 2^k calls to an oracle for FMC_q.*

Corollary 6.1 (of Lemma 6.1 and proof of Proposition 3.3). *For every query q, we have $\mathrm{SVC}^\mathsf{n}_q \leq_{poly} \mathrm{FMC}_q$.*

We now turn to the reductions in the other direction, useful for showing hardness. By adding a hypothesis, we can adapt Lemma 4.1 and Lemma 4.3 to the purely endogenous setting:

Lemma 6.2 (Adaptation of Lemma 4.1). *Let q be a C-hom-closed query that has an island minimal support S and a constant $a \notin C$ appearing in exactly one fact of S. Then* $\mathrm{FMC}_q \leq_{poly} \mathrm{SVC}_q^{\mathsf{n}}$.

Note for instance that for any connected minimal CQ q without constants and having a variable which is not part of a join, we obtain $\mathrm{FMC}_q \leq_{poly} \mathrm{SVC}_q$. Due to space constraints, the adaptation of Lemma 4.3 is deferred to Appendix D.1.

6.2 Queries with Negation

PQE and SVC have been explored beyond C-hom-closed queries, by considering queries with some form of negation. An FP/#P dichotomy has been established for SVC over self-join-free conjunctive queries with safe negations (sjf-CQ$^\neg$) in [12, Theorem 3.1], and another dichotomy for PQE has been shown for the larger class of self-join-free 1RA$^-$ queries [6, Theorem 1.1]. In fact, it follows from the proof of [6] that this latter dichotomy is also valid for PQE $\left(\frac{1}{2};1\right)$, since no other probability is used. Briefly, a sjf-CQ$^\neg$ query is a sjf-CQ that may have some negative atoms, with the restriction that their variables are present in the positive part. The hierarchical queries are defined as before (except the atoms can be negative), and they also characterize the tractable queries.

A slight adaptation of our proof technique yields the following:

Proposition 6.1. *Let q be a sjf-CQ$^\neg$, where q^+ and q^- are its positive and negative atoms, respectively. Let q^+_{vc} be any maximal variable-connected subquery of q^+, and q^-_{vc} be the atoms of q^- whose every variable is in q^+_{vc}. Then,* $\mathrm{FGMC}_{q^+_{vc} \wedge q^-_{vc}} \leq_{poly} \mathrm{SVC}_q$.

The preceding result does not cover the full dichotomy of sjf-CQ$^\neg$ because of the basic non-hierarchical query $A(x) \wedge \neg S(x,y) \wedge B(y)$. However, it does cover all queries with a variable-connected positive part, as well as sjf-CQ$^\neg$ queries with "component-guarded negation", defined as the sjf-CQ$^\neg$ queries such that the set of variables of every negative atom appears in the same maximal variable-connected subquery of the positive part. Finally, our technique also applies to some queries in 1RA$^-$ which are not definable in CQ$^\neg$, for which SVC was not yet known to be hard (*cf.* Examples D.1 and D.2 in Appendix D.2.3).

6.3 Maximum Shapley Value

A main reason for considering the Shapley value is to identify the most important facts for a given query. One might therefore consider the problem of computing the top contributor and its Shapley value. Concretely, let max-SVC$_q$ be the problem of, given a database $\mathcal{D}$, outputting any fact $\alpha \in \mathcal{D}_{\mathsf{n}}$ and its Shapley value v_α for q, such that v_α is the maximum value among all (endogenous) facts. In fact, all of our reductions to show $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_q$ can be adapted to show $\mathrm{FGMC}_q \leq_{poly}$ max-SVC$_q$, leading to the conjecture that max-SVC might be equivalent to SVC.

Proposition 6.2. *For any pair of queries q, q' for which we obtain a reduction* $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_{q'}$ *as a consequence of Lemma 4.1, 4.2 or 4.3, we have* $\mathrm{FGMC}_q \leq_{poly}$ max-SVC$_{q'}$.

Observe that max-SVC is not the same problem as the "maximum SVC$_q$ contributor problem", that is, the problem of finding a fact with maximum Shapley value. We believe this is a relevant problem which may, in principle, be simpler to address than SVC.

6.4 Shapley Value of Constants

Instead of computing the Shapley values of facts, one could consider a setting in which the players are a set of "endogenous constants". Likewise, one could consider supports consisting of constants, or assign independent probabilities to the constants instead of facts.

As an example, consider a simple schema containing two binary relations Publication(authorID, paperID) and Keyword(paperID, keywordStr) containing publications and related keywords, and the query $q^* = \exists x, y$ Publication$(x, y) \wedge$ Keyword$(y,$ `'Shapley'`$)$ testing the existence of `'Shapley'`-related papers. One could then extract the experience level of authors on this topic by computing the Shapley value of author constants, treating all constants as exogenous except those in the authorID column. Note that the Shapley value for facts would be much less informative as an author's expertise may be split across several facts.

This idea gives rise to variants of PQE, FGMC and SVC. For any database $\mathcal{D}$ and $C \subseteq$ Const, let $\mathcal{D}|_C$ denote the database induced by the constants C. For a monotone query q, let SVC_q^{const} be the task of computing, for a database $\mathcal{D}$, a partition $const(\mathcal{D}) = C_{\mathsf{n}} \uplus C_{\mathsf{x}}$, and a constant $c \in C_{\mathsf{n}}$, the Shapley value in the cooperative game where the set of players is C_{n} and the wealth function assigns 1 to any subset $C \subseteq C_{\mathsf{n}}$ such that $\mathcal{D}|_{C \cup C_{\mathsf{x}}} \models q$ and $\mathcal{D}|_{C_{\mathsf{x}}} \not\models q$, or 0 otherwise. Similarly, let FGMC_q^{const} be the task of computing, for any $k \in \mathbb{N}$, database $\mathcal{D}$ and $const(\mathcal{D}) = C_{\mathsf{n}} \uplus C_{\mathsf{x}}$, the number of sets $C \subseteq C_{\mathsf{n}}$ of size k such that $\mathcal{D}|_{C \cup C_{\mathsf{x}}} \models q$. The variants $\mathrm{SVC}_q^{\mathsf{n},const}$ and FMC_q^{const} (and the probabilistic variants) can be defined analogously.

