Equivalence and Minimization of Conjunctive Queries Under Combined Semantics

Rada Chirkova
Department of Computer Science
NC State University, Raleigh, NC 27695, USA
chirkova@csc.ncsu.edu

ABSTRACT

The problems of query containment, equivalence, and minimization are recognized as very important in the context of query processing, specifically of query optimization. In their classic work [2] published in 1977, Chandra and Merlin solved the three problems for the language of conjunctive queries *(CQ queries)* on relational data, under the "set-semantics" assumption for query evaluation. (Under set semantics, both database relations and query answers are treated as sets.) While the results of [2] have been very influential in database research, it was recognized long ago that the set semantics does not correspond to the semantics of the standard commercial query language SQL. Alternative semantics, called *bag* and *bag-set semantics*, have been studied since 1993; Chaudhuri and Vardi in [4] outlined necessary and sufficient conditions for equivalence of CQ queries under these semantics. (The problems of containment of CQ bag and bag-set queries remain open to this day.) More recently, Cohen [5, 6] introduced a formalism for treating (generalizations of) CQ queries evaluated under each of set, bag, and bag-set semantics uniformly as special cases of the more general *combined semantics.* This formalism provides tools for studying broader classes of practical SQL queries, specifically important types of queries that arise in on-line analytical processing (OLAP). Cohen in [6] provides a sufficient condition for equivalence of (generalizations of) combined-semantics CQ queries, as well as sufficient and necessary equivalence conditions for several proper sublanguages of the query language of [6]. To the best of our knowledge, no results on minimization of CQ queries beyond set-semantics queries have been reported in the literature.

Our goal in this paper is to continue the study of equivalence and minimization of CQ queries. We consider the problems of (i) finding minimized versions of combined-semantics CQ queries, and of (ii) determining whether two CQ queries are combined-semantics equivalent. We continue the tradition of [2, 4, 6] of studying these problems using the tool of containment between queries. We extend the containment, equivalence, and minimization results of [2] to general combined-semantics CQ queries, and show the limitations of each extension. We show that the minimization approach of [2] can be extended to general CQ queries without limitations. We also propose a necessary and sufficient condition for equivalence of queries belonging to a large natural sublanguage of combined-semantics CQ queries; this sublanguage encompasses (but is not limited to) all set, bag, and bag-set queries. Our equivalence and minimization results, as well as our general sufficient condition for containment of combined-semantics CQ queries, reduce correctly to the special cases reported in [4] for bag and bag-set semantics. Our containment and equivalence conditions also properly generalize the results of [6], provided the latter are restricted to the language of (combined-semantics) CQ queries.

1. INTRODUCTION

Query containment and equivalence are recognized as fundamental problems in database query evaluation and optimization. The reason is, for conjunctive queries *(CQ queries)* — a broad class of frequently used queries, whose expressive power is sufficient to express select-project-join queries in relational algebra — query equivalence can be used as a tool in query optimization. Specifically, to find a more efficient *and* answer-preserving formulation of a given CQ query, it is enough to "try all ways" of arriving at a "shorter" query formulation, by removing query subgoals, in a process called query minimization [2]. A subgoal-removal step succeeds only if equivalence (via containment) of the "original" and "shorter" query formulations can be ensured. The equivalence test of [2] for CQ queries is NP complete, whereas equivalence of general relational queries is undecidable.

The query-minimization algorithm of [2] works under the assumption of *set semantics* for query evaluation, where both the database (stored) relations and query answers are treated as sets. Query answering and reformulation in the set-semantics setting have been studied extensively in the database-theory literature. As a basis, these studies have all used the necessary and sufficient containment condition of [2] for CQ queries. At the same time, the set semantics is not the default query-evaluation semantics in database systems in practice. For instance, in the standard relational query language SQL, duplicates are removed from the answer to a SQL query *only* if the query uses the `DISTINCT` keyword in its `SELECT` clause. This and other discrepancies between the set semantics for query evaluation and the standard of the query language SQL have prompted researchers [4, 9] to consider "bag semantics" and "bag-set semantics" for query evaluation. Under *bag semantics,* both query answers and stored relations are treated as *bags* (that is, *multisets).* Under *bag-set semantics,* query answers are treated as bags, whereas the database relations are assumed to be sets.

In an extended abstract [4] published in PODS in 1993, Chaudhuri and Vardi focused on the hard problem of bag containment for CQ queries. The paper [4] formulates containment and equivalence results, including equivalence tests, for bag and bag-set queries. However, the full version of the paper [4] has never appeared, and the problems of bag and bag-set containment for CQ queries remain open to this day.

The seminal work by Cohen [5, 6] has provided a Datalog-based formalism for treating queries evaluated under each of set, bag, and bag-set semantics uniformly as special cases of the more general *combined semantics*. To show the practical value of the combined-semantics formalism, Cohen exhibited in [6] a number of real-life SQL queries that can be expressed as combined-semantics CQ queries, but cannot be expressed using any of the set, bag, or bag-set semantics alone. In the following example we show another realistic combined-semantics query, and use it to illustrate issues in equivalence and minimization of combined-semantics queries.

EXAMPLE 1.1. *The application domain used here is based on a data-warehousing example from [8]. Consider a retailer that has multiple stores. The retailer carries many items and has an elaborate relational database/warehouse for analysis, marketing, and promotion purposes. One of the tables in the database has the schema* `pos(transactionID,itemID,storeID,date,amount)`*; the table has one million rows. This table represents point-of-sale transactions, with one tuple for every item sold in a transaction. Each tuple has the transaction ID, the ID of the item sold, the ID of the store selling it, the date, and the amount of the sale.*

Suppose that the business-development division of this store chain would like to study the impact, on the total sales, of those transactions in the stores where the item prices are the same as on some fixed date.[1] *We denote this fixed date of interest by the constant* `d1`*. Consider SQL query* `Q1` *that could be used for the purpose of this analysis.*

```
(Q1) SELECT storeID, amount FROM pos P WHERE EXISTS
  (SELECT * FROM pos WHERE itemID = P.itemID AND
  storeID = P.storeID AND amount = P.amount AND date = ‘d1’)
```

For each store ID, query `Q1` *returns separately the amount for each transaction that took place in that store on the date* `d1`*. Moreover, for each item that was sold in the store on the date* `d1`*,* `Q1` *returns all the* same *purchase amounts for the same item in the same store, as many times as the purchases have happened, regardless of the date. For instance, suppose that the* `pos` *relation has the information that the store with ID* `s1` *(i) sold item* `i1` *three times on the date* `d1` *with the transaction amount* `amt1`*, (ii) sold no other items on or before that date, and (iii) has sold* `i1` *with the same transaction amount four times after the date* `d1`*. Then the query* `Q1` *on that relation would return the pair* `(s1, amt1)` *seven times. If the analysts want to calculate correctly the total per-store returns for all the transactions that have the same item prices as the same-store transactions for the date* `d1`*, then all they have to do to write the query is to (i) add to the query* `Q1` *the clause* `GROUP BY storeID`*, and to (ii) replace* `amount` *by* `sum(amount)` *in the* `SELECT` *clause of* `Q1`*.*

Due to the large size of the relation `pos`*, the self-join of* `pos` *in the query* `Q1`*, via a correlated subquery, should be avoided if at all possible. Let us see whether the following query* `Q2` *could be used instead of* `Q1`*.*

[1]We assume that the transaction amount can be used to determine the price of the item. This is true, for instance, for sales of big-ticket items, where each transaction typically records the sale of one such item. We also assume that item prices do not change in the middle of a business day.

```
(Q2) SELECT DISTINCT storeID, amount FROM pos
     WHERE date = ‘d1’
```

Suppose for the moment that we modify the query `Q1` *by adding the* `DISTINCT` *keyword to its* `SELECT` *clause; we call the resulting query* `Q3`*. We could then use the classic results of [2] to prove that the queries* `Q2` *and* `Q3` *return the same answers on all databases; we can use [2] because* `Q2` *and* `Q3` *are set-semantics queries.*[2] *Using the definitions of [5, 6], we can say that the two queries are* combined-semantics equivalent*. Because it can be shown formally [2] that the equivalence of* `Q2` *to* `Q3` *holds, we can reduce significantly the costs of evaluating the query* `Q3`*, by evaluating* `Q2` *instead.*

Unfortunately, the query `Q3` *does not work for the purposes of the analysis that we covered earlier in this example. The reason is, the query* `Q3` *returns each* `(storeID, amount)` *pair exactly once, due to the use of the* `DISTINCT` *keyword in the query. As a result, the answer to* `Q3` *(or* `Q2`*) cannot be used in a correct computation of the total returns as discussed above, in the way that* `Q1` *can be used. Note that removing the* `DISTINCT` *keyword from* `Q2` *would not bring us any closer to evaluating* `Q1` *correctly, because the resulting bag-set query would return the information only for those transactions that took place on the date* `d1`*. Even the following bag-set query* `Q4`*, which avoids the problematic correlated subquery of* `Q1`*, cannot be used for a correct evaluation of the query* `Q1`*.*

```
(Q4) SELECT P1.storeID, P1.amount FROM pos P1, pos P2
  WHERE P1.itemID = P2.itemID AND P1.storeID = P2.storeID
  AND P1.amount = P2.amount AND P2.date = ‘d1’
```

The problem with `Q4` *is that for each distinct row in* `pos P1` *such that the row produces a pair* `(storeID, amount)` *qualifying for the answer to* `Q4`*, the query* `Q4` *returns the pair not once (which would be correct, see* `Q1`*), but as many times as the item was sold on the date* `d1`*. Thus, we can show that* `Q4` *is not combined-semantics equivalent to* `Q1`*. Adding the* `DISTINCT` *keyword to* `Q4` *does not fix the problem, as it would give us a set-semantics query, which is combined-semantics equivalent to the queries* `Q2` *and* `Q3`*, but not to* `Q1`*.* □

We can use the results of [2, 6] to show that the queries `Q2` and `Q3` of Example 1.1 are combined-semantics equivalent. Specifically, we can determine formally that the evaluation costs of the query `Q3` can be minimized by evaluating `Q2` instead, because `Q2` is a *minimized* version of both `Q2` and `Q3`. As for the combined-semantics CQ queries illustrated here by the query `Q1` of Example 1.1, such queries (with grouping and aggregation added) arise naturally in on-line analytical processing *(OLAP)* applications [11, 12]. Such queries occur whenever a data-analysis task calls for a query structure with nested subqueries. Such queries also arise due to joins that go beyond “star-schema joins” [3], which are the only well-understood joins in the literature on OLAP query optimization. See [11, 12, 13, 14] for more detailed discussions of why queries with nested subqueries and with “non-star joins” are natural and frequent in OLAP. (For an additional extended illustration of such queries, see Example C.1 in Appendix C.) We can use the results of [5, 6] to show that the query `Q1` of Example 1.1 cannot be represented equivalently as a SQL query without subqueries, that is as a CQ set, bag, or bag-set query. At the same time, to the best of

[2]See Appendix B for the Datalog syntax of [6] for all the queries of this example.

our knowledge, past work cannot help us determine the most efficient equivalent SQL representation of the query `Q1`. Interestingly, it follows from our results in this current paper that the query `Q1` cannot be simplified further by equivalent reformulation, specifically by minimization.

Our contributions.
In this paper we study equivalence and minimization of unaggregated SQL queries with equality comparisons and possibly with subqueries. We follow the approach of [6], where the study concentrates on Datalog translations of such queries, that is on combined-semantics CQ queries. The requisite translations from SQL to Datalog are straightforward ("as expected").[3] In the remainder of this paper, all queries are expressed using the Datalog-based formalism of [6].

We focus on the problems of (i) finding minimized versions of combined-semantics CQ queries, and of (ii) determining whether two CQ queries are combined-semantics equivalent. We continue the tradition of [2, 4, 6] of studying these problems using the tool of containment between queries. All the results in this paper hold for queries that may have constants. Our specific contributions are as follows:

- For combined-semantics containment of CQ queries, in Section 3 we introduce two necessary conditions and a sufficient condition. The latter result properly generalizes both (i) the sufficient condition outlined in [4] for bag containment of CQ queries, and (ii) the general sufficient containment condition that can be obtained from [6] for CQ queries. To formulate our sufficient condition, we introduce covering mappings *(CVMs)* between CQ queries. We use CVMs in our results throughout the remainder of the paper.

- In Section 4 we present a necessary condition for CQ-query equivalence. To formulate this condition, we isolate a large class of CQ queries, which we call "explicit-wave queries".[4] We show that this class of queries encompasses, but is not limited to, (i) all CQ set, bag, and bag-set queries, and (ii) all CQ queries for which [6] provides its sufficient and necessary equivalence tests. We refer to all combined-semantics CQ queries that are not explicit-wave queries as "implicit-wave queries." Our necessary condition for query equivalence is asymmetric – it states that if for CQ queries Q and Q' we have the combined-semantics equivalence $Q \equiv_C Q'$, *and* Q is an explicit-wave query, then there exists a CVM from Q' to Q. We discuss why establishing this result is not trivial.

- In Section 5 we propose a sound and complete algorithm for minimizing combined-semantics CQ queries. We also show that for all CQ queries, including all implicit-wave queries, the minimized version of the query exists and is unique up to an isomorphism CVM.

- Finally, in Section 6 we study our proposed conditions for equivalence of CQ queries. Our main focus is on reformulating these conditions using minimized versions of the queries. The reformulations tie our equivalence conditions together with the results of [2, 4]. Our sufficient and necessary condition for equivalence of explicit-wave CQ queries is strictly more powerful than each of the equivalence tests of [6], provided that the latter are applied to CQ queries only.

[3]Section 1 in [6] provides some details of the translations.
[4]The term "explicit-wave query" is due to the structures generated by the proof of the main result of Section 4.

We present in the main text (the first 12 pages) only an extended abstract of our results. All the details are available in the appendices.

The results of this paper can be used directly in query optimizers for database-management systems, as well as for developing minimization methods for queries in more expressive languages than CQ queries and in presence of integrity constraints. Our results can also be used for developing algorithms for rewriting queries using views and for view selection under combined semantics.

2. PRELIMINARIES

2.1 Combined semantics: The framework [6]

2.1.1 *Syntax of queries*

Predicate symbols are denoted as p, q, r. Databases contain ground atoms for a given set of predicate symbols; we consider finite-size databases only. A database may have several copies of the same atom. To denote this fact, each atom in the database is associated with a *copy number* N. Formally, if p is an n-ary predicate, for an $n \in \mathbb{N}_+$ (with $\mathbb{N}_+$ the set of natural numbers), we write $p(c_1, \ldots, c_n; N)$, with $N \in \mathbb{N}_+$, to denote that there are precisely N copies of $p(c_1, \ldots, c_n)$ in the database. As a shorthand, if $N = 1$, we often omit the copy number N. The *active domain of database* D, denoted *adom*(D), is the set of all constants mentioned in the ground atoms of D. We adopt a convention by which, for each atom of the form $p(c_1, \ldots, c_n)$ such that database D has $N \geq 1$ copies of that atom, N is an element of *adom*(D) only if there exists in D an atom $r(c'_1, \ldots, c'_m)$ (where r and p may or may not be the same predicate) such that N is one of c'_1, …, c'_m.

For query syntax, we denote variables using X, Y, Z, possibly with subscripts, and i, j, k. The former range over constants in the database (i.e., over *adom*(D)), whereas the latter range over copy numbers. For this reason, we call the former *regular variables* (or simply *variables* for short) and we call the latter *copy variables.* We use c, d to denote constants. A *term,* denoted as S, T, is a variable or a constant.

A *relational atom* has the form $p(S_1, \ldots, S_n)$, where p is a predicate of arity n. We also use the notation $p(\bar{S})$, where $\bar{S}$ stands for a sequence of terms $S_1, \ldots, S_n$. A *copy-sensitive atom* has the form $p(\bar{S}; i)$, and is simply a relational atom with copy variable i. We call relational atom $p(\bar{S})$ *the relational template of copy-sensitive atom* $p(\bar{S}; i)$. For each relational atom, its relational template is the atom itself. A *condition,* denoted as L, is a conjunction of relational and copy-sensitive atoms, with duplicate atoms allowed, such that all copy variables in L are unique (i.e., appear in a single copy-sensitive atom, and do not appear in other atoms). Sometimes it will be convenient for us to view condition L as a bag of all and only the elements in the conjunction L.

We distinguish between variables that appear in the head of a query and those that only appear in the body. The former are *distinguished (head) variables,* and the latter are *nondistinguished (nonhead) variables.* Nondistinguished variables come in two flavors: *set variables* and *multiset variables.* The intuition for the difference between these two types of variables is as follows. When evaluating a query, different assignments for set variables do not contribute to the multiplicity in which a particular answer is returned by the query. On the other hand, different assignments for multiset variables do contribute to the multiplicity of the returned answers. Technically, in order to differentiate between set variables and multiset variables, we always specify the set of multiset variables in each condition immediately to the right of the condition. As a syntactic requirement, all copy variables must be in the set of multiset variables.

Definition 2.1. *(***Query syntax: CCQ query***) A copy-sensitive conjunctive query (CCQ query) is a nonrecursive expression of the form*

$$Q(\bar{X}) \leftarrow L, M,$$

where $\bar{X}$ contains at least one term, L is a nonempty condition, and M is a set of variables, such that:

- *L contains all the variables in $\bar{X}$; that is, Q is* safe*;*
- *M is a subset of the set of nondistinguished variables of L and contains all copy variables of L. We denote all the copy variables of Q collectively as $M_{copy} \subseteq M$, and all the remaining ("multiset noncopy") variables in M as $M_{noncopy} := M - M_{copy}$.* $\square$

We call each element of the condition L a *subgoal* of Q. The variables in M are the *multiset variables* of Q. The variables in L that are not in $\bar{X}$ or in M are the *set variables* of Q. We use $S(Q)$ to denote an arbitrary vector, without repetitions, of the set variables of Q, and $\bar{S}(Q)$ to denote an arbitrary vector, without repetitions, of the remaining variables of Q (i.e., the distinguished and multiset variables of Q). By abuse of notation, we will often refer to a query by its head $Q(\bar{X})$ or simply by its head predicate Q. For a vector of terms $\bar{X}$ with $k \geq 1$ elements, we say that a CCQ query with head $Q(\bar{X})$ is a *CCQ k-ary query.*

We will sometimes be interested in special types of queries. A CCQ query Q is a *set query* if it has no multiset variables, that is, if $M = \emptyset$. Query Q is a *multiset query* if Q has no set variables. Further, a multiset query Q is (i) a *bag query* if Q has only copy-sensitive subgoals, and is (ii) a *bag-set query* if Q has only relational subgoals.

2.1.2 Combined semantics for queries

We define how CCQ query $Q(\bar{X}) \leftarrow L, M$ yields a *multiset* of tuples on database D. Intuitively, we start by considering satisfying assignments of the condition L. We then restrict these assignments to the *nonset variables* of L, that is to $\bar{S}(Q)$. Each of these restricted assignments yields a tuple in the result. A formal description of the semantics follows.

Let γ be a mapping of the terms in condition L to values. We will also apply γ to a sequence of terms to derive a sequence of values, in the obvious way. We say that γ is a *satisfying assignment* of L with respect to database D if all of the following conditions hold:

- γ is the identity mapping on constants;
- for all relational atoms $p(\bar{T}) \in L$, there exists an $N \in \mathbb{N}_+$ such that we have $p(\gamma\bar{T}; N) \in D$; and
- for all copy-sensitive atoms $p(\bar{T}; i) \in L$, the following two conditions hold:
  - $\gamma i \in \mathbb{N}_+$ (i.e, γi is a positive natural number);
  - there is an $N \geq \gamma i$ such that $p(\gamma\bar{T}; N) \in D$.

Remark (2.6 in [6]) It may be helpful to view a database atom of the form $p(\bar{S}; N)$ as a shorthand for N copies of the atom $p(\bar{S})$. Then a query atom $p(\bar{T}; i)$ is satisfied by γ if $\gamma\bar{T} = \bar{S}$ and γi is one of the numbers $1, 2, \ldots, N$.

Let $\Gamma(Q, D)$ denote the set of satisfying assignments of L with respect to database D. Let γ be an assignment of the variables in $\bar{S}(Q)$ to constants. We say that γ is *satisfiably extendible* if there is an assignment $\gamma' \in \Gamma(Q, D)$ such that γ and γ' coincide on all terms for which γ is defined, that is, $\gamma'(X) = \gamma(X)$ for all $X \in \bar{S}(Q)$. Intuitively, this means that it is possible to extend γ to derive a satisfying assignment of L. We use $\Gamma_{\bar{S}}(Q, D)$ to denote the set of satisfiably extendible assignments of $\bar{S}(Q)$ with respect to D. For the $\gamma \in \Gamma_{\bar{S}}(Q, D)$ and for the $\gamma' \in \Gamma(Q, D)$ as specified in this paragraph, we say that γ' *contributes* γ *to* $\Gamma_{\bar{S}}(Q, D)$.

Sometimes it will be convenient to treat $\Gamma(Q, D)$ and $\Gamma_{\bar{S}}(Q, D)$ as relations. That is, $\bar{S}$ is the sequence of attributes in the schema of $\Gamma_{\bar{S}}(Q, D)$, and each assignment γ in $\Gamma_{\bar{S}}(Q, D)$ is the tuple $\gamma(\bar{S})$; similarly for $\Gamma(Q, D)$. (Recall that by definition, each γ is the identity mapping on constants.)

We now define the result of applying a query Q to a database D. (We use $\{\!\{\ldots\}\!\}$ to denote a bag of values.)

Definition 2.2. *(***Combined semantics***) Let $Q(\bar{T}) \leftarrow L, M$ be a CCQ query and let D be a database. The* result of applying Q to D under combined semantics, *denoted $Res_C(Q, D)$, is defined as*

$$Res_C(Q, D) \;:=\; \{\!\{\ \gamma(\bar{T}) \mid \gamma \in \Gamma_{\bar{S}}(Q, D)\ \}\!\}\ . \qquad \square$$

Note that $Res_C(Q, D)$ is a bag of tuples, that is, $Res_C(Q, D)$ may contain multiple occurrences of the same tuple.

Under certain circumstances, combined semantics coincides with set, bag, or bag-set semantics. Please see Appendix D for the details on the three traditional query semantics, specifically on how these semantics can be formulated as special cases of combined semantics.

2.1.3 Query containment and equivalence

Query containment under combined, set, bag, and bag-set semantics is defined in the standard manner. Formally, Q is contained in Q' under a given semantics if, for all databases, the bag of values returned by Q is a subbag of the bag of values returned by Q'. We write $Q \sqsubseteq_C Q'$, $Q \sqsubseteq_S Q'$, $Q \sqsubseteq_B Q'$, and $Q \sqsubseteq_{BS} Q'$ if Q is contained in Q' under combined, set, bag, and bag-set semantics, respectively. Similarly, we use $Q \equiv_C Q'$, $Q \equiv_S Q'$, $Q \equiv_B Q'$, and $Q \equiv_{BS} Q'$ to denote the fact that Q is equivalent to Q' under each semantics. $Q \equiv_C Q'$ holds if and only if $Q \sqsubseteq_C Q'$ and $Q' \sqsubseteq_C Q$ both hold. The definitions of $Q \equiv_S Q'$, $Q \equiv_B Q'$, and $Q \equiv_{BS} Q'$ parallel that of $Q \equiv_C Q'$ in the obvious manner.

For CCQ queries Q and Q', we have that (1) $Q \sqsubseteq_S Q'$ iff $Q \sqsubseteq_C Q'$, in case Q and Q' are set queries; (2) $Q \sqsubseteq_B Q'$ iff $Q \sqsubseteq_C Q'$, in case Q and Q' are bag queries; and (3) $Q \sqsubseteq_{BS} Q'$ iff $Q \sqsubseteq_C Q'$, in case Q and Q' are bag-set queries. (See Proposition D.1 in Appendix D.)

For a class $\mathcal{Q}$ of queries: The *$\mathcal{Q}$-containment problem for combined semantics* is: *Given queries Q and Q' in $\mathcal{Q}$, determine whether $Q \sqsubseteq_C Q'$.* The *$\mathcal{Q}$-equivalence problem* is defined similarly using $\equiv_C$ instead of $\sqsubseteq_C$. The two problems can be defined similarly for other semantics.

2.2 Equivalence and minimization results

Set queries. Given two conjunctions $\phi(\bar{U})$ and $\psi(\bar{V})$ of relational atoms, a *homomorphism* from $\phi(\bar{U})$ to $\psi(\bar{V})$ is a mapping h from the set of terms in $\bar{U}$ to the set of terms in $\bar{V}$ such that (1) $h(c) = c$ for each constant c, and (2) for each atom $r(U_1, \ldots, U_n)$ of ϕ, we have that $r(h(U_1), \ldots, h(U_n))$ is in ψ. Given two CCQ k-ary set queries $Q_1(\bar{X}) \leftarrow \phi(\bar{X}, \bar{Y}), \{\}$ and $Q_2(\bar{X}') \leftarrow \psi(\bar{X}', \bar{Y}'), \{\}$, a *containment mapping* from Q_1 to Q_2 is a homomorphism h from $\phi(\bar{X}, \bar{Y})$ to $\psi(\bar{X}', \bar{Y}')$ such that $h(\bar{X}) = \bar{X}'$.

Theorem 2.1. *[2] Given two CCQ set queries Q_1 and Q_2 of the same arity, $Q_1 \sqsubseteq_S Q_2$ holds if and only if there is a containment mapping from Q_2 to Q_1.* □

This classic result of [2] forms the basis for a sound and complete test for set-equivalence of CCQ set queries Q and Q', by definition of set-equivalence $Q \equiv_S Q'$.

We now introduce the notion of a "reduced-condition query" for CCQ query. Given a CCQ query $Q(\bar{X}) \leftarrow L, M$, a CCQ query $Q'(\bar{X}) \leftarrow L', M'$ is a *(proper) reduced-condition query for Q* if (i) L' is a (proper) subbag of L, (ii) the set M' is the set of all elements of M that occur in L', and (iii) Q' is a safe query (i.e., L' contains all the variables in $\bar{X}$).

Definition 2.3. *(***Minimized CCQ query; minimized version of CCQ query***) A CCQ query Q is a* minimized CCQ query *if for each subgoal s of Q, the removal of s from the condition of Q results in a query Q' such that $Q \not\equiv_C Q'$. A CCQ query Q is a* minimized version of CCQ query Q' *if (1) Q is a reduced-condition query for Q', (2) Q is a minimized query, and (3) $Q \equiv_C Q'$.* □

Theorem 2.2. *[2] Given two CCQ set queries Q_1 and Q_2 of the same arity:*

(1) The minimized version of Q_1 exists and is unique up to isomorphism; and

(2) $Q_1 \equiv_S Q_2$ holds if and only if the minimized versions of Q_1 and of Q_2 are isomorphic. □

Bag and bag-set queries. For bag and bag-set semantics, the following conditions are known for CCQ query equivalence. (Query Q_c is a *canonical representation* of query Q if Q_c is the result of removing all duplicate atoms from the condition of Q.)

Theorem 2.3. *[4] Let Q and Q' be CCQ queries. Then (1) When Q and Q' are bag queries, $Q \equiv_B Q'$ iff Q and Q' are isomorphic. (2) When Q and Q' are bag-set queries, $Q \equiv_{BS} Q'$ iff Q_c and Q'_c are isomorphic.* □

Note 1. Assuming that the syntax of Section 2.1 is used for queries, we can show that each isomorphism mapping in the scope of Theorem 2.3 is a homomorphism (which also maps copy variables to copy variables) that maps each head (nonhead, respectively) variable of one query to a distinct head (nonhead, respectively) variable of the other query.

Combined-semantics queries. The next result is a sufficient condition of [6] for equivalence of two queries under combined semantics. In [6], Cohen formulates each of Definition 2.4 and Theorem 2.4 for CCQ queries that may also contain negation and inequality comparisons. (In condition (3) of Definition 2.4 we treat the query conditions, which are conjunctions of atoms, as bags of the same atoms. Given a bag B, we call a set S the *core-set* of B if S is the result of dropping all duplicates of all elements of B.)

Definition 2.4. *(***Multiset-homomorphism [6]***) Let $Q(\bar{X}) \leftarrow L, M$ and $Q'(\bar{X}') \leftarrow L', M'$ be two k-ary CCQ queries, for $k \geq 1$. Let φ be a mapping from the terms of Q' to the terms of Q.*[5] *We say that φ is a* multiset-homomorphism from Q' to Q *if φ satisfies all of the following conditions:*

1. *$\varphi \bar{X}' = \bar{X}$;*
2. *φ is the identity mapping on constants;*
3. *the core-set of $\varphi L'$ is a subset of the core-set of L ;*
4. *$\varphi M' \subseteq M$; and*
5. *$\varphi Y \neq \varphi Y'$ for every two distinct variables $Y, Y' \in M'$.* □

For every mapping φ that satisfies conditions 1–3 of Definition 2.4, we call φ a *generalized containment mapping (GCM).*

We say that two CCQ queries Q and Q' are *multiset homomorphic* whenever there is a multiset-homomorphism from Q to Q' and another from Q' to Q.

Theorem 2.4. *[6] Given CCQ queries Q and Q'. If Q and Q' are multiset homomorphic then $Q \equiv_C Q'$.* □

Note 2. Theorem 2.4 is proved in [6] via showing that for two (generalized) CCQ queries Q and Q', the existence of a multiset-homomorphism from Q' to Q implies $Q \sqsubseteq_C Q'$.

The condition of Theorem 2.4 is not necessary for the query classes considered in [6].

3. CONTAINMENT AND MAPPINGS

In this section, for combined-semantics containment of CCQ queries, we introduce two necessary conditions, Theorems 3.1 and 3.2, and a sufficient condition, Theorem 3.3. The latter result properly generalizes both (i) the sufficient condition outlined in [4] for bag containment of CQ queries, and (ii) the general sufficient containment condition for CCQ queries that can be obtained from [6]. To formulate Theorem 3.3, we introduce covering mappings (CVMs) between CCQ queries. We use CVMs in our results throughout the remainder of this paper.

Throughout this paper, we use the notation $Q(\bar{X}) \leftarrow L, M$ and $Q'(\bar{X}') \leftarrow L', M'$ for the definitions of CCQ queries Q and Q'. The conditions of Q and Q' may have constants.

3.1 Necessary conditions for containment

We introduce two necessary conditions for a CCQ query Q being combined-semantics contained in CCQ query Q', Theorems 3.1 and 3.2.

Theorem 3.1. *Let Q and Q' be two k-ary CCQ queries. Then $Q \sqsubseteq_C Q'$ implies both $|M_{copy}| \leq |M'_{copy}|$ and $|M_{noncopy}| \leq |M'_{noncopy}|$.* □

(For a set S we denote by $|S|$ the cardinality of S.) The proof of Theorem 3.1 can be found in Appendix E. The idea of the proof is that we use the definition of the query Q to construct a special database D. Some answer to Q on D has a multiplicity (in the bag $Res_C(Q, D)$) that is proportional

[5] We also apply φ to atoms and conjunctions of atoms, in the obvious way, e.g., $\varphi(p(\bar{S})) = p(\varphi(\bar{S}))$.

to $|adom(D)|^{|M|}$. Then we can use several versions of this database D, to prove Theorem 3.1 by contradiction: We assume either $|M_{copy}| > |M'_{copy}|$ or $|M_{noncopy}| > |M'_{noncopy}|$, and obtain as a result that $Q \sqsubseteq_C Q'$ cannot hold. The challenge in the proof is in the combination of allowing constants in the condition of Q and of arriving at "the right" multiplicity of the answer to Q on D when we are to show that $|M_{copy}| \leq |M'_{copy}|$ must hold whenever $Q \sqsubseteq_C Q'$.

We call a pair (Q, Q') of CCQ queries a *containment-compatible CCQ pair* if (i) The (positive) head arities of Q and Q' are the same; (ii) $|M_{copy}| \leq |M'_{copy}|$; and (iii) $|M_{noncopy}| \leq |M'_{noncopy}|$. (Note the asymmetry in the notation for the pair.) Further, we call a pair (Q, Q') of CCQ queries an *equivalence-compatible CCQ pair* if each of (Q, Q') and (Q', Q) is a containment-compatible CCQ pair. By Theorem 3.1 we have that, whenever $Q \sqsubseteq_C Q'$ ($Q \equiv_C Q'$, respectively) holds, then (Q, Q') is a containment-compatible (an equivalence-compatible, respectively) CCQ pair.

We now generalize the "only-if" part of the classic result of [2], see Theorem 2.1 in Section 2.2, to CCQ queries. For the definition of generalized containment mapping, *GCM,* see Section 2.2. We begin by introducing another definition that we need to formulate our generalization, Theorem 3.2.