By a simple adaptation of our proof of Lemma 4.1, we can show a polynomial equivalence result for these variants:

Proposition 6.3. *For every hom-closed query q,* $\mathrm{SVC}_q^{const} \equiv_{poly} \mathrm{FGMC}_q^{const}$, *and* $\mathrm{SVC}_q^{\mathsf{n},const} \equiv_{poly} \mathrm{FMC}_q^{const}$.

In fact, this proposition can be extended to apply more generally to C-hom-closed queries provided the constants in C are exogenous – in particular, capturing the example query q^*.

We note that in the graph database setting these variants are equivalent to considering Shapley values of *nodes* instead of *edges*. This problem was considered in [9, §6], where the authors showed that SVC_q^{const} is #P-hard for any full CRPQ with a "non-redundant" atom whose language contains a word of length at least four.

7 DISCUSSION

After having identified FGMC as the suitable counting analog of SVC, we have exhibited reductions from FGMC to SVC. As the hypotheses of our reductions are formulated generally, they can be combined with existing hardness results for model counting or probabilistic query evaluation to establish #P-hardness for SVC for a range of query classes. In particular, we have obtained an FP/#P dichotomy for constant-free unions of connected CQs, a relevant stepping stone towards the conjectured dichotomy for UCQs. In some cases, our reductions allow us not only to transfer hardness results but also to establish the polynomial-time equivalence between FGMC_q and SVC_q, providing the strongest evidence as of yet that Shapley value computation is, from a complexity perspective, nothing other than a counting problem.

As is apparent from Figure 2, most of our new results concern constant-free queries, as the presence of constants in queries make reductions from FGMC considerably more difficult. While there are known methods for eliminating constants from queries in the probabilistic setting (most notably, the so-called query "shattering" [4, §2.5]), they interact badly with our hypotheses. For example, the shattering of a connected variable-connected query may become disconnected, and not even pseudo-connected, as illustrated by Example E.1. It is therefore remains an important question for future work how best to eliminate or otherwise handle constants when reducing FGMC to SVC.

Aside from extending our reductions to larger classes of queries, there are other interesting questions to explore related to the variants from Section 6. In particular, we would like to know whether the purely endogenous variant $\mathrm{SVC}^{\mathrm{n}}$ is as hard as SVC, which appears to be related to the open question of whether there exist queries for which MC is tractable but not GMC. We are also eager to understand better the Shapley value of constants, which we find natural. Due to Proposition 6.3, we can alternatively consider the model counting analog FGMC_q^{const}, which is conceptually simpler.

REFERENCES

[1] Antoine Amarilli. Uniform Reliability for Unbounded Homomorphism-Closed Graph Queries. In *International Conference on Database Theory (ICDT)* (2023) *(Leibniz International Proceedings in Informatics (LIPIcs))*, Floris Geerts and Brecht Vandevoort (Eds.), Vol. 255. Leibniz-Zentrum für Informatik, 14:1–14:17. https://doi.org/10.4230/LIPIcs.ICDT.2023.14

[2] Roland Bacher. 2002. Determinants of Matrices Related to the Pascal Triangle. *Journal de théorie des nombres de Bordeaux* 14, 1 (2002), 19–41. https://doi.org/10.5802/jtnb.344

[3] Nilesh N. Dalvi and Dan Suciu. 2004. Efficient Query Evaluation on Probabilistic Databases. In *International Conference on Very Large Data Bases (VLDB)*, Mario A. Nascimento, M. Tamer Özsu, Donald Kossmann, Renée J. Miller, José A. Blakeley, and K. Bernhard Schiefer (Eds.). Morgan Kaufmann, 864–875. https://doi.org/10.1016/B978-012088469-8.50076-0

[4] Nilesh N. Dalvi and Dan Suciu. 2012. The dichotomy of probabilistic inference for unions of conjunctive queries. *J. ACM* 59, 6 (2012), 30:1–30:87. https://doi.org/10.1145/2395116.2395119

[5] Daniel Deutch, Nave Frost, Benny Kimelfeld, and Mikaël Monet. 2022. Computing the Shapley Value of Facts in Query Answering. In *ACM SIGMOD International Conference on Management of Data (SIGMOD)*. ACM Press, 1570–1583. https://doi.org/10.1145/3514221.3517912

[6] Robert Fink and Dan Olteanu. 2016. Dichotomies for Queries with Negation in Probabilistic Databases. *ACM Trans. Database Syst.* 41, 1, Article 4 (feb 2016), 47 pages. https://doi.org/10.1145/2877203

[7] Ahmet Kara, Dan Olteanu, and Dan Suciu. 2023. From Shapley Value to Model Counting and Back. *CoRR* abs/2306.14211 (2023). https://doi.org/10.48550/ARXIV.2306.14211 To appear in PODS'24.

[8] Batya Kenig and Dan Suciu. 2021. A Dichotomy for the Generalized Model Counting Problem for Unions of Conjunctive Queries. In *ACM Symposium on Principles of Database Systems (PODS)*. ACM Press, 312–324. https://doi.org/10.1145/3452021.3458313

[9] Majd Khalil and Benny Kimelfeld. 2023. The Complexity of the Shapley Value for Regular Path Queries. In *International Conference on Database Theory (ICDT) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 255. Leibniz-Zentrum für Informatik, 11:1–11:19. https://doi.org/10.4230/LIPICS.ICDT.2023.11

[10] Ester Livshits, Leopoldo E. Bertossi, Benny Kimelfeld, and Moshe Sebag. 2021. The Shapley Value of Tuples in Query Answering. *Logical Methods in Computer Science (LMCS)* 17, 3 (2021). https://doi.org/10.46298/LMCS-17(3:22)2021

[11] Ester Livshits and Benny Kimelfeld. 2022. The Shapley Value of Inconsistency Measures for Functional Dependencies. *Log. Methods Comput. Sci.* 18, 2 (2022). https://doi.org/10.46298/lmcs-18(2:20)2022

[12] Alon Reshef, Benny Kimelfeld, and Ester Livshits. 2020. The Impact of Negation on the Complexity of the Shapley Value in Conjunctive Queries. In *ACM Symposium on Principles of Database Systems (PODS)*. ACM Press, 285–297. https://doi.org/10.1145/3375395.3387664

[13] Lloyd S Shapley. A Value for N-Person Games. In *Contributions to the Theory of Games II*, Harold W. Kuhn and Albert W. Tucker (Eds.). Princeton University Press, 307–317.

A APPENDIX TO SECTION 3 "STUDIED PROBLEMS"

Proposition 3.3. *For any Boolean query q, all the polynomial-time reductions denoted by the solid arrows of Figure 1 hold. In particular:*

(1) $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SPPQE}_q$;
(2) $\mathrm{FMC}_q \equiv_{poly} \mathrm{SPQE}_q$;
(3) $\mathrm{SVC}_q \leq_{poly} \mathrm{FGMC}_q$.

Moreover, the reductions for (1) and (2) only use the oracle on inputs with the same associated partitioned database.

Proof. Since the other reductions were justified in the main body, we shall only prove here the three remaining reductions stated above, by adapting the proof of [5, Proposition 3.1]. For clarity we will split them into several lemmas.

Lemma A.1. *For every Boolean query q, $\mathrm{SVC}_q \leq_{poly} \mathrm{FGMC}_q$.*

Proof. Recall the formula for Sh:

$$\mathrm{Sh}(\mathcal{D}_\mathrm{n}, v, \mu) = \sum_{B\subseteq \mathcal{D}_\mathrm{n}\setminus\{\mu\}} \frac{|B|!(|\mathcal{D}_\mathrm{n}|-|B|-1)!}{|\mathcal{D}_\mathrm{n}|!}(\mathbf{v}_q(B\cup\{\mu\})-\mathbf{v}_q(B))$$

Now denote by n the size of $|\mathcal{D}_\mathrm{n}|$ and by $\mathrm{FGMC}_j(q)(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$ the answer of FGMC_q for the partitioned database $(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$. Also abbreviate $j!(|\mathcal{D}_\mathrm{n}|-j-1)!/|\mathcal{D}_\mathrm{n}|!$ as C_j. Grouping the Bs by their sizes, the formula becomes as follows (in the first FGMC term μ becomes exogenous and in the second it is removed).

$$\mathrm{Sh}(\mathcal{D}_\mathrm{n}, v, \mu) = \sum_{j=0}^{n} C_j\left[\mathrm{FGMC}_j(q)(\mathcal{D}_\mathrm{n}\setminus\{\mu\}, \mathcal{D}_\mathrm{x}\cup\{\mu\})\right.$$
$$\left. - \mathrm{FGMC}_j(q)(\mathcal{D}_\mathrm{n}\setminus\{\mu\}, \mathcal{D}_\mathrm{x})\right]$$

Assuming an oracle to FGMC_q, one can thus obtain $\mathrm{Sh}(\mathcal{D}_\mathrm{n}, v, \mu)$ by straightforward arithmetic computations. □

Lemma A.2. *For every Boolean query q, $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SPPQE}_q$. Moreover, the reductions preserve the underlying partitioned database.*

Proof. We keep the same notations as in the previous proof.

$\mathrm{SPPQE}_q \leq_{poly} \mathrm{FGMC}_q$: Given a probabilistic database $\mathcal{D}$ where all facts have probabilities in $\{p; 1\}$, denote by $(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$ its underlying partitioned database. Recall that $\mathcal{D}_\mathrm{x} \doteq \{f \in \mathcal{D} | \pi(f) = 1\}$ and $\mathcal{D}_\mathrm{n} \doteq \mathcal{D} \setminus \mathcal{D}_\mathrm{n}$. By denoting $z \doteq \frac{p}{1-p} \in \mathbb{Q}_+$, the proper probability can be written $\frac{z}{1+z}$. Then the following relation can be obtained without much difficulty:

$$(1+z)^n \Pr(\mathcal{D} \models q) = \sum_{j=0}^{n} z^j \mathrm{FGMC}_j(q)(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$$

This means that $\Pr(\mathcal{D} \models q)$ can be computed in polynomial time with access to an oracle to FGMC_q, and only making oracle calls over the same underlying partitioned database.

$\mathrm{FGMC}_q \leq_{poly} \mathrm{SPPQE}_q$: We wish to compute $\mathrm{FGMC}_j(q)(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$ for some input database $\mathcal{D} = \mathcal{D}_\mathrm{n} \uplus \mathcal{D}_\mathrm{x}$. Denote $n \doteq |\mathcal{D}_\mathrm{n}|$ and consider for all $z \in \mathbb{Q}_+$ the probabilistic database $\mathcal{D}_z$ whose underlying database is $(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$ and where every fact in $\mathcal{D}_\mathrm{n}$ has probability $\frac{z}{1+z}$. Then the following relation can be obtained as before:

$$(1+z)^n \Pr(\mathcal{D}_z \models q) = \sum_{j=0}^{n} z^j \mathrm{FGMC}_j(q)(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$$

From there one can obtain a system of linear equations by using the SPPQE_q oracle on $\mathcal{D}_z$ for $n+1$ distinct values of z. Since the underlying matrix is a Vandermonde with distinct coefficients, it is invertible hence the system can be solved to obtain in polynomial time the different $\mathrm{FGMC}_j(q)(\mathcal{D}_\mathrm{n}, \mathcal{D}_\mathrm{x})$. □

Lemma A.3. *For every Boolean query q, $\mathrm{FMC}_q \equiv_{poly} \mathrm{SPQE}_q$. Moreover, the reductions preserve the underlying partitioned database.*

Proof. It is a direct consequence of Lemma A.2: since the underlying partitioned database is preserved, the absence of exogenous facts in the counting instance will be equivalent to the absence of deterministic tuples in the probabilistic ones. □

□

B APPENDIX TO SECTION 4 "MAIN RESULTS"

Lemma B.1. *Every connected hom-closed query is pseudo-connected.*

Proof. Let q be a connected hom-closed query; as far as the definition of pseudo-connectedness is concerned, this means $C = \varnothing$. Take a minimal support S of q. Let S' be a set of facts with $const(S) \cap const(S') = \varnothing$, and S^* a minimal support of q in $S \cup S'$. It must be connected because q is, and since there is no fact in $S \cap S'$ that contains both a constant in $const(S)$ and one in $const(S')$ (plus every relation has a positive arity), this implies it must be fully contained in either S or S'. This makes S is an island support for q and q pseudo-connected. □

Corollary 4.2.