For a CCQ query Q, we say that CCQ query Q_{ce} is a *copy-enhanced version of* Q if Q_{ce} is the result of adding a distinct copy variable to each relational subgoal of Q. (We can show that for a query Q, all copy-enhanced versions of Q are identical up to renaming of the copy variables introduced in the construction of Q_{ce}.) Further, for each CCQ query Q' that is a reduced-condition query for CCQ query Q, we obtain the query Q'_{ce} by removing from Q_{ce} those subgoals that do not correspond to the subgoals of Q'. (A formalization is omitted due to lack of space; see Appendix F.)

We are now ready to formulate Theorem 3.2.

THEOREM 3.2. *Given CCQ queries Q and Q' such that $Q \sqsubseteq_C Q'$. Then there exists a GCM from Q'_{ce} to Q_{ce}.* □

The proof of Theorem 3.2 is a straightforward generalization of the proof, via canonical databases, of the result of [2]. Appendix G has the proof and an illustration.

Neither Theorem 3.1 not Theorem 3.2 provides a sufficient condition for combined-semantics containment of two CCQ queries: The following Example 3.1 is a counterexample in both cases.

EXAMPLE 3.1. *Consider CCQ queries Q and Q':*

$Q(X) \leftarrow p(X, Y), p(Y, Z), p(Z, X; i), \{Y, i\}$.
$Q'(X) \leftarrow p(X, Y), p(Y, Z), p(Z, X; i), \{Z, i\}$.

Apart from the choice of multiset variables, Q and Q' are clearly isomorphic. However, $Q \equiv_C Q'$ does not hold, as witnessed by database $D = \{p(1, 2),\ p(2, 3),\ p(3, 1),\ p(1, 4), p(4, 3)\}$. Our results in this paper permit us to determine $Q \not\equiv_C Q'$ syntactically, see Section 4. (To the best of our knowledge, no previous work provides a formal procedure to determine $Q \not\equiv_C Q'$ for queries such as in this example.) □

Each of Theorem 3.1 and Theorem 3.2 yields a necessary condition for combined-semantics *equivalence* of CCQ queries in a natural way. For instance:

COROLLARY 3.1. *Let Q and Q' be two k-ary CCQ queries such that $Q \equiv_C Q'$. Then we have $|M_{copy}| = |M'_{copy}|$ and $|M_{noncopy}| = |M'_{noncopy}|$.* □

3.2 Covering mappings for CCQ queries

In this subsection, we define covering mappings *(CVMs)* between CCQ queries, and study properties of CVMs. Appendix H has proofs of all the results of this subsection.

DEFINITION 3.1. *(***Covering mapping (CVM)***) Given CCQ queries Q and Q', a mapping, call it μ, from the terms of Q' to the terms of Q is called a* covering mapping (CVM) from Q' to Q *whenever μ satisfies all of the following conditions:*

(1) μ maps each constant (if any) in Q' to itself;
(2) applying μ to the vector $\bar{X}'$ yields the vector $\bar{X}$;
(3) the set of terms in $\mu M'_{copy}$ is exactly M_{copy}, and the set of terms in $\mu M'_{noncopy}$ includes all of $M_{noncopy}$;
(4) for each relational subgoal of Q', of the form $s(\bar{Y})$, there exists in Q either a relational subgoal $s(\mu(\bar{Y}))$, or a copy-sensitive subgoal $s(\mu(\bar{Y}); i)$, with $i \in M_{copy}$; and
(5) for each copy-sensitive subgoal of Q' of the form $s(\bar{Y}; i)$, there exists in Q a subgoal $s(\mu(\bar{Y}); \mu(i))$. □

By Definition 3.1, if there exists a CVM from CCQ query Q' to CCQ query Q, then (Q, Q') is a containment-compatible CCQ pair. It is immediate from Definition 3.1 that if a mapping μ is a CVM from Q' to Q, then μ induces a surjection from the set of copy-sensitive subgoals of Q' to the set of copy-sensitive subgoals of Q. Observe also that in case both Q and Q' are set queries, Definition 3.1 becomes the definition of containment mapping [2] from Q' to Q.

For the special case where (Q, Q') is an *equivalence*-compatible CCQ pair, we call each CVM from Q' to Q a *same-scale covering mapping (SCVM) from Q' to Q.* By definition, each SCVM from Q' to Q is a bijection from the set M' to the set M when restricted to the domain M'.

The intuition for Definition 3.1 comes from our use of CVMs later in this paper (Section 5) as a tool for minimizing CCQ queries. Consider the following illustration.

EXAMPLE 3.2. *Let queries Q and Q' be as follows.*

$Q(X) \leftarrow p(X, X, Y; i), p(X, Z, Y), \{Y, i\}$.
$Q'(X) \leftarrow p(X, X, Y; i), \{Y, i\}$.

By Definition 2.4, there does not exist a multiset homomorphism [6], or even a GCM, from Q to Q'. At the same time, by our results of Section 5, Q' is a minimized version of Q. We can ascertain this fact by using a CVM, μ, from Q to Q': $\mu = \{\ X \rightarrow X,\ Y \rightarrow Y,\ i \rightarrow i,\ Z \rightarrow X\ \}$. □

As illustrated by Example 3.2, CVMs are not GCMs. Indeed, the definition of CVMs gives up explicitly on condition (3) for GCMs (see Definition 2.4); by this condition, for each subgoal s of Q in Example 3.2, we must have that $\mu(s)$ is a subgoal of Q'. While CVMs are not GCMs, a nice relationship exists between CVMs and GCMs, see Proposition 3.2. To formulate Proposition 3.2, we use the following definition, in which we treat query conditions as bags of atoms.

Given CCQ query Q, let $\mathcal{T}(Q)$ be the set of relational templates of all (if any) copy-sensitive subgoals of Q. We recall that CCQ query Q_c is a canonical representation of CCQ query Q if Q_c is the result of removing all duplicate atoms from the condition of Q.

Definition 3.2. *(**(Un)regularizing CCQ query**) Given CCQ query Q, with canonical representation Q_c. Then (1) A* regularized version *of Q is a CCQ query Q_r obtained by dropping from the condition of Q_c all elements of the set $\mathcal{T}(Q)$; (2) A* deregularized version *of Q is a CCQ query Q_d obtained by adding to the condition of Q_r all elements of the set $\mathcal{T}(Q)$; (3) An* unregularized version *of Q is a CCQ query Q_u obtained by adding to the condition of Q_r one or more duplicates of the existing relational subgoals, and/or one or more elements (possibly with duplicates) of the set $\mathcal{T}(Q)$.* □

The following result is straightforward.

Proposition 3.1. *Given a CCQ query Q. Then (1) Each of Q_r and Q_d is a well-defined, unique and polynomial-time computable CCQ query; (2) $Q_r \equiv_C Q$ and $Q_d \equiv_C Q$ both hold; and (3) For each unregularized version Q_u of Q, we have that $Q_u \equiv_C Q$ holds.* □

Appendix H.1 has an illustration and a discussion of the query versions as specified in Definition 3.2.

We are now ready to formulate Proposition 3.2.

Proposition 3.2. *Given CCQ queries Q and Q'. Then for each CVM, μ, from Q to Q', we have that (1) μ is a* GCM *from Q to the deregularized version of Q', and (2) μ is a CVM from Q to the regularized version of Q'.* □

In Example 3.2, we are given the regularized version Q'_r of the query Q'. The deregularized version of Q' is $Q'_d(X) \leftarrow p(X, X, Y; i), p(X, X, Y), \{Y, i\}$. The mapping μ of Example 3.2 (i) is a GCM from Q to Q'_d, (ii) is a CVM from Q to Q'_r, and (iii) is not a GCM from Q to Q'_r.

It turns out that CVMs furnish a rather general sufficient condition for CCQ combined-semantics containment:

Theorem 3.3. *Given CCQ queries Q and Q', such that there exists a CVM from Q' to Q. Then $Q \sqsubseteq_C Q'$ holds.* □

Theorem 3.3 generalizes properly both (i) the sufficient condition of [2] for containment between CCQ set queries, see Theorem 2.1, and (ii) the well-known result of [4] stating that a containment mapping[6] from CCQ bag query Q' *onto* CCQ bag query Q ensures containment $Q \sqsubseteq_B Q'$. In fact, to the best of our knowledge, the proof of our Theorem 3.3 is the first formal proof of the latter result from [4].

We can further relax Definition 3.1, by allowing a (generalization of) CVM to map the set M'_{copy} into a *superset* of M_{copy}. This relaxation provides a sufficient condition for combined-semantics containment of CCQ queries; that sufficient condition properly generalizes the condition of Theorem 3.3. Please see Appendix H.3.2 for all the details.

The condition of Theorem 3.3 does not appear to be a necessary condition for containment of CCQ queries. Indeed, a well-known example of [4] (see Appendix I), claims containment $Q \sqsubseteq_C Q'$, but no CVM exists from Q' to Q.

Finally, we compare CVMs with multiset homomorphisms [6], see Definition 2.4. For a fixed pair of CCQ queries Q and Q', with respective sets of multiset variables M and M', each CVM from Q' to Q has range *at least* M when restricted to the domain M', and each multiset homomorphism from Q' to Q has range *at most* M when restricted to the domain M'. Therefore, general CVMs and multiset homomorphisms are incomparable when applied to pairs of CCQ queries. (See Example H.4 in Appendix H.5.) At the same time, we have the following result for *SCVMs* and multiset-homomorphisms. (The proof, which is immediate from Proposition 3.2, can be found in Appendix H.5.)

[6]The "containment mapping" terminology of [4] results from the use in that paper of a syntax for bag queries that does not coincide with the syntax of [6] used in this current paper. See Appendix H.4 for a detailed discussion.

Proposition 3.3. *Given an equivalence-compatible CCQ pair (Q, Q'). Then each SCVM from Q' to Q is a multiset-homomorphism from Q' to* the deregularized version *of Q, and vice versa.* □

For instance, consider the mapping μ of Example 3.2 from the terms of the query Q to the terms of the query Q' of the example. This mapping is a CVM from Q to Q' and is also a multiset-homomorphism from Q to the deregularized version Q'_d of Q', $Q'_d(X) \leftarrow p(X, X, Y; i), p(X, X, Y), \{Y, i\}$. (Observe that there is no GCM from query Q'_d to query Q'.)

As an immediate corollary of Propositions 3.2 and 3.3, we have that for each equivalence-compatible CCQ pair (Q, Q'), the existence of a multiset-homomorphism from Q' to Q implies the existence of a CVM from Q' to Q. From this result and from Example 3.2, we obtain that the restriction of Theorem 2.4 (due to [6]) to CCQ queries does not have quite the same power as the sufficient condition for equivalence of CCQ queries that is immediate from Theorem 3.3. (See Theorem 6.1 for an explicit formulation; by Theorem 6.1, we have $Q \equiv_C Q'$ for the queries of Example 3.2.) In fact, by Example 3.2 we have that our Theorem 3.3 is a proper generalization of the (implicit) query-containment condition of [6], provided that the latter is applied to CCQ queries only; see Note 2 in Section 2.2. (By Definition 2.4 and by Theorem 3.1, the existence of a multiset-homomorphism from CCQ query Q' to CCQ query Q implies $Q \sqsubseteq_C Q'$ only when (Q, Q') is an equivalence-compatible CCQ pair.)

4. EQUIVALENCE: ASYMMETRIC NECESSARY CONDITION

In this section we present a necessary condition for CCQ query equivalence, Theorem 4.1. To formulate Theorem 4.1, we isolate a large well-behaved class of combined-semantics CQ queries, which we call "explicit-wave queries." Theorem 4.1 is asymmetric: It states that if for CCQ queries Q and Q' the combined-semantics equivalence $Q \equiv_C Q'$ holds, *and* we have that Q is an explicit-wave query, then there exists a CVM from Q' to Q. We discuss why establishing this result is not trivial. (Appendix L has the full proof.)

We begin by introducing Definition 4.1. This technical definition is required for the proof of Theorem 4.1 to go through. Given a CCQ query Q, with set $M_{noncopy} \neq \emptyset$ of multiset noncopy variables, we say that a GCM μ from Q to itself is a *noncopy-permuting GCM* if the mapping resulting from restricting the domain of μ to $M_{noncopy}$ is a bijection from $M_{noncopy}$ to itself. For two noncopy-permuting GCMs, μ_1 and μ_2, from Q to itself, we say that μ_1 and μ_2 *agree on $M_{noncopy}$* if μ_1 and μ_2 induce the same mapping from $M_{noncopy}$ to itself. If for CCQ query Q we have $M_{noncopy} = \emptyset$, we say that all GCMs from Q to itself are noncopy-permuting GCMs, and that all pairs of such GCMs agree on $M_{noncopy}$.

In Definition 4.1, for a CCQ query Q and for its copy-enhanced version Q_{ce}, we will call "the original copy-sensitive subgoals of Q" those copy-sensitive atoms that are present in the conditions of both Q and Q_{ce}.

Definition 4.1. (**Explicit-wave CCQ query**) *A CCQ query Q is an* explicit-wave (CCQ) query *if one of the following conditions holds:*

(1) Q has at most one copy-sensitive subgoal; or

(2) For the set $M_{noncopy}$ of multiset noncopy variables of Q, and for each pair (μ_1, μ_2) of noncopy-permuting GCMs from Q_{ce} to itself, such that μ_1 and μ_2 agree *on $M_{noncopy}$, for each original copy-sensitive subgoal, s, of Q we have that $\mu_1(s)$ and $\mu_2(s)$ have the same relational template.* □

The problem of determining whether a given CCQ query is an explicit-wave query can easily be seen to be in co-NP. It is open whether this upper complexity bound is tight.

As an example, any CCQ query Q that has a distinct predicate name for each subgoal (i.e., is a query without self-joins) can be shown to be an explicit-wave query.

For each CCQ query Q that is not explicit-wave, we call Q an *implicit-wave query.* Consider an illustration.

EXAMPLE 4.1. *Consider CCQ queries Q and Q'.*

$$Q(X_1) \leftarrow r(X_1, Y_1, Y_2, X_2; i), r(X_1, Y_1, Y_2, X_3; j), \{Y_1, Y_2, i, j\}.$$
$$Q'(X_1) \leftarrow r(X_1, Y_1, Y_2, X_2; i), r(X_1, Y_1, Y_2, X_2; j), \{Y_1, Y_2, i, j\}.$$

The only difference between the queries is that the two subgoals of the query Q have different set *variables, X_2 and X_3, whereas the two subgoals of Q' have the same set variable X_2. We can show (see Appendix J) that the query Q is an implicit-wave query.*

There exist both a multiset homomorphism and a CVM from the query Q to the query Q'. (Recall that each of the two mappings provides a sufficient condition for $Q' \sqsubseteq_C Q$.) Observe that there is no isomorphism between Q and Q'. The remarkable part is that no multiset homomorphism or CVM exists in the opposite direction, that is from Q' to Q. Yet, $Q \equiv_C Q'$ does hold (see Appendix K). It does not help much that there exists a GCM from Q' to Q. By Theorem 3.2, the existence of a GCM is a necessary, rather than sufficient, condition for the containment $Q \sqsubseteq_C Q'$. (To apply Theorem 3.2, observe that Q and Q_{ce} are identical, as are Q' and Q'_{ce}.) □

Queries such as the query Q of Example 4.1 are of the kind that does not seem to have been studied before. For instance, implicit-wave CCQ queries cannot occur under set, bag, or bag-set semantics.[7] By the main result of this section, Theorem 4.1, under these three traditional semantics, as well as in other cases of combined semantics, there exist *symmetric CVM* mappings between equivalent CCQ queries. That is, for each pair (Q, Q') of combined-semantics CCQ queries such that each of Q and Q' is an explicit-wave query, $Q \equiv_C Q'$ implies that a CVM exists from Q to Q'. What is important is that in all such cases, a mapping of the *same type* (i.e., also a CVM) always exists also from Q' to Q. Example 4.1 illustrates that such symmetry does not hold for unrestricted pairs of CQ queries under combined semantics.

We now state Theorem 4.1.

[7]We prove this claim in Section 6.

Theorem 4.1. *Given CCQ queries Q and Q', such that (i) Q is an explicit-wave query, and (ii) $Q \equiv_C Q'$. Then there exists a SCVM from Q' to Q.* □

Theorems 3.3 and 4.1 yield immediately a necessary and sufficient equivalence condition for CCQ explicit-wave queries. We explore this equivalence condition in detail in Section 6.

Due to the well-known example of [4] (Appendix I), it appears that condition (ii) of Theorem 4.1 cannot be replaced by condition $Q \sqsubseteq_C Q'$ (while also replacing SCVMs by CVMs), even when Q is an explicit-wave query. Alternatively, we cannot remove condition (i) of Theorem 4.1. Indeed, in Example 4.1 there is a SCVM from query Q to explicit-wave query Q', but there is no SCVM from Q' to Q, even though $Q \equiv_C Q'$ holds, see Appendix K. Thus, Theorem 4.1 provides an *asymmetric* necessary condition for CCQ-query equivalence. This asymmetry does not appear to have been explored in previous work. One reason for this is that, as we have mentioned, under the three traditional semantics all CCQ queries are explicit-wave queries. In [6], Cohen explores query classes that properly subsume the class of CCQ queries. When restricted to CCQ queries, all the necessary and sufficient conditions of [6] for combined-semantics query equivalence require the queries to be explicit-wave queries. (We note that none of the necessary and sufficient conditions of [6] applies to our Examples 3.1 or 3.2, even though all the queries in the two examples are explicit-wave queries. Yet, by an equivalence test that is immediate from our Theorems 3.3 and 4.1, $Q \not\equiv_C Q'$ for the queries of Example 3.1, and $Q \equiv_C Q'$ for the queries of Example 3.2. See Appendix N for all the details.)

In the remainder of this section, we outline the idea of the proof of Theorem 4.1. (The full proof can be found in Appendix L.) Intuitively, we generalize the proof, via canonical databases, of the existence of a containment mapping from CCQ set query Q' to CCQ set query Q whenever $Q \equiv_S Q'$. There is a major challenge in the generalization: We are now looking not just for a containment mapping, but for a SCVM from Q' to Q. That is, the desired mapping must map each multiset variable of Q' into a distinct multiset variable of Q. Showing that we have constructed a mapping with this property is thus an essential part of the proof. (Note that in Theorem 4.1, we have no information about the structural relationship between the given queries Q and Q'.)

For a given CCQ query Q, the proof of Theorem 4.1 constructs an infinite number of databases, where each database $D_{\bar{N}^{(i)}}(Q)$, $i \geq 1$, can be thought of as a union of "extended canonical databases" for Q. (See Appendix G.1 for the definition.) Similarly to canonical databases for CCQ set queries, each ground atom in each database $D_{\bar{N}^{(i)}}(Q)$ can be associated, via a mapping that we denote $\nu_Q^{(i)}$, with a unique subgoal of the query Q.

The role of each database $D_{\bar{N}^{(i)}}(Q)$ in the proof of Theorem 4.1 is that the database represents a particular combination of multiplicities of the values of (some of) the multiset variables $Y_1, Y_2, \ldots, Y_n$, for some $n \geq 1$, of the query Q. (We have that $n \geq 1$ for all CCQ queries Q and Q' such that $Q \equiv_C Q'$ and at least one of Q and Q' is not a set query.) For each database $D_{\bar{N}^{(i)}}(Q)$, we represent the n respective multiplicities as natural numbers $N_1^{(i)}$ through $N_n^{(i)}$, or equivalently via the n-ary vector $\bar{N}^{(i)}$.

By construction of the databases $D_{\bar{N}^{(i)}}(Q)$, we have that some fixed tuple, t_Q^*, is an element of the bag $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$

for each $i \geq 1$. Moreover, *for all queries Q'' such that* (Q, Q'') *is an equivalence-compatible CCQ pair,* we have that the multiplicity of the tuple t^*_Q in each bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$ (that is, for each $i \geq 1$) can be expressed using the symbolic representations, N_1 through N_n, of the respective elements $N_1^{(i)}, \ldots, N_n^{(i)}$ of the vector $\bar{N}^{(i)}$. That is, for each such query Q'', we can obtain explicitly a function, $\mathcal{F}^{(Q'')}_{(Q)}$, in terms of the n variables $N_1, \ldots, N_n$, such that whenever we substitute $N_j^{(i)}$ for N_j, for each $j \in \{1, \ldots, n\}$, the resulting expression in terms of $N_1^{(i)}, \ldots, N_n^{(i)}$ evaluates to the multiplicity of the tuple t^*_Q in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$.

A key observation in the proof of Theorem 4.1 is that for our fixed query Q and for each CCQ query Q' such that $Q' \equiv_C Q$, it must be that the functions $\mathcal{F}^{(Q')}_{(Q)}$ and $\mathcal{F}^{(Q)}_{(Q)}$ output the same value on each database $D_{\bar{N}^{(i)}}(Q)$, $i \geq 1$.

Consider the simplest case, where our query Q has no self-joins and has $|M| = n \geq 1$. In this case, by construction of the databases, we have that the function $\mathcal{F}^{(Q)}_{(Q)}$ for the query Q is the monomial $\Pi^n_{j=1} N_j$. Consider an arbitrary assignment, γ, from Q to a $D_{\bar{N}^{(i)}}(Q)$. We have that each such γ has contributed to the construction of the database; we call γ a *generative assignment from Q to $D_{\bar{N}^{(i)}}(Q)$*. We can show that the composition $\nu_Q^{(i)} \circ \gamma$ is a SCVM from Q to itself. (Note the presence in the product $\Pi^n_{j=1} N_j$ of the variables for all the n multiset variables of Q.) Moreover, for each query Q' such that $Q' \equiv_C Q$, the function $\mathcal{F}^{(Q')}_{(Q)}$ is forced (by $Q' \equiv_C Q$ and by $\mathcal{F}^{(Q)}_{(Q)}$ being a multivariate polynomial) to be exactly $\Pi^n_{j=1} N_j$, regardless of the structural relationship between Q and Q'. We show that whenever $\mathcal{F}^{(Q')}_{(Q)} = \Pi^n_{j=1} N_j$, an assignment from Q' to a database $D_{\bar{N}^{(i)}}(Q)$ can be composed with the mapping $\nu_Q^{(i)}$ to yield a SCVM from Q' to Q, precisely due to the presence in the function $\mathcal{F}^{(Q')}_{(Q)}$ of the "representative" N_j of each multiset variable Y_j of the query Q, for $1 \leq j \leq n$.

We now use the discussion of this simplest special case to provide a general high-level intuition of the proof of Theorem 4.1: It turns out that for all CCQ queries Q, there is a monomial, in terms of *all* of $N_1, \ldots, N_n$, that contributes to the construction of the function $\mathcal{F}^{(Q)}_{(Q)}$ and that reflects the multiplicity, in the set[8] $\Gamma^{(t^*_Q)}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$, of all generative assignments from Q to databases $D_{\bar{N}^{(i)}}(Q)$. We call this monomial, $\mathcal{P}^{(Q)}_*$, *the wave of the query Q*. Suppose that, for a query Q' such that $Q' \equiv_C Q$, we can show that the function $\mathcal{F}^{(Q')}_{(Q)}$ has, as a term, the wave of Q *backed up by assignments from Q' to the databases $D_{\bar{N}^{(i)}}(Q)$*. Then we can use these assignments and the mapping $\nu_Q^{(i)}$ to construct a SCVM from Q' to Q.

There are two significant challenges in extending this idea to all CCQ queries. First, the term $\mathcal{P}^{(Q)}_*$ may not be "visible" in the expression for $\mathcal{F}^{(Q)}_{(Q)}$. As a result, $\mathcal{P}^{(Q)}_*$ does not necessarily contribute to the construction of the function $\mathcal{F}^{(Q')}_{(Q)}$, even in case $Q \equiv_C Q'$. (See, e.g., queries Q and Q' of Example 4.1, and Example L.1 in Appendix L.1 for the details.) Second, in general, function $\mathcal{F}^{(Q')}_{(Q)}$ may have terms that are *not* backed up by assignments from Q' to databases $D_{\bar{N}^{(i)}}(Q)$. Both challenges arise from the fact that the function $\mathcal{F}^{(Q'')}_{(Q)}$, in terms of $N_1, \ldots, N_n$, is, in general, *not* a multivariate polynomial on its entire domain.

To overcome the first challenge, we introduce the restriction that Q be an *explicit-wave query.* (Hence Definition 4.1 is necessarily technical.) Even under this restriction, overcoming the second challenge requires a nontrivial proof. See Appendix L for all the details of the proof of Theorem 4.1.

Example L.1 in Appendix L.1 illustrates how the term $\mathcal{P}^{(Q)}_*$ may not be "visible" in the formula $\mathcal{F}^{(Q)}_{(Q)}$, and how, in general, the function $\mathcal{F}^{(Q)}_{(Q)}$ is not a multivariate polynomial on its entire domain. Example L.6 in Appendix L.9.6 is an extended variant of Example L.1. In addition, Example L.6 illustrates how function $\mathcal{F}^{(Q)}_{(Q)}$ may have terms that are *not* backed up by assignments from Q to databases $D_{\bar{N}^{(i)}}(Q)$.

5. MINIMIZING CCQ QUERIES

In this section we propose a sound and complete algorithm for minimizing CCQ queries. We also show that for *all* CCQ queries, including implicit-wave queries, the minimized version of the query exists and is unique up to an isomorphism SCVM. (An *isomorphism SCVM* from CCQ query Q to CCQ query Q' is a SCVM from Q to Q' that is an isomorphism mapping from the terms of Q to the terms of Q'.) It turns out that minimizing arbitrary CCQ queries is at most as hard as minimizing CQ *set* queries using the approach of [2]. Besides being contributions of this paper in their own right, the results of this section permit us to formulate, in Section 6, equivalence tests for CCQ queries that tie our results together with those of [2, 4].

By Definition 2.3, to find a minimized version of a CCQ query Q, one would remove subsets of subgoals of Q so as to arrive at a minimized query Q' such that $Q' \equiv_C Q$. Our approach is to generalize to the case of CCQ queries the minimization approach that was applied in [2] to CQ set queries: Given a CCQ query Q, the search space of all candidate minimized versions of Q can be restricted to the set $\mathcal{Q}_{min}(Q)$. This set comprises all reduced-condition queries Q' for Q such that (i) Q' has all the multiset variables of the query Q, and such that (ii) there exists a GCM from Q_{ce} *onto* Q'_{ce}. By Theorems 3.1 and 3.2, the set $\mathcal{Q}_{min}(Q)$ contains all minimized versions of the query Q.

It turns out that for every CCQ query Q, the search space $\mathcal{Q}_{min}(Q)$ can be found by generating all CCQ queries obtainable by applying to Q all possible SCVMs of a certain type from Q to itself. This result holds by Proposition 5.1. For CCQ queries $Q(\bar{X}) \leftarrow L, M$ and $Q'(\bar{X}) \leftarrow L', M$ (note the same head vector and the same set M) and for a SCVM μ from Q to Q', we call μ an *M-identity SCVM* whenever μ maps each multiset variable of Q to itself. SCVM μ induces a mapping from L onto some subbag L'' of L'. (Here we treat the conjunctions L', L'' as bags of atoms.) We define $\mu(Q)$ as a CCQ query that is identical to Q except that the condition of $\mu(Q)$ is L''.

Proposition 5.1. *Given a CCQ query Q, with reduced-condition query Q' that retains all the multiset variables of Q. Then there exists a GCM from Q_{ce} onto Q'_{ce} if and only if there exists an M-identity SCVM from Q onto Q'.* □

[8]For a CCQ k-ary query Q ($k \geq 1$), for a database D, and for a fixed k-tuple t, we denote by $\Gamma^{(t)}_{\bar{S}}(Q, D)$ the set of all tuples t' in $\Gamma_{\bar{S}}(Q, D)$ such that the projection of t' on the vector of the head arguments of the query Q is the tuple t.

The intuition for the only-if part of the proof is that each GCM, ν, from Q_{ce} onto Q'_{ce} induces an automorphism from the condition of Q' *as part of the condition of* Q, to the condition of Q' (in Q'). One can use this fact to take an inverse of the mapping resulting from restricting the domain of ν to the range of ν, and by composing that inverse with ν to obtain the desired M-identity SCVM from Q onto Q'. See Appendix M.1 for the full proof. Consider an illustration.

EXAMPLE 5.1. *Let queries Q and Q' be as follows.*

$Q(X) \leftarrow p(X,Y,W;i), p(X,W,Y), p(X,Y,Z), \{Y,i\}.$
$Q'(X) \leftarrow p(X,Y,W;i), p(X,W,Y), \{Y,i\}.$

Q' is a proper reduced-condition query for Q that preserves the multiset variables of Q. Q_{ce} and Q'_{ce} are as follows.

$Q_{ce}(X) \leftarrow p(X,Y,W;i), p(X,W,Y;j), p(X,Y,Z;k), \{Y,i,j,k\}.$
$Q'_{ce}(X) \leftarrow p(X,Y,W;i), p(X,W,Y;j), \{Y,i,j\}.$

There exists a GCM ν from Q_{ce} onto Q'_{ce}: $\nu = \{ X \rightarrow X, Y \rightarrow W, Z \rightarrow Y, W \rightarrow Y, i \rightarrow j, j \rightarrow i, k \rightarrow j \}$. Observe that the mapping ν_1 resulting from restricting the domain of ν to the range of ν, that is to the set $\mathcal{X} = \{ X, Y, W, i, j \}$, is a bijection. Thus, there exists an inverse mapping, $\nu_1^{-1} = \{ X \rightarrow X, Y \rightarrow W, W \rightarrow Y, i \rightarrow j, j \rightarrow i \}$. Further, the mapping $\nu' = \nu_1^{-1} \circ \nu$ is well defined, because the range of ν is the domain of ν_1^{-1}. Finally, when the domain of ν' is restricted to the set $\mathcal{X}$, then the resulting mapping is an identity mapping by definition. Hence, we obtain that ν' is a GCM from Q_{ce} onto Q'_{ce}, and that the mapping resulting from restricting the domain of ν' to the set of variables of the query Q is an M-identity SCVM from Q onto Q'. $\square$

We use the result of Proposition 5.1 to develop algorithm MINIMIZE-CCQ-QUERIES. The pseudocode is as follows.

Algorithm MINIMIZE-CCQ-QUERIES:
Input: CCQ query Q.
Output: CCQ query Q^{min} such that Q^{min} is a minimized version of Q by Definition 2.3.

```
1. Set $Q^{min}$ to the regularized version of $Q$;
        // Note that $Q^{min} \in \mathcal{Q}_{min}(Q)$
2. While (there exists an M-identity SCVM $\mu$ from
   $Q^{min}$ to itself such that $\mu(Q^{min})$ has fewer subgoals
   than $Q^{min}$) // Note that $\mu(Q^{min}) \in \mathcal{Q}_{min}(Q)$
      3. Set $Q^{min}$ to $\mu(Q^{min})$;
4. Output $Q^{min}$.
```

Algorithm MINIMIZE-CCQ-QUERIES is a straightforward generalization to CCQ queries of the minimization algorithm applied by [2] to CCQ *set* queries. That is, our algorithm obtains recursively "shorter-condition" reduced-condition queries Q' for the input query Q, such that each $Q' \in \mathcal{Q}_{min}(Q)$. The algorithm terminates once no more M-identity SCVM can "shorten" any further the condition of Q'.

PROPOSITION 5.2. *Given a CCQ query Q, algorithm* MINIMIZE-CCQ-QUERIES *outputs a minimized version of Q.* $\square$

The proof of Proposition 5.2 (in Appendix M.2) is by showing that for each input CCQ query Q, the output of the algorithm MINIMIZE-CCQ-QUERIES satisfies Definition 2.3 with respect to Q.

In addition to being sound, algorithm MINIMIZE-CCQ-QUERIES is also complete, due to the following result:

THEOREM 5.1. *Given a CCQ query Q, the minimized version of Q exists and is unique up to an isomorphism M-identity SCVM.* $\square$

Note that Theorem 5.1 reduces correctly to the special case of set queries, see Theorem 2.2 (1). (Recall that CCQ set queries have zero multiset variables.)

THEOREM 5.2. *Algorithm* MINIMIZE-CCQ-QUERIES *is sound and complete for CCQ queries.* $\square$

The result of Theorem 5.2 is immediate from Proposition 5.2 and from Theorem 5.1.

The asymptotic worst-case time complexity of the algorithm MINIMIZE-CCQ-QUERIES is the same as that for the minimization algorithm of [2] (for CCQ set queries). Indeed, by definition of M-identity SCVMs and from the fact that CCQ set queries have zero multiset variables, finding a minimized version of a set query is at least as hard as finding a minimized version of any CCQ query. This fact is due to the absence, in case of set queries, of any "identity bindings" for multiset variables of the query, in an M-identity SCVM from a set query to itselt. As a result, we obtain the following.