(1) For any connected hom-closed UCQ q, SVC_q is either in FP or #P-hard.
(2) For any connected hom-closed graph query q, SVC_q is either in FP or #P-hard. —in particular, this holds for any connected UCRPQ without constants.

Proof. (1) Let q be a connected hom-closed UCQ. By Corollary 4.1, $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SVC}_q$. Now there can be two cases:

- q is safe, in which case FGMC_q is tractable because PQE_q is [4];
- q is unsafe, in which case FGMC_q is intractable because GMC_q is [8].

(2) Let q be a connected hom-closed graph query q. Again, by Corollary 4.1, $\mathrm{FGMC}_q \equiv_{poly} \mathrm{SVC}_q$. Now there can be three cases:

- q can be written as a safe UCQ, in which case FGMC_q is tractable because PQE_q is [4];
- q can be written as an unsafe UCQ, in which case FGMC_q is intractable because GMC_q is [8];
- q is unbounded, in which case FGMC_q is intractable because MC_q is [1]. □

Lemma B.2. *Every (instantiated) RPQ whose language contains a word of length at least 2 is pseudo-connected.*

Proof. Let q be an RPQ. It can be written as a path atom $L(a,b)$ with L a regular language and a, b two constants that may be equal. q is C-hom-closed for $C \doteq \{a,b\}$. Consider a word $w = R_1 \dots R_l$ of length at least 2, and a simple path $P \doteq a \xrightarrow{R_1} a_1 \dots a_{l-1} \xrightarrow{R_l} b$ such that $\forall i \in [l-1].a_i \notin C$. P is a minimal support of q by definition, and $const(P) \nsubseteq C$. We now show that it is an island support.

Take a set of facts S' with $const(P) \cap const(S') \subseteq C$ and a minimal support $B \subseteq P \cup S'$ of q. It must be a path from a to b, and any path that connects a and b via some edge in P must contain P as a whole, since all edges of P contain some a_i and no edge of S' does. Therefore by minimality $B \subseteq P$ or $B \subseteq S'$. □

Corollary 4.3 (Originally established in [9]). *Let q be an RPQ of language L. If L contains a word of length at least 3, then* SVC$_q$ *is #P-hard, otherwise it is in FP.*

Proof. If L contains no word of length at least 2, then q can be written as a disjunction of atoms with no variable, which is trivially tractable for both FGMC and SVC (rather than the whole input database, it suffices to consider the constant-size subset of facts that could appear in a minimal support). Otherwise, q is pseudo-connected by Lemma B.2, hence Lemma 4.1 states that FGMC$_q \equiv_{poly}$ SVC$_q$.

It now suffices to prove that the desired dichotomy holds for FGMC. The tractability is easy: since all tractable queries in the dichotomy are bounded, the dichotomy for PQE [4] applies (see [9] for a detailed explanation of why these queries are safe). As for the hardness, we turn to the proof of [9, Theorem 4.1] where they showed the hardness of SVC$_q$ from the previously known hardness of SVC$_{q_{RST}}$, with $q_{RST} \doteq R(x) \wedge S(x,y) \wedge T(y)$. As it turns out, their constructions work equally well to show that GMC$_{q_{RST}} \leq_{poly}$ GMC$_q$, and in turn the hardness of GMC$_{q_{RST}}$ is well known [8]. □

Corollary 4.4. *For every C-hom-closed query q that admits a duplicable singleton support,* FGMC$_q \equiv_{poly}$ SVC$_q$.

Proof. Let q be a C-hom-closed query q that admits a duplicable singleton support S. If S isn't minimal, then q is the trivial query $\top$, thus FGMC$_q$ and SVC$_q$ are both obviously tractable. Otherwise, since it contains a constant outside of C, it only remains to show that it is an island support. Let S' be a set of facts with $const(S) \cap$ Const$(S') \subseteq C$ and S^* a minimal support for q in $S \cup S'$. If the single element of S is contained in S^*, then $S^* = S$ by minimality, otherwise $S^* \subseteq S'$. We have thus shown q to be pseudo-connected, and it suffices to apply Lemma 4.1. □

Corollary 4.5. *Let q be a non-hierarchical sjf-CQ or a non-hierarchical constant-free CQ. Then,* SVC$_q$ *is #P-hard.*

Proof. The fact that the CQ q is non-hierarchical can be seen in a variable-connected subquery. In other words, q can be written as $q_1 \wedge q_2$ with q_1 variable-connected and non-hierarchical. Now build for $i \in [2]$ a minimal support S_i of q_i by isomorphically mapping every variable to a fresh constant.

The key idea is that there cannot be any leak if the CQ is either constant-free or self-join-free. This is because a leak implies a C-homomorphism from an atom of the query to an atom of the support where a constant outside of C is mapped to a constant in C. Now if $C = \varnothing$ this is obviously impossible, and if q is self-join-free, then for every atom $\alpha \in S_i$, the only atom in q for which there exists a homomorphism is the one it is isomorphic to.

At this point the hypotheses (1), (2b) and (3) are verified. Now (2a) is free since q_2 can be removed if it is redundant, and (2c) must also be true because $const(\alpha) \subseteq C_1$ is impossible if the query is constant-free and if it is self-join-free no fact in a minimal support of q_2 can be relevant to q_1 because of the disjoint vocabularies.

We can thus apply Lemma 4.2 to obtain FGMC$_{q_1} \leq_{poly}$ SVC$_q$, and since q_1 is non-hierarchical, this means that SVC$_q$ is #P-hard because GMC$_{q_1}$ is [8]. □

Lemma 4.4. *A hom-closed Boolean query q is decomposable iff it is equivalent to $q_1 \wedge q_2$, where q_1 and q_2 are Boolean UCQ$^\infty$ queries that do not share any relation name.*

Proof. First observe that the first condition of decomposability is always satisfied when the query is hom-closed. It will thus be ignored from now.