PROPOSITION 5.3. *Finding a minimized version of a CCQ query is NP complete.* $\square$

We now establish the result of Theorem 5.1. In the proof, we use the following result. (See Appendix M.3 for the proof of Proposition 5.4.)

PROPOSITION 5.4. *Given CCQ queries Q_1 and Q_2 such that there exists a CVM μ_1 from Q_1* onto *Q_2, and another CVM μ_2 from Q_2* onto *Q_1. Then each of μ_1 and μ_2 is an isomorphism SCVM.* $\square$

The proof of Theorem 5.1 is provided in Appendix M.4. The idea of the proof is as follows. First, the existence of a minimized version of Q follows from Proposition 5.2. Second, suppose there exist two distinct minimized versions of Q, Q_1 and Q_2, where each of Q_1 and Q_2 satisfies Definition 2.3 w.r.t. Q. The proof of Theorem 5.1 establishes that there exists an M-identity SCVM from Q_1 *onto* Q_2, and another from Q_2 *onto* Q_1. Then Proposition 5.4 is used to conclude that each of the two M-identity SCVMs is an isomorphism SCVM. Consider an illustration.

EXAMPLE 5.2. *Consider CCQ query Q and two reduced-condition queries for Q, Q_1 and Q_2.*

$Q(X) \leftarrow p(X,Y;i), p(Y,W), p(Y,T), \{Y,i\}.$
$Q_1(X) \leftarrow p(X,Y;i), p(Y,W), \{Y,i\}.$
$Q_2(X) \leftarrow p(X,Y;i), p(Y,T), \{Y,i\}.$

By Definition 2.3 and by Theorem 3.3, each of Q_1 and Q_2 is a minimized version of the query Q.

Consider mapping μ_1 from Q to Q_1: $\mu_1 = \{ X \rightarrow X, Y \rightarrow Y, i \rightarrow i, W \rightarrow W, T \rightarrow W \}$. This mapping is an M-identity SCVM from Q onto Q_1. When restricted to the domain that is the set of terms of the query Q_2, call this mapping ν_1, mapping μ_1 furnishes an isomorphism M-identity SCVM from Q_2 onto Q_1. The mapping ν_1^{-1} is an isomorphism M-identity SCVM from Q_1 onto Q_2. $\square$

We now contrast these results with our Theorem 4.1. By the results of this current section, a SCVM always exists from an arbitrary CCQ query into its minimized version. Intuitively, this holds even for implicit-wave queries because a minimized version Q^{min} of a query Q is a reduced-condition query for Q. Hence, in some sense we know the *structure* of Q^{min}. In contrast, for two general CCQ queries Q and Q' such that $Q \equiv_C Q'$, all we know is that on all databases D, the bags $Res_C(Q, D)$ and $Res_C(Q', D)$ are identical. In general, no information is available about the relationship between the structures of Q and Q'. Thus, Theorem 4.1 does not necessarily hold for the case of implicit-wave queries.

6. CCQ-QUERY EQUIVALENCE

In this section we study the conditions of combined-semantics equivalence of CCQ queries that are immediate from the results of Sections 3–4. Our focus is on reformulating these conditions using minimized query versions. The reformulations tie our equivalence conditions together with the results of [2, 4]. Our necessary and sufficient query-equivalence condition, Theorem 6.3, applies to the class of all explicit-wave CCQ queries. We show that this class encompasses strictly more queries than (i) all CCQ set, bag, and bag-set queries, and than (ii) all CCQ queries for which [6] provides both sufficient and necessary equivalence conditions.[9]

First, Theorem 3.3 (Section 3) gives us a sufficient condition for combined-semantics equivalence of CCQ queries:

THEOREM 6.1. *Given CCQ queries Q_1 and Q_2. If there exists a CVM from Q_1 to Q_2, and another from Q_2 to Q_1, then we have $Q_1 \equiv_C Q_2$.* □

We reformulate this theorem using the results of Section 5:

THEOREM 6.2. *Given CCQ queries Q_1 and Q_2, with respective minimized versions Q_1^{min} and Q_2^{min}. If there exists an isomorphism SCVM from Q_1^{min} to Q_2^{min}, and another from Q_2^{min} to Q_1^{min}, then we have $Q_1 \equiv_C Q_2$.* □

It turns out that the sufficient query-equivalence conditions of Theorems 6.1 and 6.2 have the same power. (The proof of Theorem 6.2 is immediate from Theorem 6.1 and Proposition 6.1.)

PROPOSITION 6.1. *Given CCQ queries Q_1 and Q_2, with respective minimized versions Q_1^{min} and Q_2^{min}. Then:*

- *There exists a CVM from Q_1 to Q_2, and another from Q_2 to Q_1, if and only if*
- *There exists an isomorphism SCVM from Q_1^{min} to Q_2^{min}, and another from Q_2^{min} to Q_1^{min}.* □

See Appendix O for the proof of Proposition 6.1. The only-if part of the proof is based on Proposition 5.4.

Neither Theorem 6.1 nor Theorem 6.2 gives us a necessary condition for combined-semantics equivalence of two CCQ queries. (We have Example 4.1 as a counterexample. Observe that both queries in Example 4.1 are represented by their minimized versions.)

At the same time, we use Theorems 4.1 and 6.2, as well as Proposition 6.1, to formulate a sufficient and necessary condition for equivalence of explicit-wave CCQ queries.

[9]We already showed (ii) in Section 4; see Appendix N.

THEOREM 6.3. *Given explicit-wave CCQ queries Q_1 and Q_2, with respective minimized versions Q_1^{min} and Q_2^{min}. Then $Q_1 \equiv_C Q_2$ if and only if there exists an isomorphism SCVM from Q_1^{min} to Q_2^{min}, and another from Q_2^{min} to Q_1^{min}.* □

Another sufficient and necessary condition for equivalence of explicit-wave CCQ queries, in terms of CVMs, can be obtained by using only Theorems 4.1 and 6.1. (See Theorem N.1 in Appendix N.)

We now show that Theorem 6.3 generalizes Theorem 2.2 (2), due to [2], as well as Theorem 2.3, due to [4]. To do this, we show that all CCQ set, bag, and bag-set queries are explicit-wave queries, and then consider minimization of CCQ bag and bag-set queries.

By Condition (1) of Definition 4.1, we have that all set and bag-set CCQ queries are explicit-wave queries. Besides that condition, one can formulate a number of easy syntactic tests, each of which is a sufficient condition for a CCQ query to be an explicit-wave query. (E.g., it is immediate from Definition 4.1 that a CCQ query without self-joins is an explicit-wave query.) One sufficient condition is that a CCQ query Q is an explicit-wave query whenever each copy-sensitive subgoal of Q has no set variables. (In this case, it is easy to see that Condition (2) of Definition 4.1 is always satisfied; see Appendix P.) By this condition, all CCQ bag queries are explicit-wave queries. Other sufficient conditions could generalize the case of the explicit-wave CCQ query Q' of Example 4.1 (note that while being an explicit-wave query, this query does not satisfy any of the above sufficient conditions for a query to be explicit-wave), and so on. As a result, we have the following:

PROPOSITION 6.2. *The set of all CCQ set, bag, and bag-set queries is a proper subset of the set of all explicit-wave CCQ queries.* □

One can argue that CCQ queries that have set variables in copy-sensitive subgoals, such as the implicit-wave query Q of Example 4.1, would not tend to be popular – and may not even be expressible – in practical applications. Hence, we posit that implicit-wave queries may be unlikely to arise in practice.

Now that we know that all CCQ set queries are explicit-wave queries, it is easy to see that Theorem 6.3 generalizes properly Theorem 2.2 (2). (Theorem 2.2 (2) does not generalize to the case of all CCQ queries because not all CCQ queries are explicit wave, see discussion of Example 4.1 earlier in this section.) For instance, by Theorem 6.3 we have that for the queries of Example 3.2, $Q \equiv_C Q'$ holds. The reason is, both Q and Q' in that example are explicit-wave queries and, in addition, Q' is the minimized version of Q.

Observe that Theorem 6.3 is not a trivial generalization of Theorem 2.2 (2). Indeed, the two explicit-wave CCQ queries of Example 3.1 are isomorphic but, by Theorem 6.3, are not combined-semantics equivalent. (Both queries of Example 3.1 are represented by their minimized versions.)

Finally, we consider minimization of CCQ bag and bag-set queries. Recall that each subgoal of a CCQ bag query has a copy variable. Hence, by Theorem 3.1, each CCQ bag query is its unique minimized version. We conclude that the result (due to [4]) of Theorem 2.3 (1) is a special case of Theorem 6.3. (See Note 1 in Section 2.2.)

In case of CCQ bag-set queries, the only terms that can appear in such a query are multiset noncopy variables, head variables, and constants. We have from the results of Section 5 that for each CCQ query, there exists an M-identity SCVM from the regularized version of the query to its minimized version. (See Proposition M.1 and algorithm MINIMIZE-CCQ-QUERIES in Section 5.) Now the regularized version Q_r of a CCQ bag-set query Q is the canonical representation [4] of Q, that is, the result of dropping all duplicate subgoals from the condition of Q. By definition of M-identity SCVM, we have for all CCQ bag-set queries Q that for each subgoal, s, of Q_r, each M-identity SCVM maps s into itself. It follows that each M-identity SCVM maps Q_r *onto* itself. Thus, for each CCQ bag-set query Q, its canonical representation Q_r is its unique minimized version. Hence Theorem 2.3 (2) is a special case of Theorem 6.3.

7. RELATED WORK

In their classic paper [2], Chandra and Merlin presented an NP-complete containment test for CQ queries under set semantics. This sound and complete test has been used in optimization, via minimization, of CQ set-semantics queries, as well as in developing algorithms for rewriting queries (both equivalently and nonequivalently) using views. We are not aware of past work that studies minimization of queries beyond the language of CQ set-semantics queries. In this current paper we extend the results of [2] to general CQ combined-semantics queries, and show the limitations of each extension. We show that the minimization approach of [2] can be extended to general CQ queries without limitations. Remarkably, minimizing arbitrary CQ queries is at most as hard as minimizing CQ set-semantics queries.

Equivalence tests for CQ bag and bag-set queries were formulated by Chaudhuri and Vardi in [4]; correctness of the tests follows from the results of [6]. Our equivalence and minimization results for CQ combined-semantics queries reduce correctly to the special cases of CQ bag and bag-set queries, as given in [4]. Further, this current paper provides a nontrivial generalization and the first known proof of the well-known sufficient containment condition for CQ bag queries, as outlined in [4].

Definitive results on containment between CQ queries under bag and bag-set semantics have not been obtained so far. Please see Jayram, Kolaitis, and Vee [10] for original undecidability results on containment of CQ queries with inequalities under bag semantics. The authors point out that it is not known whether the problem of bag containment for *CQ* queries is even decidable. For the case of *bag-set* semantics, sufficient conditions for containment of two CQ queries can be expressed via containment of (the suitable) aggregate queries with aggregate function `count(*)`. The latter containment problem can be solved using the methods proposed in [7]. Please see [4, 1] for other results on bag and bag-set containment of CQ queries. The general problems of containment for CQ bag and bag-set queries remain open.

In her papers [5, 6], Cohen provided an elegant and powerful formalism for treating queries evaluated under each of set, bag, and bag-set semantics uniformly as special cases of the more general combined semantics. The papers also contain a general sufficient condition for combined-semantics equivalence of CQ queries with disjunction, negation, and arithmetic comparisons, as well as necessary and sufficient equivalence conditions for special cases. (Interestingly, when we restrict the language of the queries in question to the language of CQ queries, it turns out that all the necessary and sufficient query-equivalence conditions of [6] hold for queries belonging collectively to a proper subclass of the class of explicit-wave CQ queries, which (class) we introduce in this current paper.) The proof in [6] of its general sufficient condition for equivalence of queries is in terms of containment between the queries under combined semantics. That (implicit) sufficient query-containment condition is proved in [6] for the case where the two queries have the same number of multiset variables. In this current paper we provide proper generalizations of all the results of [6], including of its implicit sufficient condition for query containment, provided that the results of [6] are applied to CQ queries only.

A discussion of query equivalence and containment for query languages that properly contain the language of CQ queries is beyond the scope of this paper. The interested reader is referred to [6], which contains an excellent overview of the literature in this direction.

8. REFERENCES

[1] F. N. Afrati, M. Damigos, and M. Gergatsoulis. Query containment under bag and bag-set semantics. *Information Processing Letters*, 110(10):360–369, 2010.

[2] A. Chandra and P. Merlin. Optimal implementation of conjunctive queries in relational data bases. In *STOC*, 1977.

[3] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. *SIGMOD Record*, 26(1):65–74, 1997.

[4] S. Chaudhuri and M. Y. Vardi. Optimization of *real* conjunctive queries (extended abstract). In *PODS*, 1993.

[5] S. Cohen. Equivalence of queries combining set and bag-set semantics. In *PODS*, pages 70–79, 2006.

[6] S. Cohen. Equivalence of queries that are sensitive to multiplicities. *The VLDB Journal*, 18:765–785, 2009.

[7] S. Cohen, W. Nutt, and Y. Sagiv. Containment of aggregate queries. In *ICDT*, pages 111–125, 2003.

[8] A. Gupta and I. S. Mumick, editors. *Materialized Views: Techniques, Implementations, and Applications.* The MIT Press, 1999.

[9] Y. Ioannidis and R. Ramakrishnan. Containment of conjunctive queries: beyond relations as sets. *ACM TODS*, 20(3):288–324, 1995.

[10] T. Jayram, P. Kolaitis, and E. Vee. The containment problem for *real* conjunctive queries with inequalities. In *PODS*, pages 80–89, 2006.

[11] W. Lehner. Query processing in data warehouses. In *Encyclopedia of Database Systems*, pages 2297–2301. Springer, 2009.

[12] N. Pendse and R. Creeth. The OLAP report. *Business Intelligence*, 1995. The 2008 update available at `http://www.bi-verdict.com/fileadmin/FreeAnalyses/fasmi.htm`.

[13] A. Shukla, P. Deshpande, and J. F. Naughton. Materialized view selection for multi-cube data models. In *EDBT*, 2000.

[14] M. Zaharioudakis, R. Cochrane, G. Lapis, H. Pirahesh, and M. Urata. Answering complex SQL queries using automatic summary tables. In *SIGMOD*, pages 105–116, 2000.

APPENDIX

A. EXAMPLE FROM BOOK BY GUPTA AND MUMICK

Consider an example from the book "Materialized views: Techniques, Implementations, and Applications" by Ashish Gupta and Inderpal Singh Mumick [8]; the purpose of the example in [8] is to explore how views can be used in query answering. In this current paper, we use the application domain of this Example A.1 in Example 1.1 in Section 1.

EXAMPLE A.1. *[8] Consider a retailer that has multiple stores across the United States, and the country is divided into multiple regions for administrative and accounting purposes. Each retailer carries many items and has an elaborate relational database/warehouse for analysis, marketing, and promotion purposes. Consider some of the tables in such a database and their cardinality:*

```
pos(itemID, storeID, date, qty, price) - 1,000,000,000 rows
stores(storeID, city, region) - 100 rows
items(itemID, name, category, cost) - 50,000 rows
```

The `pos` *table represents point-of-sale transactions, with one tuple for every item sold in a transaction. The tuple has the ID of the item sold, the ID of the store selling it, the date of sale, the quantity of the item sold, and its selling price. The* `stores` *table has location information about each store – namely its ID, city, and geographical region. The* `items` *table describes each item – namely its ID, name, product category, and cost price per unit.*

Suppose that the business-development division of this store chain wants to know the total revenue generated for each store by each category of items (query `QA`*). Further, the division is also interested in monitoring the total sales for each region (query* `QB`*). The two queries are as follows:*

```
(QA): SELECT storeID, category, SUM(qty*price)
      FROM pos, items
      WHERE pos.itemID = items.itemID
      GROUP BY storeID, category

(QB): SELECT region, SUM(qty*price) FROM pos, stores
      WHERE pos.storeID = stores.storeID GROUP BY region
```

Consider a view `totalSales` *that can be defined on the relation* `pos`*:*

```
DEFINE VIEW totalSales(itemID, storeID, total) AS
      SELECT itemID, storeID, SUM(qty*price) FROM pos
      GROUP BY itemID, storeID
```

The above view can be used in place of the table `pos` *to evaluate queries* `QA` *and* `QB`*, as follows:*

```
(QA'): SELECT storeID, category, SUM(total)
       FROM totalSales, items
       WHERE totalSales.itemID = items.itemID
       GROUP BY storeID, category

(QB'): SELECT region, SUM(total) FROM totalSales, stores
       WHERE totalSales.storeID = stores.storeID
       GROUP BY region
```

It can be shown formally that for each possible database, `QA` *and* `QA'` *return the same answer on the database, and similarly for* `QB` *and* `QB'`*.* □

B. DATALOG REPRESENTATIONS OF THE SQL QUERIES OF EXAMPLE 1.1

In this section we provide Datalog-based representations as introduced in [5, 6], of the four SQL queries of Example 1.1. We refer to the Datalog version of query `Q1` as $q1$, and so on.

$$q1(S, A) \leftarrow pos(X, I, S, D, A), pos(Y, I, S, d1, A), \{X, I, D\}.$$

$$q2(S, A) \leftarrow pos(X, I, S, d1, A), \emptyset.$$

$$q3(S, A) \leftarrow pos(X, I, S, D, A), pos(Y, I, S, d1, A), \emptyset.$$

$$q4(S, A) \leftarrow pos(X, I, S, D, A), pos(Y, I, S, d1, A), \{X, I, D, Y\}.$$

All the four queries are combined-semantics CQ queries. Further, each of $q2$ and $q3$ is a set-semantics query, and $q4$ is a bag-set-semantics query. If we allow for databases that can have duplicate tuples in stored relations, then we can write a bag-semantics analog, $q5$, of the query $q4$, as follows.

$$q5(S, A) \leftarrow pos(X, I, S, D, A; j), pos(Y, I, S, d1, A; k), \{X, I, D, Y, j, k\}.$$

Unlike queries $q2$ through $q5$, query $q1$ is a combined-semantics CQ query that is not combined-semantics equivalent to any set, bag, or bag-set query that can be posed on the database schema of Example 1.1. This nonequivalence is due to the following:

(i) The multiplicity of the domain values that some nondistinguished variables of $q1$ (specifically X, I, and D) can accept influences the multiplicity of answers to the query.

(ii) At the same time, the multiplicity of the domain values that all the other nondistinguished variables of $q1$ (specifically Y) can accept does not influence the multiplicity of answers to the query.

Queries with these properties (such as $q1$) cannot be expressed equivalently as set, bag, or bag-set semantics queries. The reason is, all nondistinguished variables in each set (bag, bag-set) query are of the same kind: Either the kind described in item (i) above *(multiset variables)*, or the kind described in item (ii) *(set variables)*.

C. ADDITIONAL EXAMPLE OF A SQL COMBINED-SEMANTICS QUERY

In this section we provide an extra illustration of how combined-semantics queries arise naturally in SQL in data-analysis applications. Several more examples of combined-semantics CQ queries (and views) expressed in SQL can be found in Section 1 of [6].

EXAMPLE C.1. *Recall the relations* `pos`, `stores`, *and* `items` *that were all introduced in Example A.1. Consider a database that has all these relations, as well as two extra relations. First, it has relation* `distributors(`<u>`distribID`</u>`, city, region)`*, which stores address information about distributors/warehouses that supply items to stores. Second, it has a relation* `suppliers (`<u>`storeID`</u>`,` <u>`distribID`</u>`, contractType)`*, which associates each store with the distributors that ship items to the store. Key attributes are underlined in the schemas. We make the realistic assumption of a* *many-to-many relationship between store IDs and distributor IDs in the* `suppliers` *relation; this relationship will give rise to a "non-star-schema join" in the query* `QC`*.*

In addition to the queries `QA` *and* `QB` *of Example A.1, consider a query* `QC` *that asks for the total revenue generated by stores in each city/region, but where the revenue is based on only those stores that have at least one distributor from a fixed region: the state of Florida in the USA. Query* `QC` *can be written in SQL as follows.*

```
(QC):SELECT city, region, SUM(qty*price) FROM pos, stores S
     WHERE pos.storeID = S.storeID
     AND EXISTS
     (SELECT * FROM suppliers P, distributors D
      WHERE P.storeID = S.storeID
      AND P.distribID = D.distribID AND D.region = 'FL USA')
     GROUP BY city, region
```

To evaluate the query `QC`*, the query processor will first compute all the tuples* `(city, region, qty, price)` *that satisfy the* `WHERE` *clause of* `QC`*. In other words, the query processor will first evaluate the* unaggregated *query* `QC'` *whose definition follows. Then, to produce the answers to the query* `QC`*, the query processor will apply the grouping and aggregation of* `QC` *to the relation that is the answer to the query* `QC'`*. The query* `QC'` *is obtained easily by removing the aggregation function and the* `GROUP BY` *clause from the query* `QC`*:*

```
(QC'):SELECT city, region, qty, price FROM pos, stores S
      WHERE pos.storeID = S.storeID
      AND EXISTS
     (SELECT * FROM suppliers P, distributors D
      WHERE P.storeID = S.storeID
      AND P.distribID = D.distribID AND D.region = 'FL USA')
```

The query `QC'` *is a combined-semantics CQ query, which cannot be represented equivalently (in the sense of combined-semantics equivalence of [6]) by using a set query, or a bag query, or a bag-set query. Indeed, consider the Datalog representation of [6] of the query* `QC'`*:*

$$qc'(C, R, Q, P) \leftarrow pos(I, S, D, Q, P), stores(S, C, R), suppliers(S, F, T), distributors(F, G, fl\_usa), \{I, S, D\}.$$

The set, bag, and bag-set versions of qc' follow, named qs, qb, and qbs, respectively. Alongside each query we also provide its SQL representation. It is easy to show via counterexample databases that the query qc' (and thus its SQL version `QC'`*) is not combined-semantics equivalent to any of qs, qb, and qbs (and thus is not equivalent to their SQL counterparts). Thus, the SQL aggregate query* `QC` *cannot be evaluated by applying the grouping and aggregation of* `QC` *to the answers to the SQL counterparts of any of the queries qs, qb, and qbs. We list in this example (as* `QS`*,* `QB`*, and* `QBS`*) all SQL queries without subqueries that could be candidates for replacing* `QC'` *in the evaluation of* `QC`*. None of these queries is combined-semantics equivalent to* `QC'`*. Recall that CQ bag, set, and bag-set queries express only SQL queries without subqueries. We conclude that the SQL aggregate query* `QC` *cannot be evaluated by applying the grouping and aggregation of* `QC` *to (the SQL representation of) a CQ bag, bag-set, or set query.*

$$qs(C, R, Q, P) \leftarrow pos(I, S, D, Q, P), stores(S, C, R), suppliers(S, F, T), distributors(F, G, fl\_usa), \emptyset.$$

```
(QS):SELECT DISTINCT S.city, S.region, qty, price
     FROM pos, stores S, suppliers P, distributors D
     WHERE pos.storeID = S.storeID
     AND P.storeID = S.storeID
     AND P.distribID = D.distribID AND D.region = 'FL USA'
```

$$qb(C, R, Q, P) \leftarrow pos(I, S, D, Q, P; j), stores(S, C, R; k), suppliers(S, F, T; l), distributors(F, G, fl\_usa; m), \{I, S, D, j, k, F, T, l, G, m\}.$$

```
(QB):SELECT S.city, S.region, qty, price
     FROM pos, stores S, suppliers P, distributors D
     WHERE pos.storeID = S.storeID
     AND P.storeID = S.storeID
     AND P.distribID = D.distribID AND D.region = 'FL USA'
```

$$qbs(C, R, Q, P) \leftarrow pos(I, S, D, Q, P), stores(S, C, R), suppliers(S, F, T), distributors(F, G, fl\_usa), \{I, S, D, F, T, G\}.$$

```
(QBS):SELECT S.city, S.region, qty, price
     FROM pos, stores S, suppliers P, distributors D
     WHERE pos.storeID = S.storeID
     AND P.storeID = S.storeID
     AND P.distribID = D.distribID AND D.region = 'FL USA'
```

The only difference between the SQL queries `QB` *and* `QBS` *is that we require that* `QBS` *be posed only on databases whose all relations are sets (i.e., have no duplicate tuples). We do not impose this requirement onto the query* `QB`*.* □

D. TRADITIONAL SEMANTICS FOR CQ QUERIES

In this section we provide an overview, from [6], of the three traditional query semantics – that is, of set semantics, bag semantics, and bag-set semantics. The overview shows how these semantics can be formulated as special cases of combined semantics. This will make clear the relationship between the semantics introduced in [6] and previously studied semantics. For more details about set, bag, and bag-set semantics see [2, 4].

Combined-semantics queries differ from traditional Datalog queries in that (1) they have a set of multiset variables, and (2) they may have copy-sensitive atoms (i.e., copy variables). These two items are used to explicitly determine which variables and atoms should be interpreted under a multi-set semantics. Datalog queries with set, bag, and bag-set semantics do not have these items, since all variables and all atoms are interpreted in the same fashion, and thus these items can be implicitly determined.

Given a query $Q(\bar{T}) \leftarrow L, M$ and a database D, we use Q_{-copy} to denote the query derived from Q by removing all copy variables from Q. Similarly, we use Q_{+copy} to denote the query derived from Q by adding a copy variable to each relational atom in Q. We denote the result of applying Q to D under set, bag, and bag-set semantics as $Res_S(Q, D)$, $Res_B(Q, D)$, and $Res_{BS}(Q, D)$, respectively, and define these as follows:

$$Res_S(Q, D) \;:=\; \{\ \gamma(\bar{T}) \mid \gamma \in \Gamma(Q, D)\ \}.$$
$$Res_B(Q, D) \;:=\; \{\!\{\ \gamma(\bar{T}) \mid \gamma \in \Gamma(Q_{+copy}, D)\ \}\!\}.$$
$$Res_{BS}(Q, D) \;:=\; \{\!\{\ \gamma(\bar{T}) \mid \gamma \in \Gamma(Q_{-copy}, D)\ \}\!\}.$$

Here, $\{\ldots\}$ and $\{\!\{\ldots\}\!\}$ denote sets and bags, respectively.

Remark (2.11 in [6]) When evaluating a query under bag-set semantics, one usually assumes that the database is a *set* of ground atoms, i.e., does not contain duplication. In the definition of $Res_{BS}(Q, D)$ in [6], this assumption is not made, but the result is indifferent to the number of copies of each ground atom in the database.

Under certain circumstances, combined semantics coincides with set, bag, or bag-set semantics. The following result is immediate from the definitions of the semantics.

PROPOSITION D.1. *[6] For CCQ query Q and database D:*

- *If Q is a set query then $Res_C(Q, D)$ is $Res_S(Q, D)$.*
- *If Q is a bag query then $Res_C(Q, D)$ is $Res_B(Q, D)$.*
- *If Q is a bag-set query then $Res_C(Q, D)$ is $Res_{BS}(Q, D)$.* □

E. PROOF OF THEOREM 3.1

In this section we provide a proof of Theorem 3.1. The proof uses counterexample databases, which are all constructed as discussed in Section E.1. Section E.2 formulates and proves properties of the counterexample databases. The proof of Theorem 3.1 can be found in Section E.3.

E.1 Constructing a Boxed Database

In this section we describe a procedure that, given a CCQ query $Q(\bar{X}) \leftarrow L, M$ with $p \geq 0$ distinct constants mentioned in Q, and given a positive integer A, produces a database $D^b_{Q,A}$ such that one fixed tuple in $Res_C(Q, D^b_{Q,A})$ has multiplicity *exactly* $(max(A, p))^{|M|}$.[10] The general procedure of producing (among other databases) a "boxed database" $D^b_{Q,A}$ for Q and A as above is used in several proofs in this paper. This construction has been inspired in part by a proof of Lemma 4.1 of [6].

Procedure Template-Boxed-Database (Input: $V_{nc} \in \mathbb{N}_+$; $V_c \in \mathbb{N}_+$; CCQ k-ary ($k \geq 1$) query $Q(\bar{Y}) \leftarrow L, M$ with set P of $p \geq 0$ distinct constants.) (**Output:** Constant value c and database D^t_{Q,V_{nc},V_c}, such that k-ary tuple $\bar{d} = (c, c, \ldots, c)$ has multiplicity *exactly* $(max(V_{nc}, p))^{|M_{noncopy}|} \times V_c^{|M_{copy}|}$ in $Res_C(Q, D^t_{Q,V_{nc},V_c})$. Here, the value of c in $\bar{d}$ (i) is the constant c_0 in case c_0 is the only term occurring in the head vector $\bar{Y}$ in Q,[11] (ii) is otherwise 1 in case where either $V_{nc} > p$ or 1 is a constant in Q, and (iii) is otherwise an arbitrary constant c_1 of Q.) Throughout the construction, for the case $M \neq \emptyset$ we assume w.l.o.g. that the $|M| = n \geq 1$ multiset variables of the query Q have names $X_1, X_2, \ldots, X_n$, and that the first m of the variables in the sequence $X_1, X_2, \ldots, X_n$, with $0 \leq m \leq n$, are all the copy variables (forming the set M_{copy}) of Q.

Step 1. In case $p \geq V_{nc}$, let the set $adom(D^t_{Q,V_{nc},V_c})$ be the set P of distinct constants in Q; otherwise add to P enough constants from the list $[1, 2, \ldots, V_{nc}]$ (starting from the beginning of the list) to form a set $adom(D^t_{Q,V_{nc},V_c})$ of size V_{nc} . Let $Asize := |adom(D^t_{Q,V_{nc},V_c})|$; clearly, $Asize = max(V_{nc}, p)$. (In the special case $n = 0$, the *effective* active domain of D^t_{Q,V_{nc},V_c} is of size $max(1, p)$ and is (i) the set P if $p > 0$, or is (ii) the set $\{1\}$ otherwise. The reason for this restriction of the effective active domain of the database in case $n = 0$ is that the values from $\{2, \ldots, V_{nc}\}$, if they are not used as constants of Q but are used in the above construction of $adom(D^t_{Q,V_{nc},V_c})$, are used in this construction to "create values" of only multiset variables of the query Q, and thus are not used in case $M = \emptyset$, that is $n = 0$.)

If some constant c_0 is the only term in the head vector $\bar{Y}$ of the query Q, then set c to c_0. Otherwise, if the set $adom(D^t_{Q,V_{nc},V_c})$ includes the constant 1, then set the value of c to 1. Otherwise set $c := c_1$, where c_1 is an arbitrary constant in $adom(D^t_{Q,V_{nc},V_c})$.

In case $n > m$ we use an auxiliary predicate symbol r_1^* of arity $(n-m)$, such that r_1^* is not used in the query Q. In case $m > 0$ we use an m-ary predicate symbol r_2^* (distinct from r_1^*), also not used in the query Q. Finally, in case $n > 0$ we use an auxiliary n-ary predicate symbol r^* (distinct from r_1^* and r_2^*) not used in Q. We assume an infinite domain *Attr* of attribute names, and (i) label the attributes of r_2^* (in case $m > 0$) with m attribute names $B_1, B_2, \ldots, B_m$, (ii) label the attributes of r_1^* (in case $n > m$) with $n - m$ attribute names $B_{m+1}, B_{m+2}, \ldots, B_n$, as well as (iii) label the attributes of r^* (in case $n > 0$) with n attribute names $B_1, B_2, \ldots, B_n$ (same as in (i), (ii)). Here, in case $n > 0$ we have $B_l \in Attr$ for each $l \in \{1, \ldots, n\}$, and the attribute names B_i and B_j are distinct for each pair (i, j), $1 \leq i \neq j \leq n$.

In case $n > 0$, we build a relation R^* with schema r^* and with $Asize^{(n-m)} \times V_c^m$ tuples, as follows:

- In case $n > m$, populate relation R_1^*, with schema r_1^*, by taking all combinations of the values from the set $adom(D^t_{Q,V_{nc},V_c})$ as tuples of arity $(n - m)$ in R_1^*. As a result, R_1^* will have $Asize^{(n-m)}$ tuples.
- In case $m > 0$, populate relation R_2^*, with schema r_2^*, by taking all combinations of the values from the set $\{1, \ldots, V_c\}$ as m-tuples of R_2^*. As a result, R_2^* will have V_c^m tuples.
- (1) In case $0 < m < n$, populate R^* with all the tuples from the Cartesian product $R_2^* \times R_1^*$;

  (2) Otherwise in case $m = 0$, populate R^* with all the tuples from R_1^*;

  (3) Otherwise in case $m = n$, populate R^* with all the tuples from R_2^*.