($\Rightarrow$) Assume q is decomposable into $q_1 \wedge q_2$. We may further assume q_1 and q_2 are UCQ$^\infty$s since any hom-closed query can be written as such. Let us now assume by contradiction that q_1 and q_2 share a relation name P. We can assume w.l.o.g. that they admit respective minimal supports S_1 and S_2 that contain the relation name P: if q_i does not admit any minimal support containing P, then P can be removed from any conjunct of q_i it appears in without any consequence. By homomorphically renaming their constants, we can assume that both S_1 and S_2 contain the atom $P(\vec{a})$ that only contains the fresh constant a, which contradicts $S_1 \cap S_2 = \varnothing$.

($\Leftarrow$) Assume q can be written as $q_1 \wedge q_2$ where q_1 and q_2 are UCQ$^\infty$s that do not share any variable nor any relation name. Let S_1 and S_2 be respective supports of q_1 and q_2. Since they do not share any relation name then $S_1 \cap S_2 = \varnothing$. □

Corollary 4.6. *Let q be a constant-free cc-disjoint-CRPQ. Then* SVC$_q$ *is in FP if it can be written as a safe UCQ, or #P-hard otherwise.*

Proof. If q is connected then it is pseudo-connected by Lemma B.1, otherwise it is decomposable by Lemma 4.4. Either way, FGMC$_q \leq_{poly}$ SVC$_q$ by Lemma 4.1 or Lemma 4.3. Then, since q is a hom-closed graph query, it is in FP if it can be written as a safe UCQ, or #P-hard otherwise [1, 4, 8] (see the proof of Corollary 4.2). □

C APPENDIX TO SECTION 5 "PROOFS OF MAIN RESULTS"

Proof of Lemma 4.3

Let $q = q_1 \wedge q_2$ be a decomposable query and $\mathcal{D}$ an input partitioned database. As was explained in the proof sketch, $\mathcal{D}$ can be partitioned as $\mathcal{D}^1 \cap \mathcal{D}^2$, and it suffices to compute FGMC$_{q_1}$ over $\mathcal{D}^1$ to obtain the desired result.

Let S be a minimal support of q_2 that contains at least one constant a outside of C. Similarly to what has been done in the proof of Lemmas 4.1 and 4.2, we build the parametrized database $A^i \doteq \mathcal{D} \cup S^\circ \cup S^1 \cup \dots \cup S^i \cup S^-$ from the subset $S^\circ \subseteq S$ of facts that contain a, and use the oracle to compute the same Shapley value of some distinguished $\mu \in S^\circ$, given by the same formula:

$$\mathrm{Sh}(A^i_\mathsf{n}, \mathbf{v}_q, \mu) = \sum_{B \subseteq A^i_\mathsf{n} \setminus \{\mu\}} \frac{|B|!(|A^i_\mathsf{n}| - |B| - 1)!}{|A^i_\mathsf{n}|!} (\mathbf{v}_q(B \cup \{\mu\}) - \mathbf{v}_q(B))$$

Now we express $\mathrm{Sh}(A^i_\mathsf{n}, \mathbf{v}_q, \mu)$ as a linear combination (plus constant) of the different numbers $FGMC_j$ of generalized supports for q_1 in $\mathcal{D}^1$ with size j. For simplicity we will rather express it with their complement $\overline{FGMC_j}$ (number of subsets of size j that are *not* generalized supports) from which we can as easily obtain the value $\mathrm{FGMC}_{q_1}(\mathcal{D}^1)$ we want. For this we observe that, like what was obtained in Lemma 5.1, $(\mathbf{v}_q(B \cup \{\mu\}) - \mathbf{v}_q(B)) = 0$ in the following three mutually exclusive cases:

(1) $\exists k \in [i]. \mu^k \in B$
(2) $\forall k \in [i]. \mu^k \notin B \ \wedge \ \exists \alpha \in S^-. \alpha \notin B$
(3) $\forall k \in [i] \ \mu^k \notin B \ \wedge \ \forall \alpha \in S^- \ \alpha \in B \ \wedge \ (B \cap \mathcal{D}^1_\mathsf{n}) \cup \mathcal{D}^1_\mathsf{x} \not\models q_1$

In the first case, B contains a fact μ^k which completes the corresponding S^k and makes μ irrelevant in satisfying its component q_2 of the query. In the second case, S^- is incomplete, meaning μ cannot help to satisfy q_2. In the last case μ cannot contribute because the q_1 subquery isn't satisfied.

Note that in all other cases the addition of μ makes q_2 and q satisfied where they previously weren't: q is satisfied because of the respective models $(B \cap \mathcal{D}^1_\mathsf{n}) \cup \mathcal{D}^1_\mathsf{x}$ and S of q_1 and q_2, and q_2 previously wasn't because S is a minimal and the facts of $\mathcal{D}^1$ are irrelevant to q_2.

By taking the complement and removing a constant term as before, we are left with:

$$Sh^i \mathrel{\hat{=}} \sum_{B \subseteq A^i_\mathsf{n} \setminus \{\mu\},\, B \text{ verifies (3)}} \frac{|B|!(|A^i_\mathsf{n}| - |B| - 1)!}{|A^i_\mathsf{n}|!}.$$

Since the B sets do not contain μ nor any μ^k, they are exactly the sets that are *not* generalized supports of q_1 in $\mathcal{D}^1$. Grouping them by size, one obtains the following formula.

$$\begin{aligned} Sh^i &= \sum_{j=0}^{|\mathcal{D}^1_\mathsf{n}|} \frac{j!(|A^i_\mathsf{n}| - j - 1)!}{|A^i_\mathsf{n}|!} \cdot \overline{FGMC_j} \\ &= \sum_{j=0}^{|\mathcal{D}^1_\mathsf{n}|} \frac{j!(|\mathcal{D}^1_\mathsf{n}| + i - j)!}{(|\mathcal{D}^1_\mathsf{n}| + i + 1)!} \cdot \overline{FGMC_j} \end{aligned}$$

We can now conclude with the same linear algebra argument as in the proof of Lemmas 4.1 and 4.2.

D APPENDIX TO SECTION 6 "EXTENSIONS AND VARIANTS"

D.1 Purely Endogenous Databases

Lemma 6.1. *Fix $k \geqslant 1$ and consider a query q and instance $\mathcal{D} = (\mathcal{D}_\mathsf{n}, \mathcal{D}_\mathsf{x})$ of FGMC_q with $|\mathcal{D}_\mathsf{x}| = k$. Then FGMC_q on $\mathcal{D}$ can be solved by a polynomial-time algorithm that performs 2^k calls to an oracle for FMC_q.*

Proof. Denote by $\mathrm{FMC}_q{}^{(k)}$ the fixed-size generalized model counting problem restricted to inputs that contain at most k exogenous facts. For $k > 0$, let $\mathcal{D} = (\mathcal{D}_\mathsf{n}, \mathcal{D}_\mathsf{x})$ be an instance of FGMC_q, with $|\mathcal{D}_\mathsf{x}| = k$. Take any fact $\alpha \in \mathcal{D}_\mathsf{x}$. Then, the generalized supports of size j in $\mathcal{D}$ are exactly the generalized supports of size $j+1$ in $(\mathcal{D}_\mathsf{n} \cup \{\alpha\}, \mathcal{D}_\mathsf{x} \setminus \{\alpha\})$ that contain α, which leads to the following formula:

$$\begin{aligned} \mathrm{FMC}^{(k)}_j(q)(\mathcal{D}_\mathsf{n}, \mathcal{D}_\mathsf{x}) = \mathrm{FMC}^{(k-1)}_{j+1}(q)(\mathcal{D}_\mathsf{n} \cup \{\alpha\}, \mathcal{D}_\mathsf{x} \setminus \{\alpha\}) \\ - \mathrm{FMC}^{(k-1)}_{j+1}(q)(\mathcal{D}_\mathsf{n}, \mathcal{D}_\mathsf{x} \setminus \{\alpha\}) \end{aligned}$$

By recursively applying this formula we get the value we want after 2^k calls to FMC_q. □

We now wish to adapt Lemmas 4.1 and 4.3 to purely endogenous databases. This shall be done by defining a notion of query with an unshared constant, which will depend on the specific case. A C-hom-closed query is said to be *pseudo-connected with an unshared constant* if it has an island minimal support S and a constant $a \notin C$ appearing in exactly one fact of S. It is said to be *decomposable with an unshared constant* if it admits a decomposition $q_1 \wedge q_2$ that satisfies one of the following:

(1) for $i \in \{1, 2\}$, q_i admits a minimal support S_i with a constant $a_i \notin C_i$ appearing in exactly one fact of S_i;
(2) there exists $i \in \{1, 2\}$ such that q_i admits a minimal support S_i with a constant $a_i \notin C_i$ appearing in exactly one fact of S_i, *and* $\mathrm{FMC}_{q_i} \in \mathsf{FP}$.

Lemma D.1 (Adaptation of Lemmas 4.1 and 4.3). *Let q be a C-hom-closed query that is either pseudo-connected with an unshared constant or decomposable with an unshared constant. Then $\mathrm{FMC}_q \leq_{poly} \mathrm{SVC}^\mathsf{n}_q$.*

Note that this lemma restricted to pseudo-connected queries with an unshared constant is a simple reformulation of Lemma 6.2.

Proof. First consider the case where q is pseudo-connected with an unshared constant and apply the same construction as with the proof of Lemma 4.1, by using the island minimal support S given by the definition and choosing an unshared constant a as the constant to duplicate. Since q is also pseudo-connected, this construction reduces FGMC_q to SVC_q. Now by definition a appears in exactly one fact of S, which means that in the construction S° is a singleton. This implies that its only element has to be μ which is endogenous, hence no exogenous fact will be added during the construction. In other words, if the input database was purely endogenous then the one on which the oracle is called will be as well, hence $\mathrm{FMC}_q \leq_{poly} \mathrm{SVC}^\mathsf{n}_q$.

When q is decomposable with an unshared constant, if it is of type (1), we apply the exact same reasoning to compute both SPQE_{q_i} over $\mathcal{D}_i$ separately from oracles to $\mathrm{SVC}^\mathsf{n}_q$, using the minimal support $S_{i'}$ of the other subquery given by the definition. If it is of type (2), we compute $\mathrm{SPQE}_{q_{i'}}$ over $\mathcal{D}_{i'}$ this way, using the minimal support S_i given by the definition, and directly get SPQE_{q_i} on $\mathcal{D}_i$ from the fact that the problem is in FP. In both cases, we conclude by multiplying the probabilities together. □

D.2 Queries with Negation

D.2.1 Key lemma. We start by establishing a key lemma that will allow us to prove the other results about queries with negations. A *DNF formula* is a disjunction of conjunctions of possibly negated atoms, which we see as a generalized form of UCQ.

Lemma D.2. *Let $q = q^+ \wedge \neg q^- \wedge \neg\alpha_1 \wedge \cdots \wedge \neg\alpha_K$ be a Boolean query such that:*

- *$q^+ = q_1 \wedge q_2$ is a sjf-CQ whose constants are in C, with q_1 being variable-connected;*
- *q^- is a DNF formula over constants of C and variables from vars(q^+), such that every atom contains at least one variable and every clause contains at least a positive atom;*
- *q^- shares no relation name with q^+;*
- *for every i, we have that α_i is an atom over the constants of C which shares no relation name with q^+.*

And let $\tilde{q}^-$ be the query resulting from:

(1) removing from q^- every negative atom that contains some variable $x \notin$ vars(q_1) and
(2) removing every clause whose positive part contains some variable $x \notin$ vars(q_1).

Further, let $\tilde{q} \doteq q_1 \wedge \neg\tilde{q}^- \wedge \neg\alpha_1 \wedge \cdots \wedge \neg\alpha_K$. Then, $\mathrm{FGMC}_{\tilde{q}} \leq_{poly} \mathrm{SVC}_q$.