Finally, we associate each attribute B_i of R^*, $1 \leq i \leq n$, with the multiset variable X_i of Q.

Step 2. We now construct the database D^t_{Q,V_{nc},V_c} . In case $n > 0$ we interpret the $T = Asize^{(n-m)} \times V_c^m$ tuples of the auxiliary relation R^* as T substitutions for the head and body variables of the query Q, such that for each substitution θ_j, $j \in \{1, \ldots, T\}$ and $\theta_j \in \Gamma(Q, D^t_{Q,V_{nc},V_c})$, θ_j uses only the values from the set $adom(D^t_{Q,V_{nc},V_c})$ as values for the multiset noncopy variables of Q, and uses only the values between 1 and V_c as values for the copy variables of Q. We *construct* D^t_{Q,V_{nc},V_c} using this interpretation. By our construction, each θ_j will result in a distinct copy of the k-tuple $\bar{d} = (c, c, \ldots, c)$ in $Res_C(Q, D^t_{Q,V_{nc},V_c})$. In case $n = 0$ we create D^t_{Q,V_{nc},V_c} (with effective active domain of size $max(1, p)$) using a single substitution θ_1, which guarantees that the tuple $\bar{d} = (c, c, \ldots, c)$ is in $Res_C(Q, D^t_{Q,V_{nc},V_c})$.

We now assume that the database D^t_{Q,V_{nc},V_c} is initially empty. For the case $n = 0$, set $T := 1$. We create the

[10] Note that for the case $M = \emptyset$, i.e., Q is a set query, the result is as expected by set semantics for query evaluation.

[11] This is the case where the head $Q(\bar{Y})$ of Q has no variables and uses only one constant, c_0.

T substitutions θ_j as follows: For each $j \in \{1, \ldots, T\}$ and assuming $n > 0$, let $r^*(c_{j1}, c_{j2}, \ldots, c_{jn})$ be the jth row of the relation R^*. (We assume an arbitrary fixed ordering on the T tuples of R^*.) Let $\mathcal{X} = \{X_1, X_2, \ldots, X_n, Z_1, Z_2, \ldots, Z_l\}$, with $n \geq 0$ and $l \geq 0$, be the set of all variables of the query Q. Recall that in case that Q has $m > 0$ copy variables, $X_1, \ldots, X_m$ in the set $\mathcal{X}$ are all the copy variables of Q. We obtain θ_j (for $n \geq 0$) by (1) assigning the constant c to each variable in the set $\{Z_1, \ldots, Z_l\}$, and by (2) assigning (in case $n > 0$ *only*) the constant c_{ji} to the variable X_i, for $i \in \{1, \ldots, n\}$. In addition, we set $\theta_j(g) = g$ for each constant g ($g \in P$) used in query Q.

Now, for each $j \in \{1, \ldots, T\}$, for the θ_j constructed as above, and for each subgoal of the query Q, we add to the database D^t_{Q,V_{nc},V_c} at most one ground tuple (a new tuple is added exactly when *no* identical tuple is already present in D^t_{Q,V_{nc},V_c}), as follows:

- Let $p(\bar{Z})$ be a relational subgoal of the query Q. Then add to D^t_{Q,V_{nc},V_c} the ground tuple $p(\theta_j(\bar{Z}); V_c)$ (if not already present in the database). That is, we add to D^t_{Q,V_{nc},V_c} the ground atom $p(\theta_j(\bar{Z}))$ with multiplicity V_c .
- Let $p(\bar{Z}; W)$ be a copy-sensitive subgoal of the query Q, with copy variable W. Then add to D^t_{Q,V_{nc},V_c} the ground tuple $p(\theta_j(\bar{Z}); V_c)$ (if not already present in the database).

Step 3. Return the constant c and database D^t_{Q,V_{nc},V_c} .

Given a CCQ query Q with $p \geq 0$ distinct constants and given a positive integer number A, we use the procedure Template-Boxed-Database to construct the "boxed database" $D^b_{Q,A}$ for Q and A, by setting $V_{nc} := A$ and $V_c := max(A, p)$ in the procedure inputs. By construction, $adom(D^b_{Q,A})$ is of size $max(A, p)$.

We also do a minor straightforward modification of the procedure Template-Boxed-Database to construct a "boxed copy-variable database", denoted $D^{copy}_{Q,A}$, for an input CCQ query Q that has $p \geq 0$ distinct constants and $n \geq 1$ *copy* variables (Q can also have an arbitrary number of other multiset variables), and for a positive integer A. The construction is as above, except that $X_1, \ldots, X_n$ are (w.l.o.g.) the *copy* variables of the query Q, and in the procedure Template-Boxed-Database we (i) set $m := n$, (ii) set the procedure input V_c to value A, and (iii) set the procedure input V_{nc} to 1. By construction, $adom(D^{copy}_{Q,A})$ is of size $max(1, p)$.

Consider an illustration of the construction of databases $D^b_{Q,A}$ and $D^{copy}_{Q,A}$.

EXAMPLE E.1. *Let CCQ queries Q and Q' be as follows.*

$$Q(B) \leftarrow p(B, C; i),\ p(B, D),\ \{C, D, i\}.$$
$$Q'(E) \leftarrow p(E, F; j),\ p(E, G),\ p(E, c_1; l),\ p(E, c_2),\ \{F, G, j, l\}.$$

Here, c_1 and c_2 (in Q') are constants. The rest of the terms mentioned in Q and Q' are variables, with i, j, l being copy variables.

Let $A_1 = 3$ and $A_2 = 1$. Then:

- $D^b_{Q,3} = \{\!\{p(1, 1; 3), p(1, 2; 3), p(1, 3; 3)\}\!\}$*, and tuple* (1) *has multiplicity* $3^3 = 27$ *in* $Res_C(Q, D^b_{Q,3})$.
- $D^{copy}_{Q,3} = \{\!\{p(1, 1; 3)\}\!\}$*, and tuple* (1) *has multiplicity* $3^1 = 3$ *in* $Res_C(Q, D^{copy}_{Q,3})$.
- $D^b_{Q,1} = D^{copy}_{Q,1} = \{\!\{p(1, 1; 1)\}\!\}$ *, and tuple* (1) *has multiplicity* 1 *in both* $Res_C(Q, D^b_{Q,1})$ *and* $Res_C(Q, D^{copy}_{Q,1})$.
- $D^b_{Q',3} = \{\!\{p(1, 1; 3), p(1, c_1; 3), p(1, c_2; 3)\}\!\}$*, and tuple* (1) *has multiplicity* $3^4 = 81$ *in* $Res_C(Q', D^b_{Q',3})$.
- $D^{copy}_{Q',3} = \{\!\{p(c_1, c_1; 3), p(c_1, c_2; 3)\}\!\}$*, and tuple* (c_1) *has multiplicity* $2^2 \times 3^2 = 36$ *in* $Res_C(Q', D^{copy}_{Q',3})$.
- $D^b_{Q',1} = D^{copy}_{Q',1} = \{\!\{p(c_1, c_1; 1), p(c_1, c_2; 1)\}\!\}$*, and tuple* (c_1) *has multiplicity* $2^2 = 4$ *in both* $Res_C(Q', D^b_{Q',1})$ *and* $Res_C(Q', D^{copy}_{Q',1})$.

□

E.2 Properties of boxed databases

We now formulate some useful properties of our boxed databases. First we make a straightforward observation about multiplicities of tuples in answers to CCQ queries on arbitrary databases.

PROPOSITION E.1. *Let $Q(\bar{X}) \leftarrow L, M$ be a CCQ query with set P of $p \geq 0$ distinct constants. Let D be a database, with $A_c \in \mathbb{N}_+$ being the maximal copy number for the atoms in D and with active domain $adom(D)$ of size $A_{nc} \in \mathbb{N}_+$. Then (i) for each tuple $t \in Res_C(Q, D)$, t occurs in $Res_C(Q, D)$ with multiplicity at most $A_c^{|M_{copy}|} \times A_{nc}^{|M_{noncopy}|}$, and (ii) in case $P \subseteq adom(D)$ does* not *hold, we have that $Res_C(Q, D)$ is empty.* □

The proof follows immediately from the definition of combined query semantics, specifically from the construction of the set $\Gamma_{\bar{S}}(Q, D)$.

We use Proposition E.1 in showing the following result about boxed databases.

PROPOSITION E.2. *Let CCQ query $Q(\bar{Y}) \leftarrow L, M$, with $p \geq 0$ distinct constants, have head vector $\bar{Y}$ of length $k > 0$, and let $A \in \mathbb{N}_+$. Then there exists a value c in $adom(D^b_{Q,A})$ such that the k-tuple $\bar{d} = (c, \ldots, c)$ occurs in the bag $Res_C(Q, D^b_{Q,A})$ with multiplicity* exactly $(max(A, p))^{|M|}$. □

PROOF. (Proposition E.2) First of all, by construction of $D^b_{Q,A}$, we have that $|adom(D^b_{Q,A})| = max(A, p)$, and that the maximal copy number for the atoms in $D^b_{Q,A}$ is also $max(A, p)$. Let the value c be determined as the c in the construction of the database $D^b_{Q,A}$.

For the lower bound, observe that by construction of the database $D^b_{Q,A}$, the set $\Gamma_{\bar{S}}(Q, D^b_{Q,A})$ has at least $(max(A, p))^{|M|}$ distinct tuples, each of which has the value c of each of the head variables of the query Q. Indeed, recall the substitutions θ_j, $j \in \{1, \ldots, (max(A, p))^{|M|}\}$, used in the construction of the database $D^b_{Q,A}$. For each $i, j \in \{1, \ldots, (max(A, p))^{|M|}\}$, such that $i \neq j$, θ_i and θ_j differ on the value of at least one multiset variable of the query Q, and the values of all the multiset variables of Q are retained in the tuples in the set $\Gamma_{\bar{S}}(Q, D^b_{Q,A})$. (The satisfaction by each θ_j of the copy-sensitive subgoals of Q, if any, is ensured by the fact that each ground atom in $D^b_{Q,A}$ has multiplicity $max(A, p)$; recall that each θ_j assigns to each copy variable a positive integer value not exceeding $max(A, p)$. Observe also that in mapping each relational atom of Q to a ground atom by each θ_j, the multiplicity value of that ground atom is ignored, again by the rules of the combined query semantics.)

For the upper bound, by Proposition E.1 it holds that the multiplicity of the tuple $\bar{d} = (c, \ldots, c)$ in the bag $Res_C(Q, D^b_{Q,A})$ is at most $(max(A, p))^{|M|}$. $\Box$

PROPOSITION E.3. *Let CCQ query $Q(\bar{Y}) \leftarrow L, M$, with $m \geq 0$ copy variables and with $p \geq 0$ constants, have head vector $\bar{Y}$ of length $k > 0$, and let $A \in \mathbb{N}_+$. Then there exists a value c in $adom(D^{copy}_{Q,A})$ such that the k-tuple $\bar{d} = (c, \ldots, c)$ occurs in the bag $Res_C(Q, D^{copy}_{Q,A})$ with multiplicity at least A^m and at most $(max(1, p))^{(|M|-m)} \times A^m$.* $\Box$

The proof of Proposition E.3 is a straightforward modification of the proof of Proposition E.2, where we use the properties of the database $D^{copy}_{Q,A}$ (rather than of $D^b_{Q,A}$). Note that in case $p > 1$ we cannot guarantee an exact numerical multiplicity of tuple $\bar{d} = (c, \ldots, c)$ in $Res_C(Q, D^{copy}_{Q,A})$, because the construction of $D^{copy}_{Q,A}$ includes all the constants of the query Q in the active domain of the database. (This inclusion is required, recall item (ii) in Proposition E.1.) As an illustration, consider the databases $D^{copy}_{Q',3}$ and $D^{copy}_{Q',1}$ in Example E.1.

E.3 Proof of Theorem 3.1

We now prove Theorem 3.1. The query-equivalence corollary of this result (see Corollary 3.1) extends Lemma 5.1 of [6], for CCQ queries, from queries that may have only relational subgoals and may not have constants, to the case of general CCQ queries.[12]

LEMMA E.1. *(Lemma 5.1 of [6]: Number of multiset variables) Let $Q(\bar{X}) \leftarrow L, M$ and $Q'(\bar{X}') \leftarrow L', M'$ be copy-insensitive relational queries. If $Q \equiv_C Q'$ then $|M| = |M'|$.* $\Box$

PROOF. Suppose, by way of contradiction, that Q and Q' are copy-insensitive, relational and equivalent, but $|M| \neq |M'|$. Suppose, without loss of generality, that $|M| > |M'|$. Let G be a set of $|M|$ constants. Let Γ be the set of all possible assignments of the variables in Q to the constants in G. Let D be the database that contains all images of the body of Q with respect to the assignments of Γ, i.e., $D = \cup_{\gamma \in \Gamma} \gamma(L)$. (Note that we view L as a set of conditions, in this notation.)

Now, consider a tuple $\bar{d}$. Observe that $Res_C(Q, D)$ returns $\bar{d}$ a total of $|M|^{|M|}$ times, since every assignment of the multiset variables to constants in G is satisfiably extendible. On the other hand, $Res_C(Q', D)$ can contain at most $|M'|^{|M|}$ copies of $\bar{d}$, since Q' has only M' multiset variables. Since $|M'|^{|M|} < |M|^{|M|}$, it follows that $Q \not\equiv_C Q'$. $\Box$

The proof of Theorem 3.1 is by contradiction and is constructive. That is, the proof shows noncontainment of Q in Q' by producing a counterexample database, for (a) the case where the number of *copy* variables in Q is greater than the number of copy variables in Q', and for (b) the case where the number of multiset *noncopy* variables in Q is greater than the number of multiset noncopy variables in Q'.

PROOF. (Theorem 3.1) Suppose that $Q \sqsubseteq_C Q'$ and that Q has $p \geq 0$ distinct constants, whereas Q' has $p' \geq 0$ distinct constants. We first show that $Q \sqsubseteq_C Q'$ implies $|M_{copy}| \leq |M'_{copy}|$. The implication clearly holds in case $M_{copy} = \emptyset$. Now assume that $|M_{copy}|$ is a positive natural number m, $|M_{copy}| = m > 0$. Further, assume toward a contradiction that $m = |M_{copy}| > |M'_{copy}|$. We choose a natural number $A > 1$ in such a way that A is also greater than $(max(1, p))^{|M'|-|M'_{copy}|}$. Using this value of A, we construct the boxed copy-variable database $D^{copy}_{Q,A}$ for the query Q. By Proposition E.3, there is a value c in $adom(D^{copy}_{Q,A})$ such that the k-tuple $\bar{d} = (c, \ldots, c)$ occurs in the bag $Res_C(Q, D^{copy}_{Q,A})$ with multiplicity at least A^m. Now by Proposition E.1, for each tuple t (if any) in $Res_C(Q', D^{copy}_{Q,A})$ (for the query Q' on the same database), the multiplicity of t in $Res_C(Q', D^{copy}_{Q,A})$ cannot exceed $A^{|M'_{copy}|} \times (max(1, p))^{|M'|-|M'_{copy}|}$. (Recall that by construction of $D^{copy}_{Q,A}$, it holds that $|adom(D^{copy}_{Q,A})| = max(1, p)$.) Using our assumptions that $m = |M_{copy}| > |M'_{copy}|$ and that $A > (max\ (1, p))^{|M'|-|M'_{copy}|}$, we conclude that no tuple in $Res_C(Q', D^{copy}_{Q,A})$ has the same multiplicity as the multiplicity of the tuple $\bar{d}$ in $Res_C(Q, D^{copy}_{Q,A})$, and thus $D^{copy}_{Q,A}$ is a counterexample database to $Q \sqsubseteq_C Q'$.

Finally, we show that $Q \sqsubseteq_C Q'$ implies $|M_{noncopy}| \leq |M'_{noncopy}|$. Recall that we already know that $|M_{copy}| \leq |M'_{copy}|$ whenever $Q \sqsubseteq_C Q'$. Recall that $M_{noncopy}$ is defined using set difference as $M_{noncopy} = M - M_{copy}$. Similarly, $M'_{noncopy} = M' - M'_{copy}$. Now assume toward a contradiction that $n = |M_{noncopy}| > |M'_{noncopy}|$. We construct a variation $\tilde{D}^b_{Q,n}$ on the boxed database for the query Q and for the fixed positive integer value $n = |M_{noncopy}|$; in constructing $\tilde{D}^b_{Q,n}$, the only change from the construction of $D^b_{Q,n}$ is that $\tilde{D}^b_{Q,n}$ is the core-set of the bag $D^b_{Q,n}$. That is, while each tuple in $D^b_{Q,n}$ has multiplicity $max(n, p)$ by construction, $\tilde{D}^b_{Q,n}$ has the same (distinct) tuples as $D^b_{Q,n}$ does, but $\tilde{D}^b_{Q,n}$ has only one copy of each of its tuples. By following the proof of Proposition E.2 for the variation $\tilde{D}^b_{Q,n}$ on $D^b_{Q,n}$, we obtain that the k-tuple $\bar{d} = (c, \ldots, c)$ occurs in the bag $Res_C(Q, \tilde{D}^b_{Q,n})$ with multiplicity $(max(n, p))^n$. Now it is immediate from Proposition E.1 that no tuple in $Res_C(Q', \tilde{D}^b_{Q,n})$, for the query Q', may occur with multiplicity greater than $(max(n, p))^{|M'_{noncopy}|}$. Using our assumption that $|M'_{noncopy}| < n$, we arrive at the desired contradiction by concluding that Q is not contained in Q' under the combined semantics. $\Box$

F. FORMALIZING THE NOTIONS OF Q_{CE} AND Q'_{CE}

In this section we formalize the notions of Q_{ce} and Q'_{ce}, for CCQ queries Q and Q' such that Q' is a reduced-condition query for Q. The definition of Q_{ce} is the same as that of Q_{+copy} in [6]. We use the definitions of of Q_{ce} and Q'_{ce}, instead of using Q_{+copy}, because in some of the proofs in this paper, specifically in those of Section 5, we use the convenient similarities ("overlaps") between the syntax of the query Q_{ce} and the syntax of the query Q'_{ce}, for queries Q and Q' such that Q' is a reduced-condition query for Q.

For a CCQ query Q, we call CCQ query Q_{ce} a *copy-enhanced version of* Q if Q_{ce} is the result of adding a distinct copy variable to each relational subgoal of Q. (We can show that for a query Q, all copy-enhanced versions of Q are iden-

[12]The coverage of Lemma 5.1 of [6] is CQ queries under combined semantics, with multiset variables, disjunction, and negation, but without constants or copy variables.

tical up to renaming of the copy variables introduced in the construction of Q_{ce}.) Further, for each CCQ query Q' that is a reduced-condition query for CCQ query Q, we obtain the query Q'_{ce} by removing from Q_{ce} those subgoals (if any) that do not correspond to the subgoals of Q'. That is:

- For each copy-sensitive subgoal s of Q such that s is not a subgoal of Q', we have that s is not a subgoal of Q'_{ce};
- For each relational subgoal s of Q: Let Q have exactly $n \geq 1$ copies of atom s, and let Q' have exactly k copies of atom s, with $0 \leq k < n$. Let M_s be the set of n copy variables that have been introduced to represent the n copies of atom s in the query Q_{ce}. Let M'_s be an arbitrary subset of M_s, of cardinality $|M'_s| = k$. Then:
  - The condition L'_{ce} of query Q'_{ce} has all those subgoals s' of the condition L_{ce} of query Q_{ce} such that the copy variable of s' is an element of M'_s, and
  - L'_{ce} does not have any subgoals s' of L_{ce} such that the copy variable of s' is an element of $M_s - M'_s$ (as set difference).
- Finally, for each subgoal s of Q_{ce} such that s is not covered by the preceding conditions, s is also a subgoal of Q'_{ce}.

Consider an illustration.

EXAMPLE F.1. *For the following queries Q and Q' we have that Q' is a reduced-condition query for Q:*

$$Q(X) \leftarrow p(X, Y, Z; i),\ p(X, W, Y),\ p(W, Y, Z),\ \{Y, i\}.$$
$$Q'(X) \leftarrow p(X, Y, Z; i),\ p(X, W, Y),\ \{Y, i\}.$$

Then the queries Q_{ce} and Q'_{ce} are defined as

$$Q_{ce}(X) \leftarrow p(X, Y, Z; i),\ p(X, W, Y; j),\ p(W, Y, Z; k),\ \{Y, i, j, k\}.$$
$$Q'_{ce}(X) \leftarrow p(X, Y, Z; i),\ p(X, W, Y; j),\ \{Y, i, j\}.$$

Note that the extra copy variable added to subgoal $p(X, W, Y)$ has the same name, j, in both Q_{ce} and Q'_{ce}. □

G. PROOF OF THEOREM 3.2

In this section we provide and illustrate a proof of Theorem 3.2. In the proof we will use the notion of an "extended canonical database" of a CCQ query.

G.1 Extended canonical database of CCQ query

Set queries. We first recall the notion of a "canonical database" of a CCQ *set* query. Every CCQ set query Q can be regarded as a symbolic database $D^{(Q)}$. $D^{(Q)}$ is defined as the result of turning each subgoal $p_i(\ldots)$ of Q into a tuple in the relation P_i that corresponds to predicate p_i. The procedure is to keep each constant in the body of Q, and to replace consistently each variable in the body of Q by a distinct constant different from all constants in Q. The tuples that correspond to the resulting ground atoms are the only tuples in the *canonical database* $D^{(Q)}$ for Q, which (database) is unique up to isomorphism.

General CCQ queries. We now extend the above notion, to define an *extended canonical database* for a general (i.e., not necessarily set) CCQ query Q. We first partition all the subgoals of Q into equivalence classes $\mathcal{C}_1^{(Q)}, \ldots, \mathcal{C}_k^{(Q)}$, $k \geq 1$, where two subgoals of Q belong to the same class if and only if the subgoals have the same relational template. We then choose one representative element, $c_j^{(Q)}$, of each class $\mathcal{C}_j^{(Q)}$, $j \in \{1, \ldots, k\}$; if $\mathcal{C}_j^{(Q)}$ has at least one copy-sensitive atom then $c_j^{(Q)}$ must be a copy-sensitive atom. Finally, an extended canonical database for the query Q is constructed from the subgoals $c_1^{(Q)}, \ldots, c_k^{(Q)}$ of Q in the same way as the "standard" canonical database is constructed from the condition of a CCQ set query. The only difference is in that whenever $c_j^{(Q)}$ is a copy-sensitive atom, the copy variable of the atom must be replaced by a natural number that is distinct from all the other constants in the database (both in the active domain of the database and among the copy numbers of all ground atoms). The above mapping of terms of Q to the constants in the database, such that the mapping is used to generate the database, can be used to define in a natural way an assignment mapping from the query Q to the database. In defining that assignment mapping, we accept the convention that the mapping maps each copy variable of the query to the constant 1. We call that assignment mapping the *generative mapping* for the query and for the database; observe that, by definition, the generative mapping is always a *valid* (i.e., satisfying) assignment mapping from *all subgoals* of the query Q to the extended canonical database for Q. We can show that for each CCQ query, its extended canonical database is unique up to isomorphism.

For instance, an extended canonical database of the query Q' of Example 4.1 is $\{r(a, b, c, d; 2)\}$. The query generates exactly one equivalence class $\mathcal{C}_1^{(Q')}$, due to the fact that the two subgoals of the query Q' have the same relational template. We choose arbitrarily the atom $c_1^{(Q')}$ to be the first subgoal of the query Q'. The generative mapping in this case is $\{\ X_1 \rightarrow a,\ Y_1 \rightarrow b,\ Y_2 \rightarrow c,\ X_2 \rightarrow d,\ Y_3 \rightarrow 1,\ Y_4 \rightarrow 1\ \}$.

G.2 Proof and illustration

We now prove and illustrate Theorem 3.2.

PROOF. (sketch) Assume that CCQ queries Q and Q' are such that $Q \sqsubseteq_C Q'$ holds. We construct an extended canonical database, call it $D_c^{(Q)}$, for the query Q; denote by $c_1^{(Q)}$, $c_2^{(Q)}, \ldots, c_k^{(Q)}$, with $k \geq 1$, the set of those subgoals of Q that were used to construct the database $D_c^{(Q)}$. (See the definition of extended canonical database for the details on the subgoals $c_1^{(Q)}, c_2^{(Q)}, \ldots, c_k^{(Q)}$ of the query Q.) Further, denote by μ the generative mapping from Q to the database $D_c^{(Q)}$. Now obtain a database D, by setting each copy number in the database $D_c^{(Q)}$ to the constant 1. By definition, μ is a satisfying assignment from the query Q to the database D. Thus, clearly, the bag of answers to the query Q on the database D has the tuple $t^* = \mu(\bar{X})$, where $\bar{X}$ is the head vector of the query Q.

Further, when we restrict the domain of μ to all the terms of the query Q except its copy variables, call the resulting mapping μ', then μ' induces a bijection from the set of subgoals $c_1^{(Q)}, c_2^{(Q)}, \ldots, c_k^{(Q)}$ of the query Q onto the set of atoms in the database D. (This holds by construction of the database D.)

By $Q \sqsubseteq_C Q'$, there must exist a satisfying assignment, ν, from the query Q' to the database D, such that the restriction of ν to the head vector of Q' is the tuple t^* as constructed above. Consider a mapping $\nu' = (\mu')^{-1} \circ \nu$; it is well defined by construction. Clearly, one can construct easily an extension, ν'', of ν' to all the terms of the query Q'_{ce}, such that ν'' is a GCM from the query Q'_{ce} to the query Q_{ce}. (The whole point of introducing copy-enhanced versions of CCQ queries is to enable mapping any subgoal of one query to any subgoal of the other query. That is, in the copy-enhanced version we would never fail to construct a GCM just because a subgoal, s, of the "source" query fails to map to some subgoal, s', of the "target" query due to incompatibility at the level of copy variables. Such incompatibility would be either s being a relational atom – that is, s not having a copy variable – and s' being a copy-sensitive atom, or vice versa.) The extension ν'' of ν' would induce *the same* mapping as ν' at the subgoal level of the two queries, modulo the copy variables added when transforming query Q (Q', respectively) into Q_{ce} (into Q'_{ce}, respectively). Q.E.D. $\square$

Consider an illustration of the proof of Theorem 3.2.

EXAMPLE G.1. *Let CCQ queries Q and Q' be as follows.*

$Q(X) \leftarrow p(X,Y,Z;i),\ p(X,T,Y),\ p(X,Y,W),\ \{Y,i\}.$
$Q'(X) \leftarrow p(X,Y,Z;i),\ p(X,T,Y),\ \{Y,i\}.$

Query Q' is a proper reduced-condition query for Q. By our results in Section 5, we have that $Q \equiv_C Q'$. (In fact, Q' is a minimized version of Q.) By definition of combined-semantics equivalence, we have that $Q' \sqsubseteq_C Q$. We show that there exists a GCM from Q_{ce} to Q'_{ce}.

For the query Q', a database D as in this proof could be D = $\{\ p(1,2,3),\ p(1,4,2)\ \}$. (As the two subgoals of the query Q' have different relational templates, both subgoals of Q' would participate in creating atoms in the extended canonical database of Q' and hence in the above database D.) The generative mapping μ from Q onto D would then be $\{\ X \rightarrow 1,\ Y \rightarrow 2,\ Z \rightarrow 3,\ T \rightarrow 4,\ i \rightarrow 1\ \}$. We denote by t^* the tuple $\mu(X) = (1)$. The mapping μ', which is the result of restricting the domain of μ to all terms of Q' except its copy variables, is $\mu' = \{\ X \rightarrow 1,\ Y \rightarrow 2,\ Z \rightarrow 3,\ T \rightarrow 4\ \}$. Observe that μ' induces a bijection from the subgoals of the query Q' onto the database D: The first (second, respectively) subgoal of Q' gets mapped by μ' into the first (second, respectively) atom of D.

There does exist an assignment mapping ν from the query Q to the database D such that $\nu(X) = t^*$. We have that ν = $\{\ X \rightarrow 1,\ Y \rightarrow 2,\ Z \rightarrow 3,\ T \rightarrow 4,\ W \rightarrow 3,\ i \rightarrow 1\ \}$.

Consider the mapping $(\mu')^{-1} \circ \nu$, which we define on all the terms of the query Q except its copy variables. It is easy to see that $\nu' = (\mu')^{-1} \circ \nu$ is a well defined mapping. We have that $\nu' = \{\ X \rightarrow X,\ Y \rightarrow Y,\ Z \rightarrow Z,\ T \rightarrow T,\ W \rightarrow Z\ \}$. Observe that the mapping ν' associates subgoals of the queries Q and Q' as follows: $(p(X,Y,Z;i), p(X,Y,Z;i))$ (in each pair in this sentence, the first atom of the pair is a subgoal of the query Q, and the second atom – a subgoal of the query Q'), $(p(X,T,Y), p(X,T,Y))$, and $(p(X,Y,W), p(X,Y,Z;i))$.

Now the queries Q_{ce} and Q'_{ce} are as follows.

$Q_{ce}(X) \leftarrow p(X,Y,Z;i),\ p(X,T,Y;j),\ p(X,Y,W;k),$
$\quad \{Y,i,j,k\}.$
$Q'_{ce}(X) \leftarrow p(X,Y,Z;i),\ p(X,T,Y;j),\ \{Y,i,j\}.$

It is easy to see that the desired GCM ν'' is an extension of ν' to the copy variables of the query Q_{ce}, such that the extension preserves the association induced by ν' between the subgoals of the queries Q and Q', modulo the copy variables added when transforming query Q (Q', respectively) into Q_{ce} (into Q'_{ce}, respectively). Specifically, $\nu'' = \{\ X \rightarrow X,\ Y \rightarrow Y,\ Z \rightarrow Z,\ T \rightarrow T,\ W \rightarrow Z,\ i \rightarrow i,\ j \rightarrow j,\ k \rightarrow i\ \}$. $\square$

H. PROPERTIES OF CVM MAPPINGS

H.1 (Un)regularizing CCQ queries

In Definition 3.2 in Section 3.2 we defined the "regularized," "deregularized," and "unregularized" formats for CCQ queries. Given a CCQ query Q, the three equivalent formats for the definition of Q can be obtained easily from one another, and differ only in the amount of syntactic sugar that they provide. Specifically, the regularized format has no syntactic sugar, and the deregularized format has the most syntactic sugar while having no duplicate subgoals. The importance of these formats for query minimization is as follows: Whenever, for a given CCQ query, one of these formats is a minimized query, then only the regularized version is the minimized query.

In Section 3.2 we introduce "covering mappings" *(CVMs)* between CCQ queries, with the following property: If a CVM exists from query Q to query Q', then the CVM can be discovered regardless of the amount of the syntactic sugar present in the definitions of the two queries, *and* the CVM is a generalized containment mapping from (any version of) the source query to the deregularized version of the target query. (See Proposition 3.2 in Section 3.2.)

For a CCQ query Q, we have that the regularized version of Q, of Q_r, of Q_d, and of each Q_u (i) is Q_r, and (ii) can be obtained by dropping some subgoals of the query in question. We say that Q, Q_r, Q_d, and each Q_u are all *the same query up to regularization.*

EXAMPLE H.1. *Consider CCQ query Q.*

$Q(X) \leftarrow p(X,Y,Z;i),\ p(X,W,Y;j),\ p(W,Y,Z),$
$\quad p(W,Y,Z),\ p(X,Y,Z),\ \{Y,i,j\}.$

This query is represented by one of an infinite number of its unregularized versions. The set $\mathcal{T}(Q)$ for Q is $\mathcal{T}(Q) = \{\ p(X,Y,Z),\ p(X,W,Y)\ \}$. Thus, the queries Q_c, Q_r, and Q_d are defined as

$Q_c(X) \leftarrow p(X,Y,Z;i),\ p(X,W,Y;j),\ p(W,Y,Z),$
$\quad p(X,Y,Z),\ \{Y,i,j\}.$
$Q_r(X) \leftarrow p(X,Y,Z;i),\ p(X,W,Y;j),\ p(W,Y,Z),\ \{Y,i,j\}.$
$Q_d(X) \leftarrow p(X,Y,Z;i),\ p(X,W,Y;j),\ p(W,Y,Z),$
$\quad p(X,Y,Z),\ p(X,W,Y),\ \{Y,i,j\}.$ $\square$

H.2 Each CVM can be viewed as a GCM

In this subsection we prove Proposition 3.2. (We reformulate the Proposition here by swapping the notation for Q and Q' that was used in Section 3.2.)