Proof. We build from q^+ the same reduction as in the proof of Lemma 4.1 (see Figure 3), with S and S'' respectively isomorphic to q_1 and q_2 (see the proof of Corollary 4.5 for details on how this reduction applies to sjf-CQs.). From this we prove the following adaptation of Lemma 5.1, with $\mathcal{F} \doteq \{\mu^k \mid k \in [i]\} \cup \{\alpha_k \mid k \in [K]\}$:

Lemma D.3. *$(\mathbf{v}(B \cup \{\mu\}) - \mathbf{v}(B)) = 0$ in the following four mutually exclusive cases:*

(1) $\exists k \in [K].\alpha_k \in B$,
(2) $\forall k \in [K].\alpha_k \notin B \ \wedge \ \exists k \in [i].\mu^k \in B$,
(3) $\forall \alpha \in \mathcal{F}.\alpha \notin B \ \wedge \ \exists \alpha \in S^-.\alpha \notin B$,
(4) $\forall \alpha \in \mathcal{F}.\alpha \notin B \ \wedge \ \forall \alpha \in S^-.\alpha \in B \ \wedge \ B \cap \mathcal{D}' \models \tilde{q}$.

Case 1 is when there is some α_k that invalidates the query independently of whether μ is present or not, and the other cases are strictly analogous to those in Lemma 5.1. The only difference here is the presence of a negative part. The α_k have been dealt with, so let us consider a minimal support M^- of q^- in $B \cup \mathcal{A}^i_{\mathrm{x}}$. Since q^+ and q^- have no relation name in common and everything in the reduction other than the input database $\mathcal{D}$ is part of a minimal support of q^+, then necessarily $M^- \subseteq \mathcal{D}$ (and $M^- \neq \varnothing$ because every clause in q^- contains at least one positive atom). Since every atom in q^- contains a variable, and by construction every variable in q^+ is matched to a fresh constant in S or S'', these two cannot be affected by M^-. Overall, the only impact M^- can have is on a minimal support of q_1 in $\mathcal{D}$, but this is taken into account by the condition $B \cap \mathcal{D}' \models \tilde{q}$ in Case 4. □

D.2.2 Application to sjf-CQ¬.

Proposition 6.1. *Let q be a sjf-CQ$^\neg$, where q^+ and q^- are its positive and negative atoms, respectively. Let q^+_{vc} be any maximal variable-connected subquery of q^+, and q^-_{vc} be the atoms of q^- whose every variable is in q^+_{vc}. Then, $\mathrm{FGMC}_{q^+_{vc} \wedge q^-_{vc}} \leq_{poly} \mathrm{SVC}_q$.*

Proof. Let $q^- = q^-_v \wedge q^-_c$ be, respectively, the atoms of q^- having at least one variable and the atoms having only constants. Assume $q^-_c = \{\neg\alpha_1, \ldots, \neg\alpha_K\}$. We can now apply Lemma D.2 by taking

- q^+_{vc} as q_1;
- the DNF of $\neg q^-_v$ as q^-;
- the α_i's as above.

In this context, $\tilde{q}^-$ will be the DNF of q^-_{vc} with the variable-free atoms removed, hence the result $\mathrm{FGMC}_{\tilde{q}} \leq_{poly} \mathrm{SVC}_q$ translates to $\mathrm{FGMC}_{q^+_{vc} \wedge q^-_{vc}} \leq_{poly} \mathrm{SVC}_q$, as desired. □

D.2.3 Beyond sjf-CQ¬.

Thanks to the dichotomy for sjf-1RA$^-$ [6], Lemma D.2 shows the hardness of SVC for some queries that cannot be expressed as sjf-CQ$^\neg$ as evidenced by the two following examples. The notations are directly taken from [6], to which we refer the interested reader.

Example D.1. Consider the Boolean 1RA$^-$ query

$$q_1 \doteq \pi_\varnothing(D \bowtie (S \bowtie (A - (B - C))))$$

over the schema $(D(X), S(X,Y), A(Y), B(Y), C(Y))$, which translates into the following first-order formula:

$$\begin{aligned} q_1 &\equiv \exists x, y\ D(x) \wedge S(x,y) \wedge A(y) \wedge \neg(B(y) \wedge \neg C(y)) \\ &\equiv \exists x, y\ \big(D(x) \wedge S(x,y) \wedge A(y) \wedge \neg B(y)\big) \vee \\ &\qquad \big(D(x) \wedge S(x,y) \wedge A(y) \wedge C(y)\big) \end{aligned}$$

The first form illustrates how Lemma D.2 can be applied on it by taking $B(y) \wedge \neg C(y)$ as q^-, while the second shows that this query is not equivalent to a sjf-CQ$^\neg$. However, it is a non-hierarchical sjf-1RA$^-$ by [6, Proposition 5.4] (using pattern $\mathbf{P}_{6.1}$ from [6, Figure 9]) hence its hardness for PQE $\left(\frac{1}{2}\right)$. △

Example D.2. Consider now the 1RA$^-$ query

$$q_2 \doteq \pi_\varnothing(S - (A \bowtie B))$$

over the schema $(S(X,Y), A(X), B(Y))$, which translates into the following first-order formula:

$$\begin{aligned} q_2 &\equiv \exists x, y\ S(x,y) \wedge \neg(A(x) \wedge B(y)) \\ &\equiv \exists x, y\ \big(S(x,y) \wedge \neg A(x)\big) \vee \big(S(x,y) \wedge \neg B(x)\big) \end{aligned}$$

Once again Lemma D.2 applies, the query is not equivalent to a sjf-CQ$^\neg$, and it is a non-hierarchical 1RA$^-$ (this time using the pattern $\mathbf{P}_{4.3}$ from [6, Figure 9]). △

As these examples show, sjf-1RA$^-$ allows for richer negations, that can be nested or contain more than one atom. However, note that, in order for Lemma D.2 to apply, these complex negations must always contain some variable in every atom.

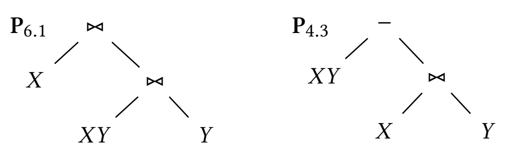


Figure 4: Reproductions of patterns from [6, Figure 9].