PROPOSITION 3.2. *Given CCQ queries Q and Q'. Then for each CVM, μ, from Q' to Q, we have that (1) μ is a* GCM *from Q' to the deregularized version of Q, and (2) μ is a CVM from Q' to the regularized version of Q.*

PROOF. Proof of (1): Let μ be a CVM from CCQ query Q' to CCQ query Q. We apply the mapping μ to the head and individually to each subgoal of the query Q'; we will refer to the result as (query) $\mu(Q')$. In addition, we impose a natural requirement that a variable Y in the query $\mu(Q')$ is a multiset variable of $\mu(Q')$ if and only if Y is a multiset variable of the query Q. It is immediate from this requirement and from item (3) of Definition 3.1 that the multiset variables of $\mu(Q')$ are exactly the multiset variables of the query Q.

Now applying μ to the head vector of the query Q' results in the head vector of Q, by item (2) of Definition 3.1. Further, for each copy-sensitive subgoal of the query Q', its image in $\mu(Q')$ is a copy-sensitive subgoal of the query Q, by item (5) of Definition 3.1. We get a similar desired behavior, by item (4) of Definition 3.1, for all relational subgoals of Q' whose images under μ are *relational* subgoals of the query Q.

The only problem in the application of the mapping μ to the query Q' would arise when, for some relational subgoal of Q' of the form $s(\bar{Y})$, the *relational* atom $s(\mu(\bar{Y}))$ is not a subgoal of the query Q. For an illustration, consider queries Q and Q' of Example 3.2: The mapping $\mu = \{X \rightarrow X, Y \rightarrow Y, Z \rightarrow X, i \rightarrow i\}$ is a CVM from query Q to query Q' of the Example. Applying this mapping to subgoal $p(X, Z, Y)$ of the query Q results in a relational subgoal $p(X, X, Y)$ that is not present in the query Q'.

However, items (3) through (5) of Definition 3.1 together ensure that for each occurrence of the above problem, query $\mu(Q')$ has "the copy-sensitive version", $s(\mu(\bar{Y}); i)$ (for some copy variable $i \in M_{copy}$), of the atom $s(\mu(\bar{Y}))$ of the previous paragraph. That subgoal $s(\mu(\bar{Y}); i)$ would be added to $\mu(Q')$ as the result of applying μ to some copy-sensitive subgoal of the query Q'. Thus, adding the *relational* atom $s(\mu(\bar{Y}))$ to $\mu(Q')$, as the result of applying μ to the *relational* subgoal $s(\bar{Y})$ of the query Q', does not "take us outside" the set of subgoals of the *deregularized version* of the query Q. (Recall the definition of the set $\mathcal{T}(Q)$ of Section **??**.) As a result, the condition of the query $\mu(Q')$ is a subset of the condition of the deregularized version of the query Q. Q.E.D.

Proof of (2): Immediate from (1) and from definition of CVM. $\square$

H.3 Sufficient conditions for combined-semantics containment of CCQ queries

In this subsection we prove Theorem 3.3, see Section H.3.1. We also formulate Theorem H.1, which is a proper generalization of Theorem 3.3; see Section H.3.2.

H.3.1 Proof of Theorem 3.3

In this section we prove Theorem 3.3. Remarkably, the proof is almost verbatim the proof, in [6], of Theorem 2.4.

PROOF. (Theorem 3.3) Consider the queries $Q(\bar{X}) \leftarrow L, M$ and $Q'(\bar{X}') \leftarrow L', M'$. Let φ be a CVM from Q' to Q. Recall that by item (3) of Definition 3.1, when φ is restricted to the domain M' then we have that the range of φ includes all of M.

By Proposition 3.2, φ is a generalized containment mapping from the query Q' to the deregularized version Q_d of Q, $Q_d(\bar{X}) \leftarrow L_d, M_d$. By Proposition 3.1, $Q_d \equiv_C Q$. (In particular, this means that the set M_d is identical to M, and that the set $\bar{S}(Q_d)$ is identical to $\bar{S}(Q)$.)

Let D be a database with active domain $adom(D)$, and let $\bar{d}$ be a tuple of constants in $adom(D)$. Suppose that $Res_C(Q_d, D)$ contains $k > 0$ occurrences of $\bar{d}$. We show that $Res_C(Q', D)$ contains at least k occurrences of $\bar{d}$. This is sufficient in order to prove combined-semantics containment of Q_d in Q'. From this result, by $Q_d \equiv_C Q$ we obtain that $Q \sqsubseteq_C Q'$ also holds.

Let Γ be the set of satisfying assignments of Q_d into D that map $\bar{X}$ to $\bar{d}$. Let $\Gamma_{\bar{S}}$ be the restriction of assignments in Γ to the variables in $\bar{S}(Q_d)$. There are exactly k assignments $\gamma_1, \ldots, \gamma_k \in \Gamma_{\bar{S}}$. We associate each assignment $\gamma_i \in \Gamma_{\bar{S}}$ with an assignment $\gamma_i^* \in \Gamma$ such that γ_i is the restriction of γ_i^* to the variables in $\bar{S}(Q_d)$. If there are several candidates for γ_i^*, we choose one arbitrarily.

Recall that φ is a generalized containment mapping from Q' to Q_d. Thus, we have that for each subgoal l' of Q', $\varphi l' \in L_d$ holds. Since γ_i^* is a satisfying assignment of Q_d into D, we have that $\gamma_i^*(L_d)$ is satisfied by the database. (In other words, all the ground atoms in $\gamma_i^*(L_d)$ appear in D.) By composing the assignments we derive that $\gamma_i^* \circ \varphi(L')$ is satisfied by D. Hence, $\gamma_i^* \circ \varphi$ is a satisfying assignment of Q' into D. In addition, $\gamma_i^* \circ \varphi(\bar{X}') = \bar{d}$, since $\varphi(\bar{X}') = \bar{X}$.

Finally, we show that no two assignments $\gamma_i^* \circ \varphi$ and $\gamma_j^* \circ \varphi$ $(i \neq j)$ agree on all the multiset variables of Q'. By the definition of $\Gamma_{\bar{S}}$, it holds that γ_i and γ_j differ on at least one multiset variable of Q_d. Hence γ_i^* and γ_j^* also differ on at least one multiset variable of Q_d. Since the image of M' under φ includes all of M, we derive that $\gamma_i^* \circ \varphi$ and $\gamma_j^* \circ \varphi$ differ on at least one multiset variable *of* Q'. Therefore, the restrictions of the assignments $\gamma_j^* \circ \varphi$ (for all $j \leq k$) to $\bar{S}(Q')$ are all different satisfiably extendible assignments of the nonset variables of Q' into the database. We conclude that $Res_C(Q', D)$ contains at least k occurrences of $\bar{d}$.

Our arguments apply for all D and for all $\bar{d}$. Therefore, $Q_d \sqsubseteq_C Q'$ and (by $Q_d \equiv_C Q$) we have $Q \sqsubseteq_C Q'$, as required. $\square$

H.3.2 Generalizing Theorem 3.3

We now formulate Theorem H.1, which is a proper generalization of Theorem 3.3. The statement of Theorem H.1 uses Definition H.1, which specifies a proper generalization of CVMs of Definition 3.1. The proof of Theorem H.1 is exactly the same (verbatim) as the proof of Theorem 3.3, see Section H.3.1.

Theorem H.1 provides a rather general sufficient condition for combined-semantics containment of CCQ queries. Consider an illustration.

EXAMPLE H.2. *Consider CCQ queries Q and Q'.*

$Q(X) \leftarrow p(X, Y, Z; i), p(X, W, Z), \{Z, W, i\}$.
$Q'(X) \leftarrow p(X, X, Z), \{Z\}$.

The containment-compatible CCQ pair (Q', Q) satisfies both necessary conditions of Section 3.1 for containment $Q' \sqsubseteq_C Q$. (That is, the pair (Q', Q) satisfies Theorems 3.1 and 3.2).

Consider a mapping μ from the terms of the query Q to the terms of the query Q': $\mu = \{\ X \rightarrow X,\ Y \rightarrow X,\ Z \rightarrow Z,\ W \rightarrow X,\ i \rightarrow 1\ \}$. Observe that (i) μ maps a multiset noncopy variable W of Q into a head variable X of Q', and that (ii) μ maps the copy variable i of Q into a constant, 1. *It follows that:*

- *μ maps the set $\{\ Z,\ W\ \}$ of multiset noncopy variables of the query Q into a* proper superset, *$\{\ Z,\ X\ \}$, of the set $\{\ Z\ \}$ of multiset noncopy variables of the query Q'; and*

- *μ maps the set $\{ \ i \ \}$ of (multiset) copy variables of the query Q into a* proper superset, *$\{ \ 1 \ \}$, of the set $\emptyset$ of (multiset) copy variables of the query Q'.*

We will see later in this subsection that (by Definition H.1), μ is a "relaxed-CVM" from Q to Q'. From the existence of the relaxed-CVM μ from Q to Q', by Theorem H.1 we have that $Q' \sqsubseteq_C Q$ holds. $\square$

To be able to generalize the result of Theorem 3.3, we extend slightly the syntax of [6], which is described in Section 2.1.1. Our extension is to allow for a relational atom $p(\bar{S})$ the equivalent notation $p(\bar{S}; 1)$. This convention does not alter the semantics defined in [6] and parallels the notation of [6] for ground atoms, see Section 2.1.1. Throughout this subsection, we assume that all relational atoms in all query definitions are rendered using this alternative notation.

Definition H.1. *(***Relaxed covering mapping (relaxed-CVM)***) Given CCQ queries Q and Q', a mapping, call it μ, from the terms of Q' to the terms of Q is called a* relaxed covering mapping (relaxed-CVM) from Q' to Q *whenever μ satisfies all of the following conditions:*

(1) μ maps each constant (if any) in Q' to itself;

(2) applying μ to the vector $\bar{X}'$ yields the vector $\bar{X}$;

(3) the set of terms in $\mu M'_{copy}$ is either M_{copy} or $M_{copy} \cup \{1\}$, and the set of terms in $\mu M'_{noncopy}$ includes all of $M_{noncopy}$;

(4) for each relational subgoal of Q', of the form $s(\bar{Y}; 1)$, there exists in Q either a relational subgoal $s(\mu(\bar{Y}); 1)$, or a copy-sensitive subgoal $s(\mu(\bar{Y}); i)$ where $i \in M_{copy}$; and

(5) for each copy-sensitive subgoal of Q' of the form $s(\bar{Y}; i)$, where $i \in M'_{copy}$, there exists in Q a subgoal $s(\mu(\bar{Y}); \mu(i))$.

$\square$

Condition (3) of Definition H.1 properly generalizes condition (3) of Definition 3.1, in the part concerning the copy variables of the queries Q and Q'.

Clearly, for each pair (Q, Q') of CCQ queries, each CVM (satisfying Definition 3.1) from Q' to Q is a relaxed-CVM (satisfying Definition H.1) from Q' to Q. Further, whenever there exists a relaxed-CVM from Q' to Q then (Q, Q') is a containment-compatible CCQ pair.

Note that for relaxed-CVMs, a generalization of Proposition 3.2 still holds:

Proposition H.1. *Given CCQ queries Q and Q'. Then for each relaxed-CVM, μ, from Q' to Q, we have that (1) μ is a GCM from Q' to the deregularized version of Q, and that (2) μ is a relaxed-CVM from Q' to the regularized version of Q.* $\square$

The proof of Proposition H.1 is exactly the same as the proof of Proposition 3.2, provided we use our alternative notation for relational atoms. Example H.3 provides an illustration.

EXAMPLE H.3. *Consider CCQ queries Q and Q'.*

$Q(X) \leftarrow p(X, X, Y; i), p(X, Z, Y; 1), \{Y, i\}$.
$Q'(X) \leftarrow p(X, X, Y; i), \{Y, i\}$.

Query Q (Q', respectively) is exactly the query Q (Q', respectively) of Example 3.2. Note our use, in the definitions of the two queries, of our proposed alternative notation for relational atoms.

Consider the following mapping μ from the terms of the query Q to the terms of the query Q': $\mu = \{ \ X \rightarrow X, Y \rightarrow Y, Z \rightarrow X, i \rightarrow i \ \}$. By Definition H.1, μ is a relaxed-CVM from Q to Q'. (By Definition 3.1, μ is also a CVM from Q to Q'.) Observe that the result of applying μ to the query Q is exactly the deregularized version, *$Q'(X) \leftarrow p(X, X, Y; i), p(X, X, Y; 1), \{Y, i\}$, of the query Q'.* $\square$

We now state and prove a general sufficient condition for combined-semantics containment of CCQ queries, Theorem H.1. The proof of Theorem H.1 is verbatim the proof of Theorem 3.3, see Section H.3.1, provided we use our alternative notation for relational atoms.

Theorem H.1. *Given CCQ queries Q and Q', such that there exists a relaxed-CVM from Q' to Q. Then $Q \sqsubseteq_C Q'$ holds.* $\square$

Theorem H.1 properly generalizes the result of Theorem 3.3: By Theorem H.1 – *but not by* Theorem 3.3 – we have that $Q' \sqsubseteq_C Q$ for the queries Q and Q' of Example H.2.

The condition of Theorem H.1 does not appear to be a necessary condition for containment of CCQ queries. Indeed, the well-known example of [4] (see Appendix I), claims containment $Q \sqsubseteq_C Q'$, but no relaxed-CVM exists from Q' to Q.

H.4 Sufficient condition for bag containment

In this subsection we provide a detailed discussion of the relationship of Theorem 3.3 with the following result of [4].

Theorem H.2. *[4] Given two CCQ bag queries Q and Q' such that there exists a CVM from Q' to Q. Then we have that $Q \sqsubseteq_B Q'$.* $\square$

It is easy to see that Theorem H.2 is an immediate corollary of Theorem 3.3.

Note that Theorem H.2 is formulated here using the syntax of [6] that we adopt in this current paper. Recall that in [4], definitions of bag queries are written using the *implicit* bag syntax. That is, suppose that we know that a query Q is a bag query. This means that we know that (i) for all the nondistinguished variables of Q that are not copy variables, each such variable is a multiset (noncopy) variable of Q, and that (ii) all subgoals of Q are copy-sensitive subgoals. In this case, we can drop altogether from the (explicit, i.e., in the style of [6]) definition of Q (a) the set M, and (b) all copy variables in all subgoals of Q – just because we know how to interpret all subgoals and all explicit variables of a bag query.

The resulting implicit notation makes a CVM from Q' to Q "look like" a containment mapping. That is, suppose there exists an "onto-style containment mapping" μ from bag query Q' to bag query Q when the definitions of both queries use the implicit bag syntax of [4]. By the definition of μ, we have that

(1) Each subgoal l' of Q' is associated by μ with a subgoal l of Q, such that $\mu(l')$ and l have identical relational templates; and that

(2) For each subgoal l of Q, there exists at least one subgoal l' of Q' such that μ associates l' with l.

When we change this definition of μ in such a way that μ still applies to Q' and Q using the (explicit) syntax of [6], it is easy to see that μ is exactly a CVM from the (explicitly defined) Q' to the (explicitly defined) Q. Hence the formulation of Theorem H.2 reflects correctly the result of [4].

H.5 CVMs vs multiset homomorphisms

In this subsection we provide Example H.4 showing that general CVMs and multiset homomorphisms are incomparable when applied to pairs of CCQ queries. We also prove Proposition 3.3.

EXAMPLE H.4. *Let CCQ queries Q and Q' be as follows.*

$Q(A) \leftarrow p(A,B),\ p(A,C),\ \{B,C\}$.
$Q'(D) \leftarrow p(D,E),\ p(D,F),\ \{E\}$.

Consider mapping μ from the terms of query Q to the terms of query Q', and mappings μ' and μ'' from the terms of Q' to the terms of Q: $\mu = \{A \rightarrow D, B \rightarrow E, C \rightarrow E\}$; $\mu' = \{D \rightarrow A, E \rightarrow B, F \rightarrow B\}$; and $\mu'' = \{D \rightarrow A, E \rightarrow B, F \rightarrow C\}$. Mapping μ is a CVM but not a multiset-homomorphism (because μ maps B and C into the same multiset variable E of Q'). Further, each of μ' and μ'' is a multiset-homomorphism but not a CVM. (For each of μ' and μ'', the image of $\{E\}$ under the mapping is not a superset of $\{B,C\}$.) □

PROOF. (Proposition 3.3) The proof is immediate from Proposition 3.2. Indeed, suppose that for an equivalence-compatible CCQ pair (Q, Q'), there exists a SCVM μ from Q' to Q. By Proposition 3.2, μ is a generalized containment mapping from Q' to the deregularized version of Q. Thus, using Definition 3.2, we obtain that condition (4) of Definition 3.1, when applied to μ, to Q', and to the deregularized version of Q, guarantees that condition (3) of Definition 2.4 is satisfied by μ. Observe that by (Q, Q') being an equivalence-compatible CCQ pair, we have that condition (3) of Definition 3.1 for μ guarantees conditions (4) and (5) of Definition 2.4 for μ. Finally, the satisfaction by μ (when applied to Q' and Q) of conditions (1) and (2) of Definition 3.1 guarantees the satisfaction by μ (when applied to Q' and to the deregularized version of Q) of conditions (1) and (2) of Definition 2.4. The opposite direction (that is, a multiset homomorphism φ from Q' to the deregularized version of Q is always a SCVM from Q' to Q) is proved using the above proof "in the opposite direction." □

I. THE NONSURJECTIVE CONTAINMENT EXAMPLE

In this section we recall an example from [4].

EXAMPLE I.1. *Let CCQ queries Q and Q' be as follows.*

$Q(X,Z) \leftarrow p(X;i), s(U,X;j), s(V,Z;k), r(Z;l),$
$\quad \{U,V,i,j,k,l\}$.
$Q'(X,Z) \leftarrow p(X;i), s(U,Y;j), s(V,Y;k), r(Z;l),$
$\quad \{U,V,Y,i,j,k,l\}$.

For bag queries Q and Q', the authors of [4] claim $Q \sqsubseteq_B Q'$, that is $Q \sqsubseteq_C Q'$ in the context of this present paper. □

Observe that for the queries Q and Q' of Example I.1, (Q, Q') is a containment-compatible CCQ pair. (That is, Q and Q' satisfy the necessary containment condition of Theorem 3.1.) At the same time, it is easy to check that no CVM exists from the query Q' to the query Q.

J. QUERY Q OF EXAMPLE 4.1 IS AN IMPLICIT-WAVE QUERY

We show that query Q of Example 4.1 is an implicit-wave query. We observe first that the query Q has two copy-sensitive subgoals. Now the copy-enhanced version Q_{ce} of Q is exactly Q. (Recall that to construct the copy-enhanced version Q_{ce} of query Q, all one needs to do is add distinct copy variables to all relational subgoals of Q. The query Q of Example 4.1 does not have any relational subgoals.) Consider two GCMs from Q_{ce} to itself. (We use here the formulation, specifically the variable naming, for the query Q as given in the beginning of Appendix K.)

$\mu_1 = \{ X_1 \rightarrow X_1, Y_1 \rightarrow Y_1, Y_2 \rightarrow Y_2, X_2 \rightarrow X_2, X_3 \rightarrow X_2, Y_3 \rightarrow Y_3, Y_4 \rightarrow Y_3 \}$; and

$\mu_2 = \{ X_1 \rightarrow X_1, Y_1 \rightarrow Y_1, Y_2 \rightarrow Y_2, X_2 \rightarrow X_3, X_3 \rightarrow X_3, Y_3 \rightarrow Y_4, Y_4 \rightarrow Y_4 \}$.

The set $M_{noncopy}$ for the query Q, as well as for the query Q_{ce}, is $\{Y_1, Y_2\}$. Each of μ_1 and μ_2 is a noncopy-permuting GCM from Q_{ce} to itself, because each of μ_1 and μ_2 maps each element of $M_{noncopy}$ to itself. For the same reason, the mappings μ_1 and μ_2 agree on $M_{noncopy}$.

Mappings μ_1 and μ_2 map the first subgoal of Q (which is an original copy-sensitive subgoal of Q) into atoms with different relational templates. Indeed, $\mu_1(r(X_1,Y_1,Y_2,X_2;Y_3))$ is atom $r(X_1,Y_1,Y_2,X_2;Y_3)$, and $\mu_2(r(X_1,Y_1,Y_2,X_2;Y_3))$ is atom $r(X_1,Y_1,Y_2,X_3;Y_4)$. We conclude that Q is not an explicit-wave query.

K. QUERIES OF EXAMPLE 4.1

In this section we show that, for queries of Example 4.1, $Q \equiv_C Q'$ holds.

PROPOSITION K.1. *For the queries Q and Q' of Example 4.1, we have that $Q \equiv_C Q'$.* □

For the convenience of the exposition in the proof, we use a version of the query Q' where all variables have been renamed into "same-name" *primed* variables: That is, we use

$Q(X_1) \leftarrow r(X_1,Y_1,Y_2,X_2;Y_3), r(X_1,Y_1,Y_2,X_3;Y_4),$
$\quad \{Y_1,Y_2,Y_3,Y_4\}$.
$Q'(X_1') \leftarrow r(X_1',Y_1',Y_2',X_2';Y_3'), r(X_1',Y_1',Y_2',X_2';Y_4'),$
$\quad \{Y_1',Y_2',Y_3',Y_4'\}$.

PROOF. We will prove the claim of Proposition K.1 if we show that for an arbitrary database D and for an arbitrary constant $a \in adom(D)$, the sets $\Gamma^{(a)}_{\bar{S}}(Q,D)$ and $\Gamma^{(a)}_{\bar{S}}(Q',D)$ are of the same cardinality. (Recall the definition of query answer under combined semantics.) To prove this, it is sufficient to show that (for the fixed database D and) for an arbitrary 3-tuple t of constants from $adom(D)$, the sets $\Gamma_{\bar{S}}(Q,D)[t]$ and $\Gamma_{\bar{S}}(Q',D)[t]$ are of the same cardinality. Here, by the set $\Gamma_{\bar{S}}(Q,D)[t]$ we denote the set of all tuples in $\Gamma_{\bar{S}}(Q,D)$ such that the projection of each tuple on the variables X_1, Y_1, Y_2, in this order, is exactly the fixed tuple t. Similarly, by the set $\Gamma_{\bar{S}}(Q', D)[t]$ we denote the set of all tuples in $\Gamma_{\bar{S}}(Q', D)$ such that the projection of each tuple on the variables X'_1, Y'_1, Y'_2, in this order, is exactly the fixed tuple t.

We now prove the latter claim. For the fixed database D, for the remainder of this proof fix a tuple $t = (a, b, c)$, for some $a, b, c \in adom(D)$, as described above.

(1) We first show that whenever the set $\Gamma_{\bar{S}}(Q, D)[t]$ is not empty, the sets $\Gamma_{\bar{S}}(Q, D)[t]$ and $\Gamma_{\bar{S}}(Q', D)[t]$ are of the same cardinality k^2, for some constant $k \in \mathbb{N}_+$ where k is a copy number of some ground atom of the database D.

Suppose that the set $\Gamma_{\bar{S}}(Q, D)[t]$ is not empty. Then there must exist in D ground atoms (perhaps identical to each other) $g_1 = r(a, b, c, d; e)$ and $g_2 = r(a, b, c, f; h)$, for some $d, f \in adom(D)$ and for some $e, h \in \mathbb{N}_+$. These atoms g_1 and g_2 must intuitively be the images of the first and of the second subgoal of the query Q, respectively, under a valid assignment mapping from Q to D. That is, formally, for the set $\Gamma_{\bar{S}}(Q, D)[t]$ to be a nonempty set, it must be that the mapping $\{ X_1 \rightarrow a,\ Y_1 \rightarrow b,\ Y_2 \rightarrow c,\ X_2 \rightarrow d,\ X_3 \rightarrow e,\ Y_3 \rightarrow 1,\ Y_4 \rightarrow 1 \}$ is a valid assignment mapping from all the terms of the query Q to the elements of $adom(D) \bigcup \mathbb{N}_+$. The validity of this assignment mapping is justified by the presence of the ground atoms g_1 and g_2 in the database D.

We now consider all those ground atoms in relation R in the database D, such that each of the atoms has a, b, c, in this order, as the values of the first three attributes of the relation R, from left to right. We know that the set, call it $S[Q]$, of all such atoms is not empty, as g_1 and g_2 of the previous paragraph will be elements of this set. Now let the constant $k \in \mathbb{N}_+$ be the maximal value of the copy number among all the ground atoms in the set $S[Q]$. From the set $S[Q]$, choose an arbitrary atom, call it g, whose copy number is k. Let g be $r(a, b, c, l; k)$, for some $l \in adom(D)$.

We now argue that for each $n_1, n_2 \in \{1, \ldots, k\}$ and for the constant l in the ground atom g, the mapping $\mu_{(n_1,n_2,l)} = \{ X_1 \rightarrow a,\ Y_1 \rightarrow b,\ Y_2 \rightarrow c,\ X_2 \rightarrow l,\ X_3 \rightarrow l,\ Y_3 \rightarrow n_1,\ Y_4 \rightarrow n_2 \}$ is a valid assignment mapping from all the terms of the query Q to the elements of $adom(D) \bigcup \mathbb{N}_+$. Indeed, the required fact comes straight from the definition of the set $\Gamma(Q, D)$ and from the presence of the atom g in the database D.

Further, we argue that for each natural number n_1 that is strictly greater than the constant k, for each $n_2 \in \mathbb{N}_+$, and for *each* constant $l \in adom(D)$, the mapping $\mu_{(n_1,n_2,l)}$ as defined above is not a valid assignment mapping from all the terms of the query Q to the elements of $adom(D) \bigcup \mathbb{N}_+$. Indeed, it is sufficient to observe that the set $S[Q]$ does not have atoms whose copy number is greater than k. (Recall that $\mu_{(n_1,n_2,l)}$ fixes the images of the variables X_1, Y_1, and Y_2 to the respective elements of the tuple $t = (a, b, c)$.) We show in a similar way that for each natural number n_2 that is strictly greater than the constant k, for each $n_1 \in \mathbb{N}_+$, and for *each* constant $l \in adom(D)$, the mapping $\mu_{(n_1,n_2,l)}$ is not a valid assignment mapping from all the terms of the query Q to the elements of $adom(D) \bigcup \mathbb{N}_+$.

From the facts established about the mappings $\mu_{(n_1,n_2)}$ we conclude that the set $\Gamma_{\bar{S}}(Q, D)[t]$ has exactly k^2 elements. Now consider the set $\Gamma_{\bar{S}}(Q', D)[t]$. It is easy to show (in fact, easier than for $\Gamma_{\bar{S}}(Q, D)[t]$ as we did above) that the set $\Gamma_{\bar{S}}(Q', D)[t]$ also has exactly k^2 elements. (For each valid assignment mapping μ from all the terms of the query Q' to the elements of $adom(D) \bigcup \mathbb{N}_+$, such that μ maps X'_1 to a, Y'_1 to b and Y'_2 to c, μ induces a mapping from both subgoals of the query Q' into *the same* ground atom of the database D. Specifically, for the ground atom $g \in S[Q]$ as defined above, there exists a valid assignment mapping of this form μ, such that the mapping induces a mapping from both subgoals of the query Q' into the atom g.)

(2) Now suppose that for the above fixed D and t, the set $\Gamma_{\bar{S}}(Q', D)[t]$ is not empty. We show that in this case, the sets $\Gamma_{\bar{S}}(Q, D)[t]$ and $\Gamma_{\bar{S}}(Q', D)[t]$ are of the same cardinality p^2, for some constant $p \in \mathbb{N}_+$ where p is a copy number of some ground atom of the database D. The proof is symmetric to the proof of the claim (1) above. Q.E.D. $\square$

L. PROOF OF THEOREM 4.1

L.1 Intuition for the Proof and Extended Example

L.1.1 Intuition for the Proof

In this subsection we outline the idea of the proof of Theorem 4.1. Intuitively, we generalize the proof, via canonical databases, of the existence of a containment mapping from CCQ set query Q' to CCQ set query Q whenever $Q \equiv_S Q'$. The challenge in the generalization is that we are looking for a SCVM from Q' to Q, that is, the desired mapping must map each multiset variable of Q' into a distinct multiset variable of Q. Showing that we have constructed a mapping with this property is thus an essential part of the proof. (Recall that, unlike in our minimization problem of Section 5, here we have no information about the structural relationship between the two queries.)

For a given CCQ query Q, the proof of Theorem 4.1 constructs an infinite number of databases, where each database $D_{\bar{N}^{(i)}}(Q)$, $i \geq 1$, can be thought of as a union of "extended canonical databases" for Q. (See Appendix G.1 for the definition.) Similarly to canonical databases for CCQ set queries, each ground atom in each database $D_{\bar{N}^{(i)}}(Q)$ can be associated, via a mapping that we denote $\nu_Q^{(i)}$, with a unique subgoal of the query Q. See Section L.3 for the details.

The role of each database $D_{\bar{N}^{(i)}}(Q)$ in the proof is that the database represents a particular combination of multiplicities of the values of (some of) the multiset variables Y_1, Y_2, $\ldots$, Y_n, for some $n \geq 1$, of the query Q. (We have that $n \geq 1$ for all CCQ queries Q and Q' such that $Q \equiv_C Q'$ and at least one of Q and Q' is not a set query.) For each database $D_{\bar{N}^{(i)}}(Q)$, we represent the n respective multiplicities as natural numbers $N_1^{(i)}$ through $N_n^{(i)}$, or equivalently via the n-ary vector $\bar{N}^{(i)}$.

By construction of the databases, we have that some fixed tuple, t_Q^*, is an element of the bag $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$ for each $i \geq 1$. Moreover, *for all queries Q'' such that (Q, Q'') is an equivalence-compatible CCQ pair,* we have that the multiplicity of the tuple t_Q^* in each bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$ (that is, for each $i \geq 1$) can be expressed using the symbolic representations, N_1 through N_n, of the respective elements $N_1^{(i)}$, $\ldots$, $N_n^{(i)}$ of the vector $\bar{N}^{(i)}$. That is, for each such query Q'', we can obtain explicitly a function, $\mathcal{F}_{(Q)}^{(Q'')}$, in terms of the n variables $N_1, \ldots, N_n$, such that whenever we substitute $N_j^{(i)}$ for N_j, for each $j \in \{1, \ldots, n\}$, the resulting expression in terms of $N_1^{(i)}$, ..., $N_n^{(i)}$ evaluates to the multiplicity of the tuple t^*_Q in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$. See Sections L.4 through L.6 and Section L.9 for the construction of the function. Sections L.8 and L.9 contain extended illustrations of the constructions.

Observe that for each CCQ query Q' such that $Q' \equiv_C Q$, it must be that the functions $\mathcal{F}^{(Q')}_{(Q)}$ and $\mathcal{F}^{(Q)}_{(Q)}$ output the same value on each database $D_{\bar{N}^{(i)}}(Q)$, $i \geq 1$.

Consider the simplest case, where our query Q has no self-joins and has $|M| = n \geq 1$. In this case, by construction of the databases, we have that the function $\mathcal{F}^{(Q)}_{(Q)}$ *for the query* Q is the monomial $\Pi^n_{j=1} N_j$. Consider an arbitrary assignment, γ, from Q to a $D_{\bar{N}^{(i)}}(Q)$. Note that each such γ has contributed to the construction of the database; we call γ a *generative assignment from* Q *to* $D_{\bar{N}^{(i)}}(Q)$. We can show that the composition $\nu^{(i)}_Q \circ \gamma$ is a SCVM from Q to itself. (Note the presence in the product $\Pi^n_{j=1} N_j$ of the variables for all the n multiset variables of Q.) Moreover, for each query Q' such that $Q' \equiv_C Q$, the function $\mathcal{F}^{(Q')}_{(Q)}$ is forced (by $Q' \equiv_C Q$ and by $\mathcal{F}^{(Q)}_{(Q)}$ being a multivariate polynomial) to be exactly $\Pi^n_{j=1} N_j$, regardless of the relationship between the structures of Q and Q'. We show that whenever $\mathcal{F}^{(Q')}_{(Q)}$ $= \Pi^n_{j=1} N_j$, an assignment from Q' to a database $D_{\bar{N}^{(i)}}(Q)$ can be composed with the mapping $\nu^{(i)}_Q$ to yield a SCVM from Q' to Q, precisely due to the presence in the function $\mathcal{F}^{(Q')}_{(Q)}$ of the "representative" N_j of each multiset variable Y_j of the query Q, for each $j \leq n$.

The above exposition conveys the general intuition of the proof of Theorem 4.1: For all CCQ queries Q, there is a monomial, in terms of *all* of N_1, ..., N_n, that contributes to the construction of the function $\mathcal{F}^{(Q)}_{(Q)}$ and that reflects the multiplicity, in the set $\Gamma^{(t^*_Q)}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$, of all generative assignments from Q to databases $D_{\bar{N}^{(i)}}(Q)$. We call this monomial, $\mathcal{P}^{(Q)}_*$, *the wave of the query* Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$. (See Section L.7 for the definition.) Suppose that, for a query Q' such that $Q' \equiv_C Q$, we can show that the function $\mathcal{F}^{(Q')}_{(Q)}$ has, as a term, the wave of Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$, *backed up by assignments from* Q' *to the databases* $D_{\bar{N}^{(i)}}(Q)$. Then we can use these assignments and the mapping $\nu^{(i)}_Q$ to construct a SCVM from Q' to Q.

There are two challenges in implementing this idea for general CCQ queries. First, the term $\mathcal{P}^{(Q)}_*$ may not be "visible" in the expression for $\mathcal{F}^{(Q)}_{(Q)}$. As a result, the term $\mathcal{P}^{(Q)}_*$ does not necessarily contribute to the construction of the function $\mathcal{F}^{(Q')}_{(Q)}$, even in case $Q \equiv_C Q'$. (This is exactly the case of queries Q and Q' of Example 4.1, see Example L.1 in Appendix L.1 for the details.) Second, in general, function $\mathcal{F}^{(Q')}_{(Q)}$ may have terms that are *not* backed up by assignments from Q' to databases $D_{\bar{N}^{(i)}}(Q)$. Both challenges arise from the fact that the function $\mathcal{F}^{(Q'')}_{(Q)}$, in terms of N_1, ..., N_n, is, in general, *not* a multivariate polynomial on its entire domain.

To overcome the first challenge, we introduce the restriction that Q be an *explicit-wave query.* (Hence Definition 4.1 is necessarily technical.) Even under this restriction, overcoming the second challenge requires a nontrivial proof. (See Section L.10, with its main result Proposition L.47.)

In a little more detail, the proof of Theorem 4.1 is immediate from the following three results. (*Monomial classes* and their *multiplicity monomials* are constructs that we introduce to build functions $\mathcal{F}^{(Q)}_{(Q)}$ and $\mathcal{F}^{(Q')}_{(Q)}$. All the details on these constructs can be found in Section L.6. Section L.8 contains an extended illustration of these constructs.)

- Proposition L.33 of Section L.7 states the following: Given a CCQ query Q, there exists a nonempty monomial class, call it $\mathcal{C}^{(Q)}_*$, for the query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of $\mathcal{C}^{(Q)}_*$ is the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$.
- Proposition L.34 of Section L.7 states the following: Given CCQ queries $Q(\bar{X}) \leftarrow L, M$ and $Q'(\bar{X}') \leftarrow L', M'$, such that (i) Q and Q' have the same (positive-integer) head arities, (ii) $|M_{copy}| = |M'_{copy}|$, and (iii) $|M_{noncopy}|$ $= |M'_{noncopy}|$. Suppose that there exists a nonempty monomial class $\mathcal{C}^{(Q')}_*$ for the query Q' w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of $\mathcal{C}^{(Q')}_*$ is the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$. Then there exists a SCVM from the query Q' to the query Q.
- Proposition L.47 of Section L.10 states the following: Whenever
  (a) $Q \equiv_C Q'$ for CCQ queries Q and Q', and
  (b) Q is an explicit-wave CCQ query (as specified by Definition 4.1),

  then there exists a (nonempty) monomial class $\mathcal{C}^{(Q')}_*$ for the query Q' and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of $\mathcal{C}^{(Q')}_*$ is the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$.

L.1.2 An Illustration

We now provide an extended illustration of how the term $\mathcal{P}^{(Q)}_*$ may not be "visible" in the expression for $\mathcal{F}^{(Q)}_{(Q)}$, and of how, in general, the function $\mathcal{F}^{(Q)}_{(Q)}$ is not a multivariate polynomial on its entire domain. Example L.6 in Appendix L.9.6 is an extended variant of this Example L.1. In addition, Example L.6 illustrates how function $\mathcal{F}^{(Q)}_{(Q)}$ may have terms that are *not* backed up by assignments from Q to databases $D_{\bar{N}^{(i)}}(Q)$.

EXAMPLE L.1. *For the query Q of Example 4.1, we associate a variable N_j with each of the $n = 4$ multiset variables Y_j of the query, $1 \leq j \leq 4$. Consider assignments $N_1 := N_2 := 1$, $N_3 := 2$, and $N_4 := 3$. For some fixed i and for $1 \leq j \leq 4$, each of these assignments associates the "value" $N_j^{(i)}$ with the variable N_j. For the resulting vector $\bar{N}^{(i)} = [\ 1\ 1\ 2\ 3\ ]$, we construct the database $D_{\bar{N}^{(i)}}(Q)$ $= \{\ r(a, b, c, d; 2), r(a, b, c, e; 3)\ \}$, as specified in the proof of Theorem 4.1. We will refer to the ground atom $r(a, b, c, d; 2)$ as d_1, and to $r(a, b, c, e; 3)$ as d_2.*

Each generative assignment from Q to $D_{\bar{N}^{(i)}}(Q)$ maps $X_1 \rightarrow a$, $Y_1 \rightarrow b$, $Y_2 \rightarrow c$, $X_2 \rightarrow d$, and $X_3 \rightarrow e$. By definition of combined-semantics query evaluation, these generative assignments together contribute to the set $\Gamma^{(t^*_Q)}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$, for $t^*_Q = (a)$, exactly $\Pi^4_{j=1}\ N_j^{(i)} = 6$ distinct tuples. As *shown in the proof of Theorem 4.1, the number of tuples contributed to the set $\Gamma_{\bar{S}}^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$ by the set of generative assignments from Q to $D_{\bar{N}^{(i)}}(Q)$, for an* arbitrary $i \geq 1$, *can be obtained by substituting, into the formula $\Pi_{j=1}^{4}\ N_j$, the values $N_j^{(i)}$ of the variables N_j. The values $N_j^{(i)}$ come from the specific vector $\bar{N}^{(i)}$ representing the database $D_{\bar{N}^{(i)}}(Q)$. Recall that $\Pi_{j=1}^{4}\ N_j$ is the wave $\mathcal{P}_*^{(Q)}$ of the query Q.*

The variety of possible assignments from the query Q to the above database $D_{\bar{N}^{(i)}}(Q)$ (for the fixed vector $\bar{N}^{(i)}$) stems from the ability of the first subgoal, g_1, of Q to map to each of the ground atoms d_1 and d_2, and from the ability of the second subgoal, g_2, of Q to independently also map to each of d_1 and d_2. (All the above generative assignments map g_1 to d_1, and map g_2 to d_2.) It is easy to see that for those assignments from Q to $D_{\bar{N}^{(i)}}(Q)$ that map each of g_1 and g_2 to d_1, the set of all such assignments contributes to the set $\Gamma_{\bar{S}}^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$ exactly $(N_3^{(i)})^2 = 4$ distinct tuples. Similarly, mapping g_1 to d_2 and g_2 to d_1 contributes to the set $\Gamma_{\bar{S}}^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$ exactly $N_3^{(i)} \times N_4^{(i)} = 6$ distinct tuples, and mapping each of g_1 and g_2 to d_2 contributes to the set $\Gamma_{\bar{S}}^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$ exactly $(N_4^{(i)})^2 = 9$ distinct tuples. (Recall that for our fixed database $D_{\bar{N}^{(i)}}(Q)$, $N_1^{(i)} = N_2^{(i)} = 1$.) Similarly to the previous paragraph, the contribution of each such class of assignments from Q to $D_{\bar{N}^{(i)}}(Q)$ for an arbitrary $i \geq 1$ *can be expressed symbolically using monomials in terms of some of the variables $N_1, \ldots, N_4$. For instance, for the class of all assignments that map each of g_1 and g_2 to d_2, the number of distinct tuples contributed by these assignments to $\Gamma_{\bar{S}}^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$, for each $i \geq 1$, can be obtained using the monomial $\mathcal{T}_{(d_2)} = N_1 \times N_2 \times (N_4)^2$ and each specific vector $\bar{N}^{(i)}$.*

In constructing the function $\mathcal{F}_{(Q)}^{(Q)}$ using all the above monomials in terms of N_1 through N_4, the problem is that we cannot simply set $\mathcal{F}_{(Q)}^{(Q)}$ to the sum of the monomial $\mathcal{P}_*^{(Q)}$ with the monomial $\mathcal{T}_{(d_2)}$ and with the monomials that can be built from the other classes of assignments from Q to $D_{\bar{N}^{(i)}}(Q)$ using the above reasoning. The reason is, for our fixed vector $\bar{N}^{(i)} = [\ 1\ 1\ 2\ 3\ ]$, the total number of tuples in the set $\Gamma_{\bar{S}}^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$ – and thus the total number of copies of the tuple $t^*_Q = (a)$ in the bag $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$ – is the result of substituting the values from the vector $\bar{N}^{(i)}$ into the single monomial $\mathcal{T}_{(d_2)}$. The problem stems from these different classes of assignments possibly contributing the same *tuples into the set $\Gamma_{\bar{S}}^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$. Thus, in general, constructing the function $\mathcal{F}_{(Q)}^{(Q)}$ from the monomials representing different classes of assignments has to be done in a way that takes into account these overlapping contributions.*

For our specific query Q, we show in Example L.6 in Appendix L.9.6 that (1) $\mathcal{F}_{(Q)}^{(Q)} = N_1 \times N_2 \times (N_4)^2$ for all vectors $\bar{N}^{(i)}$ where $N_3^{(i)} \leq N_4^{(i)}$, and that (2) $\mathcal{F}_{(Q)}^{(Q)} = N_1 \times N_2 \times (N_3)^2$ for all vectors $\bar{N}^{(i)}$ where $N_3^{(i)} \geq N_4^{(i)}$. A compact representation of $\mathcal{F}_{(Q)}^{(Q)}$ that works for all $i \geq 1$ is $\mathcal{F}_{(Q)}^{(Q)} = N_1 \times N_2 \times (max(N_3, N_4))^2$. Clearly, this expression cannot be rewritten equivalently as a multivariate polynomial on the entire domain $\{\ \bar{N}^{(i)},\ i \geq 1\ \}$ of the vector $\bar{N}$.

Consider an illustration of the problem with the term $\mathcal{P}_*^{(Q)}$ not being "visible" in any of the above expressions for the function $\mathcal{F}_{(Q)}^{(Q)}$. Indeed, recall the query Q' of Example 4.1; we have that $Q \equiv_C Q'$. It turns out that, even though $\mathcal{P}_*^{(Q)}$ has (technically) contributed to the construction of the function $\mathcal{F}_{(Q)}^{(Q)}$ for the query Q, *there still does not exist a class of assignments from* the query Q' *to database $D_{\bar{N}^{(i)}}(Q)$, such that the total number of tuples contributed to the set $\Gamma_{\bar{S}}^{(t^*_Q)}(Q', D_{\bar{N}^{(i)}}(Q))$ by the assignments in this class can be expressed by the monomial $\mathcal{P}_*^{(Q)}$. It is easy to verify that for the queries of Example 4.1, there does not exist a SCVM from Q' to Q.* □

L.2 Assumptions, Conventions, Basic Results

To streamline the exposition in the proof of Theorem 4.1, we reserve a number of uppercase and lowercase Latin letters, in a variety of fonts and some with sub- and superscripts, to each have "the standard meaning" throughout the proof. Every effort has been made to introduce all of the notation in subsections that are separate from sections for (portions of) the proof of Theorem 4.1.

L.2.1 The Queries

The basics.

For the fixed (input) CCQ queries Q and Q', as well as for the CCQ query Q'' in this proof, we use the notation $Q(\bar{X}) \leftarrow L, M$; $Q'(\bar{X}') \leftarrow L', M'$; and $Q''(\bar{X}'') \leftarrow L'', M''$. Denote by $l \geq 1$ the head arity of Q. Denote by P (by P', respectively) the (possibly empty) set of all constants in the query Q (in the query Q', respectively).

Proposition L.1. *Let $Q \equiv_C Q'$. Then $P = P'$.* □

Proof. The proof is by contradiction: Assume that P and P' are not the same sets. Pick a constant c in $P - P'$. (If $P - P' = \emptyset$ then pick a constant $c \in P' - P$ and modify the rest of this proof by swapping Q with Q' in all the statements of the proof.) Construct the canonical database $D^{(Q')}$ of Q' in such a way that $adom(D^{(Q')})$ does not have the value c. (It is always possible to construct a $D^{(Q')}$ that would satisfy this restriction.) Then it is easy to see that (i) the bag $Res_C(Q', D^{(Q')})$ must be nonempty by construction of the database, and (ii) the bag $Res_C(Q, D^{(Q')})$ must be empty due to the absence of the constant c in $adom(D^{(Q')})$. Hence we arrive at a contradiction with the assumption that $Q \equiv_C Q'$. □

We use the notation M_{copy} (M'_{copy}, M''_{copy}, respectively) for the set of copy variables of query Q (of Q', of Q'', respectively). We use the notation $M_{noncopy}$ ($M'_{noncopy}$, $M''_{noncopy}$, respectively) for the set of multiset noncopy variables of query Q (of Q', of Q'', respectively).

We denote by m the number $|M_{noncopy}|$ of multiset noncopy variables in the query Q, and by r the number $|M_{copy}|$ of copy variables in the query Q. For the CCQ query Q'' in this proof, we assume that (i) Queries Q and Q'' have the same head arity (l); (ii) $|M_{noncopy}| = |M''_{noncopy}|$; and (iii) $|M_{copy}| = |M''_{copy}|$.

The following observation is immediate from the assumption $Q \equiv_C Q'$ (of Theorem 4.1) and from Theorem 3.1.

Proposition L.2. *(i) Queries Q and Q' have the same head arity (l); (ii) $|M_{noncopy}| = |M'_{noncopy}|$; and (iii) $|M_{copy}| = |M'_{copy}|$.* □

The following result is immediate from the containment-mapping theorem of [2]. Thus, for the remainder of the proof of Theorem 4.1, *we assume that the set M of multiset variables of Q is not empty (that is, $m + r \geq 1$).*

PROPOSITION L.3. *Let $M = \emptyset$. Then Theorem 4.1 holds.* $\square$

Throughout the proof of Theorem 4.1 we assume that we are given the regularized version of the query Q.

Equivalence classes of subgoals of query Q.

We now introduce the notation that will help us to deal cleanly with the case where query Q has more than one copy-sensitive subgoal for a particular relational template. (See, e.g., query Q in Example **??**.

We first partition all the copy-sensitive subgoals (in case $r \geq 1$) of the query Q into equivalence classes: Place two copy-sensitive subgoals of Q into the same equivalence class if and only if the two subgoals agree on the predicate name and on all the arguments except the copy variable. That is, two distinct copy-sensitive subgoals g_1 and g_2 of Q are in the same equivalence class if and only if the relational templates of g_1 and g_2 are the same. Denote by $C'_1, \ldots, C'_w$, $w \geq 1$, the resulting equivalence classes for all the copy-sensitive subgoals of Q. (We have $w \geq 1$ only in case where $r = |M_{copy}| > 0$. Otherwise we set $w := 0$.)

Further, we assume that each (if any) relational subgoal, g, of the query Q is in its own equivalence class $\{g\}$. Let $C_1, \ldots, C_v$, $v \geq 0$, be the resulting equivalence classes of the relational subgoals of the query Q. (The case $v = 0$ holds if and only if all subgoals of the query Q are copy-sensitive, rather than relational, atoms.)

Denote by $\mathcal{C}_Q = \{C_1, \ldots, C_v, C'_1, \ldots, C'_w\}$, with $v+w \geq 1$, the set of all equivalence classes of the subgoals of the query Q, as defined in the preceding paragraphs. (By Definition 2.1, the condition of the query Q contains at least one atom, thus $v + w \geq 1$ must hold.) Further, for each class $C \in \mathcal{C}_Q$, choose one arbitrary element of C, call this element $s(C)$, and fix $s(C)$ as *the representative element of the class C*. Denote by $\mathcal{S}_{C(Q)} = \{s(C_1), \ldots, s(C_v), s(C'_1), \ldots, s(C'_w)\}$, with $v+w \geq 1$, the set of the representative-element subgoals of the query Q. The following observation is immediate from the definitions and from the fact that we use the regularized version of the query Q. (Recall that L is the condition of the query Q, and that $r = |M_{copy}|$. Item (v) is immediate from item (iv) and from our assumption $m + r = |M| \geq 1$.)

PROPOSITION L.4. *(i) $\mathcal{S}_{C(Q)} \subseteq L$. (ii) For an arbitrary (relational or copy-sensitive) atom g, the set $\mathcal{S}_{C(Q)}$ has at most one element whose relational template is the same as the relational template of g. (iii) $r \geq w$ always holds. (iv) If $r > 0$ then $w > 0$. (v) $m + w \geq 1$.* $\square$

Notation for query variables.

For ease of exposition, we assume that each of Q, Q', and Q'' has l distinct head variables. (Recall that $l \geq 1$ denotes the head arity of the query Q. Handling the cases where Q, Q', or Q'' has either repeated occurrences of the same variable in the head, or has constants in the head, would be straightforward extensions of this proof but would make the exposition considerably harder to follow.) Suppose that Q has $u \geq 0$ nonhead *set* variables. W.l.o.g. denote by $X_1, \ldots, X_l$ all the head variables of Q (from left to right in the head vector $\bar{X}$ of Q), and (in case $u > 0$) denote by $X_{l+1}, \ldots, X_{l+u}$ all the u nonhead set variables of Q.

In case $m = |M_{noncopy}| \geq 1$, let $Y_1, \ldots, Y_m$ be w.l.o.g. all the multiset *noncopy* variables of the query Q. Further, in case $w \geq 1$ let $Y_{m+1}, \ldots, Y_{m+w}$ be the copy variables of the elements $s(C'_1), \ldots, s(C'_w)$ of the set $\mathcal{S}_{C(Q)}$. By Proposition L.4 (v), the set of variables $\{Y_1, \ldots, Y_m, Y_{m+1}, \ldots, Y_{m+w}\}$ is not empty.

Further, consider the set of all r copy-sensitive subgoals of Q. Whenever $r > w$ — that is, in case there exists a copy-sensitive subgoal of Q that is not the representative element of any of the classes $C'_1, \ldots, C'_w$ — let $Y_{m+w+1}, \ldots, Y_{m+r}$ be (w.l.o.g.) the copy variables of all the copy-sensitive subgoals of Q that are not the representative elements of any of the classes $C'_1, \ldots, C'_w$.

In case $m \geq 1$ we denote by Y'_1, …, Y'_m (by Y''_1, …, Y''_m, respectively) the multiset noncopy variables of query Q' (of query Q'', respectively). In case $r \geq 1$ we denote by $Y'_{m+1}, \ldots, Y'_{m+r}$ (by $Y''_{m+1}, \ldots, Y''_{m+r}$, respectively) the copy variables of query Q' (of query Q'', respectively). Finally, we denote by X'_1, …, X'_l (by X''_1, …, X''_l, respectively) the (distinct) head variables of query Q' from left to right in the vector $\bar{X}'$ (of query Q'' from left to right in the vector $\bar{X}''$, respectively). All of this notation is w.l.o.g. by the basic results and assumptions about Q' and Q'' in the beginning of this section.

L.2.2 Convention for Ground Atoms in Databases

We use the following convention for ground atoms in databases in this proof. Let $g = p(\bar{Y})$, for some choice of p and $\bar{Y}$, be a ground atom in database D, and let $n \geq 1$ be the total number of copies of g in D. Then we treat the n copies of g in D as a *single* ground atom $p(\bar{Y})$ with associated copy number n, and represent it as $p(\bar{Y}; n)$, as defined in Section 2.1.1. As a result, every database is a set when using this representation.

L.3 Constructing Family of Databases $\mathcal{D}_{\bar{N}}(Q)$

This section describes the construction of an infinite family of databases based on the input query Q. Each database in the family is constructed as a union of extended canonical databases for the query Q. In the exposition in this section we use the notation introduced in Section L.2.

Throughout the proof of Theorem 4.1, whenever we discuss or use "the family of databases as constructed in Section L.3", we always refer to the family of databases built based on the fixed input query Q (see Section L.2.1), *regardless of the context.* (This convention is lifted in part of Example L.5.)

L.3.1 Mappings ν_0 and $\nu_Q^{(i)}$, vector $\bar{N}$ and its domain $\mathcal{N}$, sets S_0 and $S_1^{(i)}, \ldots, S_m^{(i)}$, tuple t_Q^*

We begin by defining a bijective mapping called ν_0. The domain of ν_0 is the set of all terms of the query Q that are not multiset variables of the query. The range of ν_0 is a subset of the active domain of each database in the family $\mathcal{D}_{\bar{N}}(Q)$. *The assignment ν_0 is fixed to be the same across all the databases in the family $\mathcal{D}_{\bar{N}}(Q)$.*

We define ν_0 as follows: To each head variable and each set variable of Q, that is to each of the variables $X_1, \ldots, X_{l+u}$ ($l + u \geq 1$), ν_0 assigns a distinct constant value that is also distinct from all the values in the set P of constants used in the query Q. Denote by S_0 the set of $l+u$ constant values in the range of ν_0, $S_0 \bigcap P = \emptyset$. Further, to each (if any) constant c used in Q, $c \in P$, $\nu_0(c) := c$. The mapping ν_0 is bijective by construction.

We define tuple t^*_Q as $t^*_Q = \nu_0[\bar{X}]$. (Recall that $\bar{X}$ is the vector of head terms of the query Q.) By definition, tuple t^*_Q is fixed to be the same across all the databases in the family $\mathcal{D}_{\bar{N}}(Q)$.

Define $\bar{N}$ as a vector of variables $N_1, N_2, \ldots, N_{m+w}$, $m + w \geq 1$. (For m and w, see Section L.2.) The vector $\bar{N}$ is defined on the Cartesian product of $m+w$ copies of the set $\mathbb{N}_+$ of natural numbers; we denote by $\mathcal{N}$ this domain of $\bar{N}$. Fix an arbitrary enumeration (starting with 1) of the set $\mathcal{N}$. By the vector $\bar{N}^{(i)} \in \mathcal{N}$, $i \in \mathbb{N}_+$, we denote the ith element of $\mathcal{N}$ in this enumeration. We use the notation $N^{(i)}_1, N^{(i)}_2, \ldots, N^{(i)}_{m+w}$ for the natural-number values, from left to right, in the vector $\bar{N}^{(i)}$. That is, the natural number $N^{(i)}_j$, $1 \leq j \leq m+w$, in the jth position of vector $\bar{N}^{(i)}$, is the value, w.r.t. the fixed i, of the variable N_j in the vector $\bar{N}$.

Suppose that the query Q is such that we have $m = |M_{noncopy}| \geq 1$. Fix an $i \in \mathbb{N}_+$ and consider the vector $\bar{N}^{(i)}$. For each value $N^{(i)}_j$ such that $1 \leq j \leq m$, define $S^{(i)}_j$ as a set of $N^{(i)}_j$ distinct constants such that all the sets $S^{(i)}_j$ ($1 \leq j \leq m$) are pairwise disjoint and such that $S^{(i)}_j \bigcap P = \emptyset$ and $S^{(i)}_j \bigcap S_0 = \emptyset$ for all $j \in \{1, \ldots, m\}$.

We define the set $S^{(i)}_*$ as the empty set in case $m = 0$, and as the union $\bigcup_{j=1}^{m} S^{(i)}_j$ in case $m \geq 1$.

Finally, for the vector $\bar{N}^{(i)}$, for each $i \in \mathbb{N}_+$, we define mapping $\nu^{(i)}_Q$, to be used in Section L.3.3 and in other parts of the proof of Theorem 4.1. Define the domain of the mapping $\nu^{(i)}_Q$ as the union $P \bigcup S_0 \bigcup S^{(i)}_*$. Define mapping $\nu^{(i)}_Q$ as follows:

- For each $c \in S_0 \bigcup P$, let $\nu^{(i)}_Q(c) := \nu_0^{-1}(c)$.
- In case $m \geq 1$: For each $j \in \{1, \ldots, m\}$ and for each $c \in S^{(i)}_j$, let $\nu^{(i)}_Q(c) := Y_j$.

L.3.2 Main Construction Cycle for $D_{\bar{N}^{(i)}}(Q) \in \mathcal{D}_{\bar{N}}(Q)$

The family of databases $\mathcal{D}_{\bar{N}}(Q)$ that we are about to construct is the infinite set $\{D_{\bar{N}^{(1)}}(Q), D_{\bar{N}^{(2)}}(Q), \ldots, D_{\bar{N}^{(i)}}(Q), \ldots\}$, where each database $D_{\bar{N}^{(i)}}(Q)$, $i \in \mathbb{N}_+$, is associated with the vector $\bar{N}^{(i)} \in \mathcal{N}$. We will refer to the family $\mathcal{D}_{\bar{N}}(Q)$ either as $\{D_{\bar{N}^{(i)}}(Q) \mid i \in \mathbb{N}_+ \}$ or simply as $\{D_{\bar{N}^{(i)}}(Q)\}$.

Fix an $i \in \mathbb{N}_+$ and consider the vector $\bar{N}^{(i)}$. We first define the set $\mathcal{S}^{(i)}$, to be used in the main construction cycle for creating the database $D_{\bar{N}^{(i)}}(Q)$ for the vector $\bar{N}^{(i)}$. In case that we have $m = 0$, define the set $\mathcal{S}^{(i)}$ as a singleton set consisting of a single empty tuple. With that (only) element of the set $\mathcal{S}^{(i)}$, we associate an *empty* assignment $\nu^{noncopy}_t$. Now consider the case where the query Q is such that we have $m \geq 1$. Let $\mathcal{S}^{(i)}$ be the cross product $S^{(i)}_1 \times S^{(i)}_2 \times \ldots \times S^{(i)}_m$. For each tuple t in $\mathcal{S}^{(i)}$ in this case $m \geq 1$, we treat t as an assignment $\nu^{noncopy}_t$ of values to the multiset noncopy variables (from left to right) $Y_1, \ldots, Y_m$ of Q.

In case $r \geq 1$ we define a mapping ν^{copy} on the set $Y_{m+1}, \ldots, Y_{m+r}$ of copy variables of the query Q. (I) For each copy variable from among $Y_{m+1}, \ldots, Y_{m+w}$ of Q, define $\nu^{copy}(Y_j) := N^{(i)}_j$ for each $j \in \{m+1, \ldots, m+w\}$. (II) Whenever $r > w$, for each $j \in \{m+w+1, \ldots, m+r\}$ we define $\nu^{copy}(Y_j) := \nu^{copy}(Y_{k(j)})$. Here, by $Y_{k(j)}$, $k(j) \in \{m+1, \ldots, m+w\}$, we denote the copy variable of the representative element of the class $C'_{k(j)} \in \{C'_1, \ldots, C'_w\} \subseteq \mathcal{C}_Q$, such that the subgoal of Q having copy variable Y_j belongs to that class $C'_{k(j)}$.

It is convenient to also define here the mapping ν^{copy}_Q, to be used in Section L.4 and beyond, again for the case where $r \geq 1$. The mapping ν^{copy}_Q uses "the same logic" as the mapping ν^{copy}, except that ν^{copy}_Q maps each copy variable of Q into the variable *name* from among $N_{m+1}, \ldots, N_{m+w}$ in the vector $\bar{N}$, whereas ν^{copy} maps each copy variable of Q into the variable *value* from among $N^{(i)}_{m+1}, \ldots, N^{(i)}_{m+w}$ in the vector $\bar{N}^{(i)}$ for a fixed $i \in \mathbb{N}_+$. That is, (I) For each copy variable from among $Y_{m+1}, \ldots, Y_{m+w}$ of Q, define $\nu^{copy}_Q(Y_j) := N_j$ for each $j \in \{m+1, \ldots, m+w\}$. (II) Whenever $r > w$, for each $j \in \{m+w+1, \ldots, m+r\}$ we define $\nu^{copy}_Q(Y_j) := \nu^{copy}_Q(Y_{k(j)})$. Here, by $Y_{k(j)}$, $k(j) \in \{m+1, \ldots, m+w\}$, we denote the copy variable of the representative element of the class $C'_{k(j)} \in \{C'_1, \ldots, C'_w\} \subseteq \mathcal{C}_Q$, such that the subgoal of Q having copy variable Y_j belongs to that class $C'_{k(j)}$.

We now associate with each tuple $t \in \mathcal{S}^{(i)}$ exactly one assignment ν_t of constants to all the variables and constants of the query Q, such that (i) $\nu_t[X_j]$ coincides with $\nu_0[X_j]$ for all $j \in \{1, \ldots, l+u\}$; (ii) $\nu_t[Y_j]$ (in case $m \geq 1$ only) coincides with $\nu^{noncopy}_t[Y_j]$ for all $j \in \{1, \ldots, m\}$; (iii) for each copy variable (in case $r \geq 1$ only) from among $Y_{m+1}, \ldots, Y_{m+r}$ of Q, define $\nu_t(Y_j) := \nu^{copy}(Y_j)$ for each $j \in \{m+1, \ldots, m+r\}$; and (iv) for each constant c used in Q, $\nu_t(c) := \nu_0(c)$.

We now construct all the ground atoms in the database $D_{\bar{N}^{(i)}}(Q)$. In the construction, we use only the elements of the subset $\mathcal{S}_{C(Q)}$ of the condition of the query Q.

Main construction cycle: For each tuple t in $\mathcal{S}^{(i)}$, we add to the database $D_{\bar{N}^{(i)}}(Q)$ the following ground atoms:

- For each relational subgoal of Q of the form $b(\bar{R})$, where b is a predicate name and $\bar{R}$ is a vector of variables from among $X_1, \ldots, X_{l+u}$ and (in case $m \geq 1$) from among $Y_1, \ldots, Y_m$, and of constants from P. Add to the database $D_{\bar{N}^{(i)}}(Q)$ the ground atom $b(\nu_t(\bar{R}); 1)$ (if not already there). Note that the copy number of this ground atom equals one.
- In case $w \geq 1$: For each copy-sensitive atom from among $s(C'_1), \ldots, s(C'_w)$: Suppose the copy-sensitive atom in question is of the form $b(\bar{R}; z)$, where b is a predicate name, $\bar{R}$ is a vector of variables from among $X_1, \ldots, X_{l+u}$ and (in case $m \geq 1$) from among $Y_1, \ldots, Y_m$, and of constants from P, and z is a copy variable from among $Y_{m+1}, \ldots, Y_{m+w}$. Add to the database $D_{\bar{N}^{(i)}}(Q)$ the ground atom $b(\nu_t(\bar{R}); \nu_t(z))$ (if not already there).

Finally, $D_{\bar{N}^{(i)}}(Q)$ has no other ground atoms than those added for the tuples t in $\mathcal{S}^{(i)}$ as described above.

L.3.3 Properties of Databases $D_{\bar{N}^{(i)}}(Q)$

We now state some properties of the databases in the set $\{D_{\bar{N}^{(i)}}(Q)\}$ and of the answer to the query Q on each database. The first four results are immediate from the constructions and definitions in this section and in Section L.2.

PROPOSITION L.5. *Let $i \in \mathbb{N}_+$. (i) The set $adom(D_{\bar{N}^{(i)}}(Q))$ is the union $P \bigcup S_0 \bigcup S^{(i)}_*$. (ii) The mapping $\nu^{(i)}_Q$ is defined on all elements of $adom(D_{\bar{N}^{(i)}}(Q))$. (iii) For each subset $T \neq \emptyset$ of $adom(D_{\bar{N}^{(i)}}(Q))$ such that (in case $m \geq 1$)*

the cardinality of $T \bigcap S_j^{(i)}$ does not exceed 1 for each $j \in \{1, \ldots, m\}$, the mapping $\nu_Q^{(i)}$ is a bijection on the domain T. □

Proposition L.6. *Let $i \in \mathbb{N}_+$. For the database $D_{\bar{N}^{(i)}}(Q)$, we have that:*

(i) $D_{\bar{N}^{(i)}}(Q) \neq \emptyset$.

(ii) For each ground atom $h \in D_{\bar{N}^{(i)}}(Q)$, of the form $p(\bar{W}; n)$, for some predicate p and with copy number $n \geq 1$, there exists exactly one element, call it g, of the set $\mathcal{S}_{C(Q)}$ of (representative-element) subgoals of query Q, where the relational template of g is $p(\bar{Z})$, and such that (a) the result of applying the mapping $\nu_Q^{(i)}$ to the vector $\bar{W}$ in h is the vector $\bar{Z}$ in g, and (b) $\nu_Q^{(i)}$ is a bijection from $\bar{W}$ to $\bar{Z}$. □

For the next result, we introduce some terminology. Let $i \in \mathbb{N}_+$. For a ground atom in $h \in D_{\bar{N}^{(i)}}(Q)$ and for the corresponding atom $g \in \mathcal{S}_{C(Q)}$ as defined by Proposition L.6 (ii), we call g *the $\nu_Q^{(i)}$-image of h*. Further, by $adom(h)$ we denote all those terms of ground atom $h \in D_{\bar{N}^{(i)}}(Q)$ that are elements of $adom(D_{\bar{N}^{(i)}}(Q))$. (That is, for atom h of the form $p(\bar{W}; n)$, for some predicate p and with copy number $n \geq 1$, $adom(h)$ is the set of all elements of the vector $\bar{W}$.) Finally, in case $m \geq 1$, for a $j \in \{1, \ldots, m\}$, we denote by $S_{h,j}^{(i)}$ the intersection of the set $adom(h)$ with the set $S_j^{(i)}$.