D.3 Maximum Shapley Value

Proposition 6.2. *For any pair of queries q, q' for which we obtain a reduction $\mathrm{FGMC}_q \leq_{poly} \mathrm{SVC}_{q'}$ as a consequence of Lemma 4.1, 4.2 or 4.3, we have $\mathrm{FGMC}_q \leq_{poly} \text{max-}\mathrm{SVC}_{q'}$.*

Proof. We first show that the Shapley value of a fact that is a generalized support on its own is always maximal, using the following lemma (whose proof is provided afterwards):

Lemma D.4. *Let $(P, \mathbf{v})$ be a monotone[3] binary[4] Shapley game that contains some $s \in P$ such that $\mathbf{v}(\{s\}) = 1$. Then:*

$$\forall p \in P.\ \ \mathrm{Sh}(P, \mathbf{v}, p) \leq \mathrm{Sh}(P, \mathbf{v}, s).$$

Note that if we assume that $\mathcal{D}_{\mathsf{x}} \not\models q$ (otherwise every fact has a zero hence maximal Shapley value), a fact μ that is a generalized support on its own indeed verifies $\mathbf{v}(\{s\}) = 1$. Now if we go back to the constructions used to prove Lemmas 4.1 to 4.3, they are equally valid if we take $S^\circ \mathrel{\hat{=}} S$ and $S^- \mathrel{\hat{=}} \varnothing$ (the only point of S^- was to limit the number of exogenous facts). If we do so, then μ, which is the only fact the SVC_q oracle is called on, is a singleton generalized support and therefore has a maximal Shapley value. $\square$

Proof of Lemma D.4. For every $p \in P$, denote $\mathcal{W}_p \mathrel{\hat{=}} \{\sigma \in \mathbb{P}(P) \mid \mathbf{v}(\sigma_{<p} \cup \{p\}) - \mathbf{v}(\sigma_{<p}) = 1\}$. Since the game is monotone and binary, $\mathbf{v}(\varnothing) = 0$ and $\mathbf{v}_q\{s\} = 1$, the $\mathcal{W}_p$ form a partition of $\mathbb{P}(P)$, and by Equation (1) $\mathrm{Sh}(P, \mathbf{v}, p)$ is proportional to $|\mathcal{W}_p|$.

Now let $p \in P$ and $\sigma \in \mathcal{W}_p$. Since the game is binary, then necessarily $\mathbf{v}(\sigma_{<p}) = 0$, which implies $s \notin \sigma_{<p}$ because the game is monotone. Therefore, if τ denotes the function that swaps the positions of s and p in a permutation, then $\mathbf{v}(\tau(\sigma)_{<s} \cup \{s\}) - \mathbf{v}(\tau(\sigma)_{<s}) = 1$ because $\tau(\sigma)_{<s} \subseteq \sigma_{<p}$ and $\mathbf{v}(\sigma_{<p}) = 0$. This means that $\tau_{|\mathcal{W}_p}$ injects in $\mathcal{W}_s$ hence $|\mathcal{W}_p| \leq |\mathcal{W}_s|$, hence $\mathrm{Sh}(P, \mathbf{v}, p) \leq \mathrm{Sh}(P, \mathbf{v}, s)$. $\square$

D.4 Shapley Value of Constants

Proposition 6.3. *For every hom-closed query q, $\mathrm{SVC}_q^{const} \equiv_{poly} \mathrm{FGMC}_q^{const}$, and $\mathrm{SVC}_q^{\mathsf{n},const} \equiv_{poly} \mathrm{FMC}_q^{const}$.*

Proof. The reduction from SVC_q^{const} (resp. $\mathrm{SVC}_q^{\mathsf{n},const}$) to FGMC_q^{const} (resp. FMC_q^{const}) can be obtained by directly applying the proofs of Proposition 3.3 and Corollary 6.1. For the other direction, we shall prove the stronger result that for any C-hom-closed query q, $\mathrm{FGMC}_q^{const}(\mathcal{D})$ (resp. $\mathrm{FMC}_q^{const}(\mathcal{D})$) can be computed in polynomial time from an oracle to SVC_q^{const} (resp. $\mathrm{SVC}_q^{\mathsf{n},const}$) provided that $\mathcal{D} \cap C \subseteq \mathcal{D}_{\mathsf{x}}$. The argument is a simple adaptation of the proof of Lemma 4.1.

Let $\mathcal{D}$ be an input database such that $\mathcal{D} \cap C \subseteq \mathcal{D}_{\mathsf{x}}$. If all minimal supports of q have their constants in C, then either all coalitions are generalized supports or none are, but either way the computation of $\mathrm{FGMC}_q^{const}(\mathcal{D})$ is trivial. Otherwise, take a minimal support S of q such that $const(S) \not\subseteq C$, and by C-homomorphically renaming it if necessary we can assume $const(S) \setminus C$ is reduced to a singleton $\{a_\mu\}$ with $a_\mu \notin const(\mathcal{D})$. Since the players are the constants, S is essentially a duplicable singleton support (see Corollary 4.4): in the formula that defines $\mathrm{Sh}(C_{\mathsf{n}} \cup \{a_\mu\}, \mathbf{v}_q, a_\mu)$, any coalition that contains a_μ will satisfy the query, and any that does not will not be contained in the input database.

Finally, note that no exogenous element (constants instead of facts in this setting) is added since S, the minimal support used in the construction, only contains a single constant. $\square$

E APPENDIX TO SECTION 7 "DISCUSSION"

Example E.1. Consider the *CQ* $R(x, y) \wedge S(a, x) \wedge S(x, a) \wedge T(x, z)$. It is variable-connected since every atom contains x but its complete shattering [4] contains the disjunct $R_{a,*}(y) \wedge S_{a,a}() \wedge T_{a,*}(z)$ which isn't connected. $\triangle$

[3]Such that $B \subseteq B' \Rightarrow \mathbf{v}_q(B) \leq \mathbf{v}_q(B')$.
[4]Such that $\mathbf{v}_q$ has an image $\{0, 1\}$.