The following result holds by construction of the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$.

Proposition L.7. *Suppose $m \geq 1$. Let $i \in \mathbb{N}_+$. Let h be an arbitrary ground atom in $D_{\bar{N}^{(i)}}(Q)$, of the form $h = p(\bar{W}; n)$, for some predicate p and with copy number $n \geq 1$. Then we have that:*

(i) For each $j \in \{1, \ldots, m\}$, the set $S_{h,j}^{(i)}$ is either the empty set or a singleton set.

(ii) In case the ground atom h is such that for at least one $j \in \{1, \ldots, m\}$, the set $S_{h,j}^{(i)}$ is not the empty set: Let μ be an arbitrary mapping from all elements of the vector $\bar{W}$ to $adom(D_{\bar{N}^{(i)}}(Q))$, such that (a) μ is the identity mapping on each element of $\bar{W}$ that does not belong to the set $S_*^{(i)}$, and such that (b) for each element e of $\bar{W}$ such that e belongs to a $S_{h,j}^{(i)}$ for some $j \in \{1, \ldots, m\}$, we have that $\mu(e)$ is an element of $S_j^{(i)}$ for the same j. *Then it holds that:*

(ii-a) The ground atom $h' = p(\mu(\bar{W}); n)$ is an element of the set $D_{\bar{N}^{(i)}}(Q)$; and

(ii-b) The $\nu_Q^{(i)}$-image of h and the $\nu_Q^{(i)}$-image of h' are the same element of the set $\mathcal{S}_{C(Q)}$.

□

Proposition L.8. *Let $i \in \mathbb{N}_+$. For the database $D_{\bar{N}^{(i)}}(Q)$, we have that:*

(i) In the construction of database $D_{\bar{N}^{(i)}}(Q)$, each main construction cycle (for some fixed tuple t in $\mathcal{S}^{(i)}$) generates in the database $D_{\bar{N}^{(i)}}(Q)$ an extended canonical database for query Q, call this database D_t. For each $t \in \mathcal{S}^{(i)}$, the mapping $\nu_Q^{(i)}$ induces an isomorphism from D_t to the set $\mathcal{S}_{C(Q)}$ of (representative-element) subgoals of the query Q.

(ii) There exists in $D_{\bar{N}^{(i)}}(Q)$ at least one extended canonical database for query Q.

(iii) For each pair (D_1, D_2) of extended canonical databases for query Q within database $D_{\bar{N}^{(i)}}(Q)$, such that each of D_1 and D_2 was generated using the main construction cycle (using some $t_1 \in \mathcal{S}^{(i)}$ to construct D_1, and using some $t_2 \in \mathcal{S}^{(i)}$ to construct D_2), the only difference (if any) between the values of variables of Q, in D_1 and D_2, is in the values of multiset noncopy variables of Q.

(iv) Let $T \neq \emptyset$ be an arbitrary subset of $adom(D_{\bar{N}^{(i)}}(Q))$ such that (a) $P \bigcup S_0$ is a subset of T, and (b) in case $m \geq 1$, the cardinality of $T \bigcap S_j^{(i)}$ is exactly 1 for each $j \in \{1, \ldots, m\}$. Then there exists in $D_{\bar{N}^{(i)}}(Q)$ an extended canonical database of the query Q such that the active domain of that extended canonical database is exactly T. □

For an $i \in \mathbb{N}_+$, let $\mathcal{T}$ be a nonempty set of ground atoms of database $D_{\bar{N}^{(i)}}(Q)$. We denote by $adom(\mathcal{T})$ the set of all values of $adom(D_{\bar{N}^{(i)}}(Q))$ that are used in the atoms of $\mathcal{T}$.

Proposition L.9. *Let $i \in \mathbb{N}_+$. Let $\mathcal{T}$ be a nonempty set of ground atoms of database $D_{\bar{N}^{(i)}}(Q)$. Suppose the set $\mathcal{T}$ is such that (in case $m = |M_{noncopy}| \geq 1$) for each $j \in \{1, \ldots, m\}$, the cardinality of the intersection of $adom(\mathcal{T})$ with $S_j^{(i)}$ is at most one.*[13] *Then there exists in $D_{\bar{N}^{(i)}}(Q)$ an extended canonical database for Q, call this database D, such that $\mathcal{T} \subseteq D$ (as set of ground atoms* with copy numbers*).* □

Proof. For each ground atom h in $\mathcal{T}$, we label h with the subgoal s of Q such that s "has generated" h in the construction of database $D_{\bar{N}^{(i)}}(Q)$. (See Proposition L.6 (ii) for the details.) We then partition all the atoms of $\mathcal{T}$ into equivalence classes, call them $\mathcal{E}_1, \ldots, \mathcal{E}_n$, $n \geq 1$, using the labels. (That is, two atoms of $\mathcal{T}$ belong in the same equivalence class if and only if they have the same label.) Now we show that for each possible label (w.r.t. query Q), the equivalence class for $\mathcal{T}$ and for this label has at most one element. Indeed, recall that for each $j \in \{1, \ldots, m\}$, the cardinality of the intersection of $adom(\mathcal{T})$ with $S_j^{(i)}$ is at most one. By construction of the database $D_{\bar{N}^{(i)}}(Q)$ and of the equivalence classes for $\mathcal{T}$, we obtain the desired result.

Now we use the classes $\mathcal{E}_1, \ldots, \mathcal{E}_n$ to construct from $\mathcal{T}$ some of the subgoals of Q. By the properties of $adom(\mathcal{T})$ and of the mapping $\nu_Q^{(i)}$, as well as from the fact that each of $\mathcal{E}_1, \ldots, \mathcal{E}_n$ is a singleton set, we obtain that the mapping $\nu_Q^{(i)}$ induces an isomorphism from the set $\mathcal{T}$ to a nonempty subset, call it $\mathcal{S}$, of the set $\mathcal{S}_{C(Q)}$ of (representative-element) subgoals of the query Q. Denote by $adom(\mathcal{S})$ the set of all terms of the query Q that are not copy variables of Q, such that these terms are used in the atoms of $\mathcal{S}$. By construction, the mapping $(\nu_Q^{(i)})^{-1}$ is a bijection from $adom(\mathcal{S})$ to $adom(\mathcal{T})$. By definition of the main construction cycle in constructing the database $D_{\bar{N}^{(i)}}(Q)$, at least one iteration of the main construction cycle in the construction has used (some extension of) the mapping $(\nu_Q^{(i)})^{-1}$, to generate in $D_{\bar{N}^{(i)}}(Q)$ an extended canonical database for Q. (See Proposition L.8 for the justifications.) As that iteration of the main construction cycle mapped the set $\mathcal{S}_{C(Q)}$ of (representative-element) subgoals of the query Q to a superset of the set $\mathcal{T}$, Q.E.D. □

[13]In case $m = 0$, we require only that $\mathcal{T} \neq \emptyset$.

We now establish that the fixed tuple $t^*_Q = \nu_0[\bar{X}]$ (where $\bar{X}$ is the vector of head terms of the query Q) is present in $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$, for each $i \geq 1$.

Proposition L.10. *Let $i \in \mathbb{N}_+$. Then the bag Res_C $(Q, D_{\bar{N}^{(i)}}(Q))$ has at least one copy of the tuple t^*_Q.* □

Proof. (sketch) This result holds by construction of the databases in the family $\{D_{\bar{N}^{(i)}}(Q)\}$. Indeed, recall that the assignment mapping ν_0 is fixed to be the same across all the databases in $\{D_{\bar{N}^{(i)}}(Q)\}$. Choose an arbitrary tuple t in the set $\mathcal{S}^{(i)}$ for $D_{\bar{N}^{(i)}}(Q)$; consider the mapping ν_t associated with t. It is easy to verify that ν_t is an assignment mapping from the query Q to the database $D_{\bar{N}^{(i)}}(Q)$, such that ν_t contributes the tuple $t^*_Q = \nu_0[\bar{X}]$ to the bag $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$. □

L.4 t^*_Q-Valid Assignment Mappings for Query Q'' and Databases in $\{D_{\bar{N}^{(i)}}(Q)\}$

Consider the family $\{D_{\bar{N}^{(i)}}(Q)\}$ of databases constructed as described in Section L.3. Let Q'' be an arbitrary CCQ query that satisfies all the requirements (for Q'' specifically) of Section L.2.1. In this section, for the query Q'' and for an arbitrary $i \in \mathbb{N}_+$, we provide an algorithm for generating the set of all assignment mappings from Q'' to the database $D_{\bar{N}^{(i)}}(Q)$ such that each of the assignments "generates"[14] the fixed tuple t^*_Q (as defined in Section L.3) in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$. We call these assignment mappings "t^*_Q-valid assignment mappings". We also formulate some useful properties of the t^*_Q-valid assignment mappings, for Q'' and for each $i \in \mathbb{N}_+$.

L.4.1 Definitions and Construction

The basics.

Denote by $G > 0$ the number of subgoals, call them $g_1, \ldots, g_G$, of the query Q''. For convenience, we choose the ordering $g_1, \ldots, g_G$ of the subgoals of Q'' in such a way that all the copy-sensitive atoms, if any, in the ordering precede all the relational atoms, if any, in the ordering, and such that, in case $r \geq 1$, each jth atom g_j in the ordering has copy variable Y''_{m+j} of the query Q'', $1 \leq j \leq r$. (See Section L.2.1 for the notation Y''_j.) Fix an arbitrary $i \in \mathbb{N}_+$, and denote by $F > 0$ the number of ground atoms in the database $D_{\bar{N}^{(i)}}(Q)$.

We find all the t^*_Q-valid assignment mappings for Q'' and $D_{\bar{N}^{(i)}}(Q)$ (i.e., those satisfying assignments, in the terminology of Section 2.1.2, that are in the set $\Gamma^{(t^*_Q)}$ $(Q'', D_{\bar{N}^{(i)}}(Q)))$ by enumerating *all* associations between the G subgoals of Q'' and the F ground atoms in $D_{\bar{N}^{(i)}}(Q)$. The definition of "valid assignment mapping" is "as expected". However, since we use associations between atoms to determine which assignment mappings are valid, we provide here the required formal definitions.

Definition L.1. *(***Association for Q and D***) Given a CCQ query Q with $G > 0$ subgoals and a nonempty database D, a set of G pairs $\{(g_1, d_{j1}), (g_2, d_{j2}), \ldots, (g_G, d_{jG})\}$ between all the G subgoals of Q and some (not necessarily distinct) G ground atoms of D is called an* association for Q and D. □

[14]That is, each of these assignments generates a separate element of the set $\Gamma^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$.

Definition L.2. *(***Candidate assignment mapping***) Given a CCQ query Q with $G > 0$ subgoals, a nonempty database D, and an association $\mathcal{A} = \{(g_1, d_{j1}), (g_2, d_{j2}), \ldots, (g_G, d_{jG})\}$ for Q and D. Define a* candidate assignment mapping θ for Q and D w.r.t. $\mathcal{A}$ *as follows:*

(a) θ is the empty mapping in case there exists an integer k, $1 \leq k \leq G$, such that g_k and d_{jk} have different predicate names, and

(b) θ is the union of the G associations of the terms of each g_k, $1 \leq k \leq G$, to the terms of its respective d_{jk}, where each association is a set of assignments of the values in $\bar{W}^{(k)}$, from left to right, to the terms in $\bar{Z}^{(k)}$ in the same (from left to right) positions. Here, $\bar{Z}^{(k)}$ is the vector of all terms including the copy variable *(if any) in g_k, and $\bar{W}^{(k)}$ is the vector of all arguments of d_{jk}. We do not include the copy number of d_{jk}* per se *as an element of $\bar{W}^{(k)}$. Instead, we do the following. In case g_k is a relational atom, the copy number of d_{jk} is not used in the construction of θ for g_k. If g_k is a copy-sensitive atom, then we consider the copy variable of g_k to be the last element of vector $\bar{Z}^{(k)}$, and add to the vector $\bar{W}^{(k)}$, as its rightmost element, a natural-number value between (inclusively) 1 and the copy number of d_{jk}.* □

Note that whenever query Q has copy variables *and* database D has ground atoms with nonunity copy numbers, for a single association $\mathcal{A}$ for Q and D there could be more than one (but always a finite number, on finite databases) candidate assignment mappings for Q and D w.r.t. $\mathcal{A}$. Definition L.2 provides a straightforward algorithm to generate *all* candidate assignment mappings for any CCQ query Q, finite database D, and association $\mathcal{A}$ for Q and D.

Definition L.3. *((***t^*_Q-)Valid assignment mapping***) Given a CCQ query Q, a nonempty database D, an association $\mathcal{A}$ for Q and D, and a candidate assignment mapping θ for Q and D w.r.t. $\mathcal{A}$. Then θ is a* valid assignment mapping from the variables and constants of Q to the values in $adom(D) \cup \mathbb{N}_+$ w.r.t. $\mathcal{A}$ *if: (i) θ is not an empty mapping; (ii) θ associates all occurrences of each constant of Q with the same constant; and (iii) for each variable of Q, θ associates all occurrences of the variable with the same value in $adom(D) \cup \mathbb{N}_+$. A valid assignment mapping θ is a* t^*_Q-valid assignment mapping *for Q, D, and $\mathcal{A}$ if the restriction of θ to the head vector of the query Q results in the tuple t^*_Q.* □

For brevity, we will refer to each valid assignment mapping from the variables and constants of Q to the values in $adom(D) \cup \mathbb{N}_+$ w.r.t. $\mathcal{A}$, for some Q, D, and $\mathcal{A}$, as a "valid assignment mapping for Q, D, and $\mathcal{A}$". In addition, we say that θ is a "valid assignment mapping for Q and D" if there exists an association, $\mathcal{A}$, for Q and D, such that θ is a valid assignment mapping for Q, D, and $\mathcal{A}$.

By definition, each valid assignment mapping for Q and D is an element of the set $\Gamma(Q, D)$, and each t^*_Q-valid assignment mapping for Q and D is an element of the set $\Gamma^{(t^*_Q)}(Q, D)$. If θ is a valid assignment mapping for query Q, database D, and association $\mathcal{A}$ (for Q and D), then we say that $\mathcal{A}$ *generates the mapping θ*, or that $\mathcal{A}$ *contributes the mapping θ* to the set $\Gamma(Q, D)$.

Definition L.4. *(***Unity (t^*_Q-)valid assignment mapping***) Given a CCQ query Q, a nonempty database D, an* *association $\mathcal{A}$ for Q and D, and a (t^*_Q-)valid assignment mapping θ for Q, D, and $\mathcal{A}$. Then θ is a* unity (t^*_Q-)valid assignment mapping *for Q, D, and $\mathcal{A}$ if θ maps each (if any) copy variable of the query Q into value* 1. □

EXAMPLE L.2. *Let CCQ query Q'' be defined as*

$$Q''(X) \leftarrow p(X,Y),\ p(X,X;i),\ \{Y,i\}.$$

Let g_1 be the rightmost subgoal of Q'' (that is, $p(X,X;i)$); and let g_2 be the leftmost subgoal of Q'' (that is, $p(X,Y)$); then G for Q'' equals 2.

For ease of exposition, assume that the database, call it D, is $D = \{p(1,1;3), p(1,2;5), p(3,3;7)\}$. We refer to the ground atom $p(1,1;3)$ of D as d_1, to the ground atom $p(1,2;5)$ of D as d_2, and to the ground atom $p(3,3;7)$ of D as d_3.

Consider three (from the total of nine possible) associations between the subgoals of the query Q'' and the ground atoms of the database D:

- *$\mathcal{A}_1 : \{(g_1,d_2),(g_2,d_1)\}$;*
- *$\mathcal{A}_2 : \{(g_1,d_1),(g_2,d_1)\}$; and*
- *$\mathcal{A}_3 : \{(g_1,d_3),(g_2,d_3)\}$.*

It is easy to see that each candidate assignment mapping associated with $\mathcal{A}_1$ is not a valid assignment mapping, as each such candidate mapping maps the two occurrences of X in g_1 into distinct values (1 and 2) in $adom(D)$.

At the same time:

- *Given $\mathcal{A}_2$, $\theta_{21} = \{X \rightarrow 1, Y \rightarrow 1, i \rightarrow 1\}$ is a unity valid assignment mapping from the terms of Q'' to the elements of $adom(D) \cup \mathbb{N}_+$.*
- *Given $\mathcal{A}_3$, $\theta_3 = \{X \rightarrow 3, Y \rightarrow 3, i \rightarrow 7\}$ is a valid assignment mapping from the terms of Q'' to the elements of $adom(D) \cup \mathbb{N}_+$; it is not a* unity *valid assignment mapping.*

Now observe that in addition to θ_{21}, two more candidate mappings w.r.t. $\mathcal{A}_2$ would also be valid assignment mappings from the terms of Q'' to the elements of $adom(D) \cup \mathbb{N}_+$. These mappings are $\theta_{22} = \{X \rightarrow 1, Y \rightarrow 1, i \rightarrow 2\}$ and $\theta_{23} = \{X \rightarrow 1, Y \rightarrow 1, i \rightarrow 3\}$. The only difference between θ_{21}, θ_{22}, and θ_{23} is in the value assigned to the copy variable i of the query Q''. Unlike θ_{21}, neither θ_{22} nor θ_{23} is a unity valid assignment mapping for Q'', D, and $\mathcal{A}_2$. No other candidate mappings are possible for Q'', D, and $\mathcal{A}_2$, because the copy number of d_1 in D is three.

Finally, suppose we are given tuple $t^*_Q = (1)$. Then θ_{21} is a (unity) t^*_Q-valid assignment mapping for Q'', D, and $\mathcal{A}_2$, because $\theta_{21}[X]$ coincides with t^*_Q. At the same time, while being a valid assignment mapping from Q'' to D, θ_3 is not *a t^*_Q-valid assignment mapping for Q'', D, and $\mathcal{A}_3$, because $\theta_3[X] = 3$ does not coincide with t^*_Q.* □

Signatures of associations.

Fix an $i \in \mathbb{N}_+$. Recall Proposition L.6 (ii), which says that by construction of the database $D_{\bar{N}^{(i)}}(Q)$, each ground atom in the database can be "mapped into" a *unique* subgoal of the subset $\mathcal{S}_{C(Q)}$ of the condition of the fixed input query Q, using the mapping $\nu^{(i)}_Q$ defined in Section L.3.1. We denote by $\psi^{gen(Q)}_{\bar{N}^{(i)}}$ this mapping, induced by the mapping $\nu^{(i)}_Q$, from the ground atoms of $D_{\bar{N}^{(i)}}(Q)$ to the elements of the set $\mathcal{S}_{C(Q)}$.

DEFINITION L.5. *(***Atom-signature of association***) Let $i \in \mathbb{N}_+$, and let $\mathcal{A} = \{(g_1,d_{j1}),\ldots,(g_G,d_{jG})\}$ be an association for query Q'', with $G \geq 1$ subgoals, and for the database $D_{\bar{N}^{(i)}}(Q)$. Then the G-ary vector $\Psi_a[\mathcal{A}] = [\ \psi^{gen(Q)}_{\bar{N}^{(i)}}[d_{j1}]$, $\psi^{gen(Q)}_{\bar{N}^{(i)}}[d_{j2}]$, $\ldots$, $\psi^{gen(Q)}_{\bar{N}^{(i)}}[d_{jG}]\ ]$ is the* atom-signature of *$\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$.* □

Please see Example L.3 for an extended illustration of atom-signatures of associations.

Recall the notation $Y''_1,\ldots,Y''_{m+r}$ of Section L.2.1 for the multiset variables of the query Q''. Here, $Y''_1,\ldots,Y''_m$ are all the multiset noncopy variables of Q'' (in case $m \geq 1$), and $Y''_{m+1},\ldots,Y''_{m+r}$ are all the copy variables of Q'' (in case $r \geq 1$).

Suppose that the query Q is such that $r = |M_{copy}| \geq 1$. Recall the mapping ν^{copy}_Q defined in Section L.3.2: ν^{copy}_Q maps each copy variable of the query Q into one of the variables $N_{m+1},\ldots,N_{m+w}$ in the vector $\bar{N}$.

We now define the following mapping $\nu^{copy-sig}$ on all atoms in the set $\mathcal{S}_{C(Q)}$ for the query Q:

- For each (if any) relational atom h in $\mathcal{S}_{C(Q)}$, $\nu^{copy-sig}(h)$:= 1.
- For each (if any) copy-sensitive atom h in $\mathcal{S}_{C(Q)}$, where h has copy variable of the name $Z \in M_{copy}$, $\nu^{copy-sig}(h)$:= $\nu^{copy}_Q(Z)$.

Recall (see beginning of Section L.4.1) that we have fixed an ordering of the subgoals $g_1,\ldots,g_G$ of the query Q'' in such a way that:

- all the copy-sensitive atoms in the ordering precede all the relational atoms, if any, in the ordering, and
- each jth atom g_j in the ordering has copy variable Y''_{m+j} of the query Q'', $1 \leq j \leq r$ (in case $r \geq 1$).

DEFINITION L.6. *(***Copy-signature of association***) Let $i \in \mathbb{N}_+$, and let $\mathcal{A} = \{(g_1,d_{j1}),\ \ldots,\ (g_G,d_{jG})\}$ be an association for query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$. Then vector $\Phi_c[\mathcal{A}]$, called the* copy-signature of *$\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$, is defined as follows:*

- *In case $r = 0$, $\Phi_c[\mathcal{A}]$ is the empty vector; and*
- *In case $r \geq 1$, $\Phi_c[\mathcal{A}]$ is the r-ary vector $[\ \nu^{copy-sig}$ $(\psi^{gen(Q)}_{\bar{N}^{(i)}}[d_{j1}])$, $\ldots$, $\nu^{copy-sig}$ $(\psi^{gen(Q)}_{\bar{N}^{(i)}}[d_{jr}])\ ]$.*

□

DEFINITION L.7. *(***Noncopy-signature of association***) Let $i \in \mathbb{N}_+$, and let $\mathcal{A} = \{(g_1,d_{j1}),\ \ldots,\ (g_G,d_{jG})\}$ be an association for query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, such that there exists a valid assignment mapping, θ, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$. Then vector $\Phi_n[\mathcal{A}]$, called* noncopy-signature of *$\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$, is defined as follows:*

- *In case $m = 0$, $\Phi_n[\mathcal{A}]$ is the empty vector; and*
- *In case $m \geq 1$, $\Phi_n[\mathcal{A}]$ is the m-ary vector $[\ \nu^{(i)}_Q\ (\theta(Y''_1))$, $\ldots$, $\nu^{(i)}_Q\ (\theta(Y''_m))\ ]$.*

□

In Section L.6 we will find a very practical use for copy-signatures and for noncopy-signatures of associations, in that these signatures of association $\mathcal{A}$ will help us compute the number of distinct entries, in the set $\Gamma^{(t^*_Q)}_{\bar{S}}(Q'', D_{\bar{N}^{(i)}}(Q))$, that (entries) are contributed by the (valid) mappings of $\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$. Please see Example L.3 for an extended illustration of copy-signatures and of noncopy-signatures of associations.

The intuition for copy-signature is that a copy variable, Z, of the query Q'' gets mapped into constant 1 in case the subgoal (of Q'') that has Z gets mapped by the association $\mathcal{A}$ into a ground atom of the database that (ground atom) has been generated from a *relational* subgoal of the query Q. Conversely, a copy variable, Z, of the query Q'' gets mapped into variable N_j, for some $j \in \{m+1, \ldots, m+w\}$, of vector $\bar{N}$, in case the subgoal (of Q'') that has Z gets mapped by the association $\mathcal{A}$ into a ground atom of the database that (ground atom) has been generated from a *copy-sensitive* subgoal h of the query Q such that the copy variable of h corresponds to the variable N_j (via mapping ν_Q^{copy}).

The intuition for noncopy-signature is that it shows, for a given valid assignment θ, the correspondences that θ induces from the multiset noncopy variables of the query Q'' to the terms (represented in the database by rigidly delineated sets of values) of the query Q used to construct the database.

The following result is immediate from the definitions.

Proposition L.11. *Let $i \in \mathbb{N}_+$, and let $\mathcal{A}$ be an association for query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, such that there exists a valid assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$. Then the noncopy-signature of $\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$ exists and is unique.* □

Proposition L.11 lets us refer to *the* noncopy-signature of a given association, for some query and database, provided that the association generates at least one valid assignment mapping.

L.4.2 Properties of Associations for Q'' and $D_{\bar{N}^{(i)}}(Q)$

The results of Propositions L.12 through L.17 are immediate from the definitions (unless discussed further in this subsection).

Proposition L.12. *Let $i \in \mathbb{N}_+$, and let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two associations for query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, such that $\mathcal{A}_1$ and $\mathcal{A}_2$ have the same atom-signature. Then:*

(i) $\mathcal{A}_1$ and $\mathcal{A}_2$ have the same copy-signature.

(ii) Suppose that, in addition, there exists a valid assignment mapping, call it θ_1, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_1$. Let $t_{(\theta_1)}$ be the tuple resulting from restricting θ_1 to the head vector of the query Q''. Then we have that:

(ii-a) There exists a valid assignment mapping, call it θ_2, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_2$;

(ii-b) $\mathcal{A}_1$ and $\mathcal{A}_2$ have the same noncopy-signature; and

(ii-c) If $t_{(\theta_1)}$ is the tuple t^*_Q, then restricting θ_2 to the head vector of the query Q'' results in the same *tuple $t_{(\theta_1)}$ (which is t^*_Q).*

□

The following result holds due to our convention for ground atoms in databases, see Section L.2.2. Note that the associations $\mathcal{A}_1$ and $\mathcal{A}_2$, in the statement of Proposition L.13, are not restricted to have either the same atom-signature or different atom-signatures.

Proposition L.13. *Let $i \in \mathbb{N}_+$, and let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two distinct associations for query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$. Let (θ_1, θ_2) be an arbitrary pair (if any exists), such that θ_1 is a valid assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_1$, and θ_2 is a valid assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_2$. Then there exists a variable X of Q'' such that (i) X is not a copy variable of Q'', and (ii) $\theta_1(X) \neq \theta_2(X)$.* □

Proposition L.14. *Let $i \in \mathbb{N}_+$, and let $\mathcal{A}$ be an association for query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, such that there exists a valid assignment mapping, θ, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$. Let tuple t_θ be the result of restricting θ to the head vector of the query Q''. Then there exists a* unity *valid assignment mapping, call it $\theta^{(u)}$, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$, such that the restriction of $\theta^{(u)}$ to the head vector of the query Q'' results in the* same *tuple t_θ.* □

Proposition L.15. *Let $i \in \mathbb{N}_+$, and let $\mathcal{A}$ be an association for query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, such that there exists a unity valid assignment mapping, θ, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$. Then there does not exist any unity valid assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$, that (unity valid assignment mapping) would be distinct from θ.* □

Proposition L.16. *Let $i \in \mathbb{N}_+$, and let $\mathcal{A}$ be an association for query $Q''$and for the database $D_{\bar{N}^{(i)}}(Q)$. Suppose $\mathcal{A}$ is such that there exists a unity valid assignment mapping, $\theta^{(u)}$, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$. Let tuple $t_{\theta^{(u)}}$ be the restriction of $\theta^{(u)}$ to the head vector of the query Q''. Then we have that for each candidate assignment mapping, call it θ', for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$, θ' is a valid assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$, and the restriction of θ' to the head vector of the query Q'' is the tuple $t_{\theta^{(u)}}$.* □

For the next result we introduce some notation. Let $i \in \mathbb{N}_+$, and let $\mathcal{A}$ be an association for query $Q''$and for the database $D_{\bar{N}^{(i)}}(Q)$. In case $r \geq 1$, denote the entries in the copy-signature $\Phi_c[\mathcal{A}]$ as $\Phi_c[\mathcal{A}] = [V_{j1}, \ldots, V_{jr}]$. Here, for each $k \in \{1, \ldots, r\}$ we have that $V_{jk} \in \{1,\ N_{m+1},\ \ldots,\ N_{m+w}\}$, where the N-values are variables in the vector $\bar{N}$. (In case $r = 0$, $\Phi_c[\mathcal{A}]$ is the empty vector by definition.) Further, again for each $k \in \{1, \ldots, r\}$, by $V_{jk}^{(i)}$ we denote the value of V_{jk} in the vector $\bar{N}^{(i)}$ whenever $V_{jk} \in \{N_{m+1}, \ldots, N_{m+r}\}$; we set $V_{jk}^{(i)} := 1$ for the case $V_{jk} = 1$.

We also use the notation $\Gamma^{(\mathcal{A})}$ (for all cases $r \geq 0$ of the value $r = |M_{copy}|$): For an $i \in \mathbb{N}_+$ and an association $\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$, $\Gamma^{(\mathcal{A})}$ stands for the set of all tuples contributed to the set $\Gamma_{\bar{S}}^{(t_{\theta^{(u)}})}(Q'', D_{\bar{N}^{(i)}}(Q))$, for some tuple $t_{\theta^{(u)}}$, by the valid assignment mappings (if any) for the association $\mathcal{A}$. (By Proposition L.16, all the valid assignment mappings (if any) for an association $\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$, agree on their restriction to the head vector of the query Q''.) Finally, in case $r \geq 1$, we denote by $\Gamma_c^{(\mathcal{A})}$ the set projection of the set $\Gamma^{(\mathcal{A})}$ on the columns $Y''_{m+1}, \ldots, Y''_{m+r}$, for all the copy variables of the query Q''.

Proposition L.17. *Let $i \in \mathbb{N}_+$, and let $\mathcal{A}$ be an association for query $Q''$and for the database $D_{\bar{N}^{(i)}}(Q)$. Suppose $\mathcal{A}$ is such that there exists a unity valid assignment mapping, $\theta^{(u)}$, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$. Let tuple $t_{\theta^{(u)}}$ be the restriction of $\theta^{(u)}$ to the head vector of the query Q''. Then the following holds:*

(i) The set $\Gamma^{(\mathcal{A})}$ is not empty;

(ii) In case $r = 0$, the association $\mathcal{A}$ has exactly one valid assignment mapping and contributes exactly one tuple to the set $\Gamma_{\bar{S}}^{(t_{\theta(u)})}(Q'', D_{\bar{N}^{(i)}}(Q))$;

(iii) In case $r \geq 1$, the set $\Gamma_c^{(\mathcal{A})}$ has the tuple $(n_1, n_2, \ldots, n_r)$ for each $1 \leq n_k \leq V_{jk}^{(i)}$ for each $k \in \{1, \ldots, r\}$, and $\Gamma_c^{(\mathcal{A})}$ does not have any other tuples;

(iv) The total number of distinct *valid assignment mappings for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$, call it $T^{(\mathcal{A})}$, is 1 in case $r = 0$, and is $\Pi_{k=1}^{r} V_{jk}^{(i)}$ in case $r \geq 1$; and*

(v) The cardinality of the set $\Gamma^{(\mathcal{A})}$ is equal to $T^{(\mathcal{A})}$.

□

L.5 Sets of Associations for Q'' and $D_{\bar{N}^{(i)}}(Q)$

Consider each of the F^G associations, $\mathcal{A}_1, \ldots, \mathcal{A}_{F^G}$, of the form $\mathcal{A}$ as defined in Section L.4, between the $G \geq 1$ subgoals $g_1, \ldots, g_G$ of the fixed query Q'' and the $F \geq 1$ ground atoms of the database $D_{\bar{N}^{(i)}}(Q)$, for an arbitrary fixed $i \in \mathbb{N}_+$. Clearly, enumerating all the F^G associations is a way to find *all* satisfiable assignments for Q'' and $D_{\bar{N}^{(i)}}(Q)$. In this section we construct a set, $\mathbb{A}_{Q''}^{(i)}$, which includes some of the above associations, and "captures", in a very precise sense (see Proposition L.18), all of the t_Q^*-valid assignment mappings for Q'' and $D_{\bar{N}^{(i)}}(Q)$. In Section L.6 we will use the set $\mathbb{A}_{Q''}^{(i)}$ in our undertaking to define a function, $\mathcal{F}_{(Q)}^{(Q'')}$, in terms of the variables in the vector $\bar{N}$. For each $i \in \mathbb{N}_+$, the function $\mathcal{F}_{(Q)}^{(Q'')}$ uses the values (in $\bar{N}^{(i)}$) of the variables in the vector $\bar{N}$ to return the multiplicity of the tuple t_Q^* in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$.

DEFINITION L.8. *(***Set $\mathbb{A}_{Q''}^{(i)}$ of t_Q^*-valid assignment mappings for Q'' and $D_{\bar{N}^{(i)}}(Q)$***) Let $i \in \mathbb{N}_+$.* The set $\mathbb{A}_{Q''}^{(i)}$ of t_Q^*-valid assignment mappings for CCQ query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, *is the set of all of the associations for Q'' and $D_{\bar{N}^{(i)}}(Q)$, such that for each $\mathcal{A} \in \mathbb{A}_{Q''}^{(i)}$ there exists at least one t_Q^*-valid assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}$.* □

For each $i \in \mathbb{N}_+$, we denote the cardinality of the set $\mathbb{A}_{Q''}^{(i)}$ by $R_{Q''}^{(i)}$. Further, whenever $R_{Q''}^{(i)} \geq 1$, we refer to the individual elements of the set $\mathbb{A}_{Q''}^{(i)}$ as $A_j^{(i)}$, for $1 \leq j \leq R_{Q''}^{(i)}$. (We will avoid the confusion as to which query $A_j^{(i)}$ "refers to", by always using the notation $A_j^{(i)}$ in the context of exactly one query.)

Consider an arbitrary set $\mathbb{A}_{Q''}^{*}$ of associations for query Q'' and for database $D_{\bar{N}^{(i)}}(Q)$. Let $\mathcal{A}$ be an association for Q'' and $D_{\bar{N}^{(i)}}(Q)$ such that there exists a valid assignment mapping, θ, for Q'', for $D_{\bar{N}^{(i)}}(Q)$, and for $\mathcal{A}$. Then we say that the set $\mathbb{A}_{Q''}^{*}$ *captures the valid assignment mapping* θ if we have that $\mathcal{A} \in \mathbb{A}_{Q''}^{*}$. (See Proposition L.13 for a justification of this definition.)

PROPOSITION L.18. *Let $i \in \mathbb{N}_+$. Then (i) The set $\mathbb{A}_{Q''}^{(i)}$ of t_Q^*-valid assignment mappings for CCQ query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$ captures all the t_Q^*-valid assignment mappings for Q'' and $D_{\bar{N}^{(i)}}(Q)$; and (ii) For each valid assignment mapping θ for Q'' and $D_{\bar{N}^{(i)}}(Q)$ such that $\mathbb{A}_{Q''}^{(i)}$ captures θ, θ is a t_Q^*-valid assignment mapping for Q'' and $D_{\bar{N}^{(i)}}(Q)$.* □

PROPOSITION L.19. *Suppose that there exists an $i^* \in \mathbb{N}_+$ such that for some association $\mathcal{A}_{j^*}^{(i^*)} \in \mathbb{A}_{Q''}^{(i^*)}$, a valid assignment mapping for $\mathcal{A}_{j^*}^{(i^*)}$ induces a mapping from all the subgoals of Q'' to a* single *extended canonical database for query Q.*[15] *Then for* each *$i \in \mathbb{N}_+$, there exists a j, $1 \leq j \leq R_{Q''}^{(i)}$, such that a valid assignment mapping for the association $\mathcal{A}_j^{(i)}$ (exists and) induces a mapping from all the subgoals of the query Q'' to a* single *extended canonical database for query Q (within database $D_{\bar{N}^{(i)}}(Q)$). Moreover, $\mathcal{A}_j^{(i)}$ and $\mathcal{A}_{j^*}^{(i^*)}$ have the same atom-signature.* □

PROOF. We are given that there exists a pair (i^*, j^*), with $i^* \in \mathbb{N}_+$ and with $1 \leq j^* \leq R_{Q''}^{(i^*)}$, such that the association $\mathcal{A}_{j^*}^{(i^*)}$ generates a valid assignment mapping from query Q'' to a single extended canonical database, call it D^*, for query Q (within database $D_{\bar{N}^{(i^*)}}(Q)$). By Proposition L.14 in Section L.4.2, there exists a unity valid assignment mapping, call it θ', for Q'' and $D_{\bar{N}^{(i^*)}}(Q)$, such that θ' uses only the atoms of the database D^* to generate the tuple t_Q^* in the answer to Q'' on database $D_{\bar{N}^{(i^*)}}(Q)$.

Now fix an abitrary $i \in \mathbb{N}_+$. By Proposition L.8 in Section L.3.3, database $D_{\bar{N}^{(i)}}(Q)$ has at least one extended canonical database for query Q. Choose and fix in $D_{\bar{N}^{(i)}}(Q)$ one arbitrary such extended canonical database for Q, call this database D. By definition of extended canonical database, there is an isomorphism from all the ground atoms of the database D^* (within $D_{\bar{N}^{(i^*)}}(Q)$, see first paragraph of this proof), to all the ground atoms of database D (within $D_{\bar{N}^{(i)}}(Q)$). Moreover, there exists at least one isomorphism from D^* to D, call this isomorphism ι^*, such that for each atom d^* in D^*, it holds that $\psi_{\bar{N}^{(i)}}^{gen(Q)}[d^*]$ is the same (atom in set $\mathcal{S}_{C(Q)}$) as $\psi_{\bar{N}^{(i)}}^{gen(Q)}[\iota^*(d^*)]$. Then the composition, call it φ, of θ' (of the first paragraph of this proof) with ι^* gives us a valid assignment mapping from query Q'' to the extended canonical database D for query Q in database $D_{\bar{N}^{(i)}}(Q)$, such that the restriction of φ to the head variables of Q'' is the tuple t_Q^*. By Proposition L.18, the association represented by φ must be in the set $\mathbb{A}_{Q''}^{(i)}$ for database $D_{\bar{N}^{(i)}}(Q)$. By construction, this association and $\mathcal{A}_{j^*}^{(i^*)}$ have the same atom-signature. By the fact that i has been chosen arbitrarily, Q.E.D. □

PROPOSITION L.20. *Suppose that, for some $i \in \mathbb{N}_+$, there exists an association $\mathcal{A}_j^{(i)}$ in the set $\mathbb{A}_{Q''}^{(i)}$ such that the unity valid assignment mapping, θ_j, for $\mathcal{A}_j^{(i)}$ induces a mapping from the subgoals of Q'' into elements of two or more extended canonical databases of Q in database $D_{\bar{N}^{(i)}}(Q)$. Then there exists an association $\mathcal{A}_{j'}^{(i)}$ in $\mathbb{A}_{Q''}^{(i)}$, and there exists in $D_{\bar{N}^{(i)}}(Q)$ an extended canonical database of Q, call this database D^*, such that the unity valid assignment mapping, $\theta_{j'}$, for $\mathcal{A}_{j'}^{(i)}$ induces a mapping from all the subgoals of Q'' into ground atoms belonging to D^* only. Moreover, $\mathcal{A}_j^{(i)}$ and $\mathcal{A}_{j'}^{(i)}$ have the same atom-signature.* □

PROOF. We observe first that if $M_{noncopy} = \emptyset$ then database $D_{\bar{N}^{(i)}}(Q)$ comprises exactly one extended canonical database for Q. This observation is immediate from the construction of $D_{\bar{N}^{(i)}}(Q)$.

[15]That extended canonical database for Q is within database $D_{\bar{N}^{(i^*)}}(Q)$, see Proposition L.8 in Section L.3.3.

Thus we assume for the remainder of this proof that $m = |M_{noncopy}| \geq 1$. For the association $\mathcal{A}_j^{(i)}$ as in the statement of this Observation, denote by (set of ground atoms) $\mathcal{T}$ the image of the condition of query Q'' under the mapping θ_j. By Proposition L.9, the only way the valid assignment mapping θ_j for $\mathcal{A}_j^{(i)}$ can map the subgoals of Q'' into elements of two or more extended canonical databases of Q in database $D_{\bar{N}^{(i)}}(Q)$ is when the result of intersecting $adom(\mathcal{T})$ with $S_l^{(i)}$, for at least one $l \in \{1, \ldots, m\}$, has size two or more. (For the notation $adom(\mathcal{T})$, see note before Proposition L.9.)

We show an algorithm for producing the association $\mathcal{A}_{j'}^{(i)}$ and the (extended canonical) database D^*, of the statement of this Proposition, from the association $\mathcal{A}_j^{(i)}$. First, for each $l \in \{1, \ldots, m\}$ such that the result of intersecting $adom(\mathcal{T})$ with $S_l^{(i)}$ is not empty, fix a single value in that intersection. Let the result be values $v_{l_1}^{(i)}, \ldots, v_{l_k}^{(i)}$, where: $1 \leq k \leq m$, all the values $l_1, \ldots, l_k$ are distinct, each l_n (for all $1 \leq n \leq k$) satisfies $1 \leq l_n \leq m$, and each $v_{l_n}^{(i)}$ (for all $1 \leq l_n \leq m$, for all $1 \leq n \leq k$) satisfies $v_{l_n}^{(i)} \in S_{l_n}^{(i)}$. Call all values in the set $E = ((\bigcup_{n=1}^{m} S_n^{(i)}) \bigcap adom(\mathcal{T})) - \{v_{l_1}^{(i)}, \ldots, v_{l_k}^{(i)}\}$ the "extra multiset noncopy" values in $adom(\mathcal{T})$. By the assumptions in the statement of this Proposition, the set E is not empty.

The next step of the algorithm is to modify the mapping θ_j for the association $\mathcal{A}_j^{(i)}$, by replacing in θ_j each value v belonging to $S_n^{(i)} \bigcap E$, $1 \leq n \leq m$, by the value $v_n^{(i)}$ that we fixed as described in the previous paragraph. (The intuition is that for each mapping in θ_j of a variable of Q'' to an "extra multiset noncopy" value, we "redirect" the mapping to a mapping of the same variable into an "appropriate" value from among the values $v_{l_1}^{(i)}, \ldots, v_{l_k}^{(i)}$ fixed in the previous paragraph.) As we modify θ_j this way, we also modify the association $\mathcal{A}_j^{(i)}$, by replacing in it all occurrences of each $v \in S_n^{(i)} \bigcap E$, $1 \leq n \leq m$, by the value $v_n^{(i)}$. Denote by $\theta_{j'}$ the result of this modification of θ_j, and by $\mathcal{A}_{j'}^{(i)}$ the result of this modification of $\mathcal{A}_j^{(i)}$. By construction, we have that:

(a) $\theta_{j'}$ is a mapping;

(b) for all the terms of Q'' that are not copy variables, $\theta_{j'}$ maps all these terms of Q'' into the values in $adom(\mathcal{T}) - E$;

(c) all ground atoms mentioned in $\mathcal{A}_{j'}^{(i)}$ belong to $D_{\bar{N}^{(i)}}(Q)$ (by construction of the database);

(d) $\theta_{j'}$ is a candidate assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_{j'}^{(i)}$; and

(e) $\mathcal{A}_j^{(i)}$ and $\mathcal{A}_{j'}^{(i)}$ have the same atom-signature.

We now show that the association $\mathcal{A}_{j'}^{(i)}$ belongs to the set $\mathbb{A}_{Q''}^{(i)}$. This is immediate from the fact that θ_j and $\theta_{j'}$ agree on the images for all the head variables of Q'' and from items (a) and (c) of the previous paragraph. Finally, let $\mathcal{T}'$ be the set of all ground atoms mentioned in the association $\mathcal{A}_{j'}^{(i)}$. From item (b) of the previous paragraph and from Proposition L.9, we have that there exists in $D_{\bar{N}^{(i)}}(Q)$ a (single) extended canonical database for query Q, call that database D^*, such that $\mathcal{T}' \subseteq D^*$. We conclude that the unity valid assignment mapping $\theta_{j'}$ for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_{j'}^{(i)}$ maps query Q'' into a single extended canonical database (in $D_{\bar{N}^{(i)}}(Q)$) for the query Q. □

PROPOSITION L.21. *Suppose there exists an $i \in \mathbb{N}_+$ such that the set $\mathbb{A}_{Q''}^{(i)}$ for database $D_{\bar{N}^{(i)}}(Q)$ is not empty. Then the set $\mathbb{A}_{Q''}^{(i)}$ is not empty for* each $i \in \mathbb{N}_+$. □

PROOF. The proof is immediate from Propositions L.19 and L.20. That is:

- Case 1: Suppose that, for some $i \in \mathbb{N}_+$, the set $\mathbb{A}_{Q''}^{(i)}$ for database $D_{\bar{N}^{(i)}}(Q)$ has an association corresponding to a valid assignment mapping from query Q'' to a single extended canonical database for Q in $D_{\bar{N}^{(i)}}(Q)$. Then, by Proposition L.19, for *all* $i \in \mathbb{N}_+$, the set $\mathbb{A}_{Q''}^{(i)}$ for database $D_{\bar{N}^{(i)}}(Q)$ has an association corresponding to a valid assignment mapping from query Q'' to a single extended canonical database for Q in $D_{\bar{N}^{(i)}}(Q)$. Q.E.D.
- Case 2: Suppose that, for some $i \in \mathbb{N}_+$, the set $\mathbb{A}_{Q''}^{(i)}$ for database $D_{\bar{N}^{(i)}}(Q)$ has an association corresponding to a valid assignment mapping from query Q'' to (ground atoms in) two or more extended canonical databases for Q in $D_{\bar{N}^{(i)}}(Q)$. Then, by Proposition L.20, the set $\mathbb{A}_{Q''}^{(i)}$ for *the same value of* i has an association corresponding to a valid assignment mapping from query Q'' to a single extended canonical database for Q in $D_{\bar{N}^{(i)}}(Q)$. Thus we have reduced this case to Case 1. Q.E.D.

□

PROPOSITION L.22. *Suppose there exists an $i^* \in \mathbb{N}_+$ such that the query Q'' has no answer t_Q^* on database $D_{\bar{N}^{(i^*)}}(Q)$. Then the multiplicity of the tuple t_Q^* in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$ equals zero on the database $D_{\bar{N}^{(i)}}(Q)$ for each $i \in \mathbb{N}_+$.* □

PROOF. It is immediate from Proposition L.21 that if there exists an $i^* \in \mathbb{N}_+$ such that the set $\mathbb{A}_{Q''}^{(i^*)}$ for database $D_{\bar{N}^{(i^*)}}(Q)$ is empty, then the set $\mathbb{A}_{Q''}^{(i)}$ is empty for *each* $i \in \mathbb{N}_+$. □

L.6 Monomials for the Multiplicity of Tuple t_Q^* in Bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$

In this section we provide an algorithm for constructing monomials for a function, call it $\mathcal{F}_{(Q)}^{(Q'')}$, defined in terms of the variables in the vector $\bar{N}$. $\mathcal{F}_{(Q)}^{(Q'')}$ computes the multiplicity of the tuple t_Q^* in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$ for each $i \in \mathbb{N}_+$, by using the values in the vector $\bar{N}^{(i)}$ as values of the variables in the vector $\bar{N}$.

We observe first that, by Proposition L.22, $\mathcal{F}_{(Q)}^{(Q'')}$ either equals zero for all input vectors $\bar{N}^{(i)}$, or returns a positive-integer value for each $\bar{N}^{(i)}$, $i \in \mathbb{N}_+$. In the remainder of the proof of Theorem 4.1, we assume that the function $\mathcal{F}_{(Q)}^{(Q'')}$ returns a positive-integer value for each $\bar{N}^{(i)}$. By the results of Section L.5, we infer from this assumption that the cardinality $R_{Q''}^{(i)}$ of the set $\mathbb{A}_{Q''}^{(i)}$ is a positive integer for each $i \in \mathbb{N}_+$.

L.6.1 Defining the Monomial Classes $\mathcal{C}^{(Q'')}$

Fix an $i \in \mathbb{N}_+$. We partition all the elements of the set $\mathbb{A}_{Q''}^{(i)} \neq \emptyset$ into equivalence classes: Two distinct elements (in case $R_{Q''}^{(i)} \geq 2$) $\mathcal{A}_j^{(i)}$ and $\mathcal{A}_k^{(i)}$ of the set $\mathbb{A}_{Q''}^{(i)}$ belong to the same *monomial class* if and only if $\mathcal{A}_j^{(i)}$ and $\mathcal{A}_k^{(i)}$ have the same atom-signature. Call all the resulting nonempty monomial classes $\mathcal{C}_1^{(Q'')(i)}$, $\mathcal{C}_2^{(Q'')(i)}$, $\ldots$, $\mathcal{C}_{n^{(i)}}^{(Q'')(i)}$, $n^{(i)} \leq R_{Q''}^{(i)}$. From the definition of the monomial classes, we have that $n^{(i)} \geq 1$, and that $n^{(i)}$ is exactly the number of all the atom-signatures of the elements of the set $\mathbb{A}_{Q''}^{(i)}$. In addition, by Proposition L.12 and from the definition of the set $\mathbb{A}_{Q''}^{(i)}$ we have that for each j, $1 \leq j \leq n^{(i)}$, all the elements of the set $\mathcal{C}_j^{(Q'')(i)}$ (by having the same atom-signature) have the same noncopy-signature and have the same copy-signature. Hence, for each monomial class $\mathcal{C}_j^{(Q'')(i)}$ we can refer to *the* atom-signature of $\mathcal{C}_j^{(Q'')(i)}$, to *the* noncopy-signature of $\mathcal{C}_j^{(Q'')(i)}$, and to *the* copy-signature of $\mathcal{C}_j^{(Q'')(i)}$.

By Proposition L.18, we have that for each $i \in \mathbb{N}_+$ and for each monomial class for Q'' and $D_{\bar{N}^{(i)}}(Q)$, all the valid assignment mappings of all the elements of the class contribute tuples to the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$. That is, for all the valid assignment mappings in each monomial class, the restriction of each valid assignment mapping to the head vector of the query Q'' is the tuple t_Q^*.

This result follows from the results of Sections L.4.2 and L.5:

Proposition L.23. *Let Ξ be a G-ary vector of (not necessarily distinct) elements of the set $\mathcal{S}_{C(Q)}$. Suppose there exists an $i^* \in \mathbb{N}_+$ such that the monomial class $\mathcal{C}^{(Q'')(i^*)}$ with atom-signature Ξ is not empty. Then for* all $i \in \mathbb{N}_+$ *it holds that the monomial class $\mathcal{C}^{(Q'')(i)}$ with atom-signature Ξ is not empty.* □

From Proposition L.23 it follows that for a fixed query Q'', we can drop the (i)-superscript from the notation for monomial classes. (That is, the set of nonempty monomial classes for Q'', w.r.t. the family $\{D_{\bar{N}^{(i)}}(Q)\}$, does not depend on the specific database $D_{\bar{N}^{(i)}}(Q)$ in the family.) From now on, when referring to the set of all nonempty monomial classes for query Q'' on database $D_{\bar{N}^{(i)}}(Q)$, we will use the notation $\mathcal{C}_1^{(Q'')}$, $\mathcal{C}_2^{(Q'')}$, $\ldots$, $\mathcal{C}_{n^*}^{(Q'')}$, for a constant (w.r.t. i) positive-integer value $n^* \geq 1$. We will abuse the notation somewhat, by using, in the context of a fixed $i \in \mathbb{N}_+$, the expression "the set $\mathcal{C}^{(Q'')}$" (where $\mathcal{C}^{(Q'')}$ is one of the $\mathcal{C}_1^{(Q'')}$, $\mathcal{C}_2^{(Q'')}$, $\ldots$, $\mathcal{C}_{n^*}^{(Q'')}$) to refer to the contents of the set $\mathcal{C}^{(Q'')}$ w.r.t. the set $\mathbb{A}_{Q''}^{(i)}$ for the fixed i.

L.6.2 Monomials Corresponding to the Monomial Classes for Q'' and $D_{\bar{N}^{(i)}}(Q)$: Useful Properties

In this subsection we set the stage for the introduction, in Section L.6.3, of "multiplicity monomials" for the monomial classes $\mathcal{C}_1^{(Q'')}$, $\ldots$, $\mathcal{C}_{n^*}^{(Q'')}$.

Assuming a fixed $i \in \mathbb{N}_+$, we recall the mapping ν_0 and the sets $S_j^{(i)}$, which (sets) were introduced (for $1 \leq j \leq m$) for the case $m \geq 1$, see Section L.3.1. We use these constructs to define the domain, on the database $D_{\bar{N}^{(i)}}(Q)$, of each term of the query Q that (term) is not a copy variable of Q.

Definition L.9. *(**Domain of term of Q in $D_{\bar{N}^{(i)}}(Q)$**) Let $i \in \mathbb{N}_+$. For each term s of query Q such that s is not a copy variable of Q, the domain $Dom_Q^{(i)}(s)$ of the term in the database $D_{\bar{N}^{(i)}}(Q)$ is defined as follows:*

- *If s is a constant, or a head variable of Q, or a set variable of Q, then $Dom_Q^{(i)}(s) := \{\nu_0(s)\}$.*
- *In case $m \geq 1$, for each variable Y_j of the query Q, for $1 \leq j \leq m$, $Dom_Q^{(i)}(Y_j) := S_j^{(i)}$.*

□

Proposition L.24. *Let $i \in \mathbb{N}_+$. (i) For each (if any) pair (s, t) of terms of query Q such that $s \neq t$ and such that neither s nor t is a copy variable of Q, $Dom_Q^{(i)}(s) \bigcap Dom_Q^{(i)}(t) = \emptyset$. (ii) For each term s of query Q such that s is not a multiset variable of Q, $|Dom_Q^{(i)}(s)| = 1$. (iii) In case $m \geq 1$, for each $j \in \{1, \ldots, m\}$ we have that $|Dom_Q^{(i)}(Y_j)| = N_j^{(i)}$ (in the vector $\bar{N}^{(i)}$).* □

For the next results, we introduce some notation. Given a query Q'', an $i \in \mathbb{N}_+$, and a nonempty monomial class $\mathcal{C}^{(Q'')}$ of associations in the set $\mathbb{A}_{Q''}^{(i)}$ for the query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, denote by $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ the set of all tuples contributed to the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$ by all the valid assignment mappings for all the elements of the class $\mathcal{C}^{(Q'')}$. The following result is immediate from the definitions.

Proposition L.25. *Let $i \in \mathbb{N}_+$. Then*

(i) For each $j \in \{1, \ldots, n^*\}$, $\Gamma^{(i)}[\mathcal{C}_j^{(Q'')}] \neq \emptyset$.

(ii) The set $\Gamma_{\bar{S}}^{(t_Q^*)}$ $(Q'', D_{\bar{N}^{(i)}}(Q))$ is the union $\bigcup_{j=1}^{n^*} \Gamma^{(i)}[\mathcal{C}_j^{(Q'')}]$.

□

We introduce some further notation: In case $m \geq 1$, for a monomial class $\mathcal{C}^{(Q'')}$ and for some $j \in \{1, \ldots, m\}$, we denote by $\Gamma_{(Y_j'')}^{(i)}[\mathcal{C}^{(Q'')}]$ the set projection of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on the multiset noncopy variable Y_j'' of the query Q''.

Proposition L.26. *Suppose $m \geq 1$. Let Ξ be a G-ary vector of (not necessarily distinct) elements of the set $\mathcal{S}_{C(Q)}$, such that the monomial class $\mathcal{C}^{(Q'')}$ with atom-signature Ξ is not empty. Then for each $i \in \mathbb{N}_+$ the following holds:*

(i) For each $j \in \{1, \ldots, m\}$: Suppose Z is the jth component of the noncopy-signature vector of the monomial class $\mathcal{C}^{(Q'')}$. Then the set $\Gamma_{(Y_j'')}^{(i)}[\mathcal{C}^{(Q'')}]$:

(i-a) has all the elements of $Dom_Q^{(i)}(Z)$, and

(i-b) has no values from the set $(adom(D_{\bar{N}^{(i)}}(Q)) - Dom_Q^{(i)}(Z))$.

(ii) The set projection of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the multiset noncopy variables Y_1'', Y_2'', $\ldots$, Y_m'' of the query Q'' is the Cartesian product of the sets $\Gamma_{(Y_1'')}^{(i)}$ $[\mathcal{C}^{(Q'')}]$, $\Gamma_{(Y_2'')}^{(i)}$ $[\mathcal{C}^{(Q'')}]$, $\ldots$, $\Gamma_{(Y_m'')}^{(i)}$ $[\mathcal{C}^{(Q'')}]$.

□

Now suppose $r \geq 1$. In this case, we denote by $\Gamma_{(Y_j'')}^{(i)}[\mathcal{C}^{(Q'')}]$ the set projection of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on the copy variable Y_j'' of the query Q'', for some $j \in \{m+1, \ldots, m+r\}$.

Proposition L.27. *Suppose $r \geq 1$. Let Ξ be a G-ary vector of (not necessarily distinct) elements of the set $\mathcal{S}_{C(Q)}$, such that the monomial class $\mathcal{C}^{(Q'')}$ with atom-signature Ξ is not empty. Then for each $i \in \mathbb{N}_+$ the following holds:*

(i) For each $j \in \{1, \ldots, r\}$: Suppose Z is the jth component of the copy-signature vector of the monomial class $\mathcal{C}^{(Q'')}$. Then the set $\Gamma^{(i)}_{(Y''_{m+j})}[\mathcal{C}^{(Q'')}]$ is the set $\{1, \ldots, Z^{(i)}\}$, where (a) $Z^{(i)}$ is 1 in case $Z = 1$, and (b) $Z^{(i)}$ is $N^{(i)}_k$ in case $Z = N_k$ for some $k \in \{m+1, \ldots, m+w\}$.

(ii) The set projection of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the copy variables $Y''_{m+1}, Y''_{m+2}, \ldots, Y''_{m+r}$ of the query Q'' is the Cartesian product of the sets $\Gamma^{(i)}_{(Y''_{m+1})}[\mathcal{C}^{(Q'')}]$, $\Gamma^{(i)}_{(Y''_{m+2})}[\mathcal{C}^{(Q'')}]$, $\ldots$, $\Gamma^{(i)}_{(Y''_{m+r})}[\mathcal{C}^{(Q'')}]$.

$\square$

(The proof is immediate from Proposition L.17, once we recall that all associations in a monomial class share the same copy-signature.)

For each $i \in \mathbb{N}_+$, we now characterize the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ for each nonempty monomial class $\mathcal{C}^{(Q'')}$ for the query Q'' and family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, for all combinations of values of $m \geq 0$ and of $r \geq 0$.

Proposition L.28. *Let Ξ be a G-ary vector of (not necessarily distinct) elements of the set $\mathcal{S}_{C(Q)}$, such that the monomial class $\mathcal{C}^{(Q'')}$ with atom-signature Ξ is not empty. Then for each $i \in \mathbb{N}_+$ the following holds:*

- *In case $m \geq 1$ and $r \geq 1$, the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ is the Cartesian product of two sets:*
  - *the set projection of $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the multiset noncopy variables $Y''_1, \ldots, Y''_m$ of the query Q'', and*
  - *the set projection of $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the copy variables $Y''_{m+1}, \ldots, Y''_{m+r}$ of the query Q''.*
- *In case $r = 0$, $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ is its own set projection on all the multiset noncopy variables of the query Q''.*
- *In case $m = 0$, $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ is its own set projection on all the copy variables of the query Q''.*

$\square$

(Recall from Section L.2.1 that we assume $m + r \geq 1$; thus in case $r = 0$ we have $m \geq 1$, and in case $m = 0$ we have $r \geq 1$. For a characterization of the set projection of $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the multiset noncopy variables $Y''_1, \ldots, Y''_m$ of the query Q'', in case $m \geq 1$, see Proposition L.26. For a characterization of the set projection of $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the copy variables $Y''_{m+1}, \ldots, Y''_{m+r}$ of the query Q'', in case $r \geq 1$, see Proposition L.27.)

L.6.3 Multiplicity Monomials for the Monomial Classes $\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}$

In this subsection, for each nonempty monomial class $\mathcal{C}^{(Q'')}$ for the query Q'' we construct an expression, such that for each $i \in \mathbb{N}_+$, this expression will return the number of distinct tuples contributed by the associations in $\mathcal{C}^{(Q'')}$ to the set $\Gamma^{(t^*_Q)}_{\bar{S}}(Q'', D_{\bar{N}^{(i)}}(Q))$. That is, we construct an expression that, for each $i \in \mathbb{N}_+$, will provide the cardinality of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$. (See Section L.6.2 for the notation $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$.) For each monomial class $\mathcal{C}^{(Q'')} \in \{\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}\}$, we call the respective expression "the multiplicity monomial of the monomial class $\mathcal{C}^{(Q'')}$." Each multiplicity monomial is a product of (some powers of) the elements of the noncopy singature and of the copy signature of the corresponding monomial class. These multiplicity monomials, together with the copy-signatures and the noncopy-signatures of the monomial classes, are all that will be needed in Section L.9 to construct the function $\mathcal{F}^{(Q'')}_{(Q)}$.

We begin by introducing the necessary notation. For each term s of the query Q such that s is not a copy variable of Q, by $DomLabel_Q(s)$ we denote (i) variable N_j in case where $m \geq 1$ and where s is the variable Y_j of Q for some $1 \leq j \leq m$; and (ii) constant value 1 in case s is either a constant used in Q or is one of the variables $X_1, \ldots, X_{l+u}$ of Q.

Further, for Propositions L.29 and L.30, we use the following notation, for ease of reference to the elements of the noncopy signatures and of the copy signatures of the monomial classes. Let Ξ be a G-ary vector of (not necessarily distinct) elements of the set $\mathcal{S}_{C(Q)}$, such that the monomial class $\mathcal{C}^{(Q'')}$ with atom-signature Ξ is not empty.

(1) Let $\Phi^{\mathcal{C}^{(Q'')}}_n$ be the noncopy-signature of the class $\mathcal{C}^{(Q'')}$. Then:

- In case $m \geq 1$, denote the elements of $\Phi^{\mathcal{C}^{(Q'')}}_n$, from left to right, as $Z_1, Z_2, \ldots, Z_m$. For all $j \in \{1, \ldots, m\}$, we have that $Z_j \in \{Y_1, \ldots, Y_m, X_1, \ldots, X_{l+u}\} \bigcup P$.
- For an $i \in \mathbb{N}_+$, denote by $\Pi^{(i)}_{\Phi^{\mathcal{C}^{(Q'')}}_n}$ the value 1 in case $m = 0$, and the product $\Pi^m_{j=1}|Dom^{(i)}_Q(Z_j)|$ in case $m \geq 1$.
- Finally, denote by $\Pi_{\Phi^{\mathcal{C}^{(Q'')}}_n}$ the value 1 in case $m = 0$, and the product $\Pi^m_{j=1} DomLabel_Q(Z_j)$ in case $m \geq 1$.

(2) Let $\Phi^{\mathcal{C}^{(Q'')}}_c$ be the copy-signature of the class $\mathcal{C}^{(Q'')}$. Then:

- In case $r \geq 1$, denote the elements of $\Phi^{\mathcal{C}^{(Q'')}}_c$, from left to right, as $W_1, W_2, \ldots, W_r$. For all $j \in \{1, \ldots, r\}$, we have that $W_j \in \{1, N_{m+1}, \ldots, N_{m+w}\}$.
- For an $i \in \mathbb{N}_+$ and for each $j \in \{1, \ldots, r\}$ (still assuming $r \geq 1$), denote by $W^{(i)}_j$ the value of the variable W_j in the vector $\bar{N}^{(i)}$, in case $W_j \neq 1$. (That is, whenever $W_j = N_{m+l}$, for some $l \in \{1, \ldots, w\}$, then $W^{(i)}_j = N^{(i)}_{m+l}$.) If $W_j = 1$ then let $W^{(i)}_j := 1$.
- For an $i \in \mathbb{N}_+$, denote by $\Pi^{(i)}_{\Phi^{\mathcal{C}^{(Q'')}}_c}$ the value 1 in case $r = 0$, and the product $\Pi^r_{j=1} W^{(i)}_j$ in case $r \geq 1$.
- Finally, denote by $\Pi_{\Phi^{\mathcal{C}^{(Q'')}}_c}$ the value 1 in case $r = 0$, and the product $\Pi^r_{j=1} W_j$ in case $r \geq 1$.

Proposition L.29. *Let Ξ be a G-ary vector of (not necessarily distinct) elements of the set $\mathcal{S}_{C(Q)}$, such that the monomial class $\mathcal{C}^{(Q'')}$ with atom-signature Ξ is not empty. Let $i \in \mathbb{N}_+$. Then the cardinality of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ (that is, the number of distinct tuples contributed, to the set $\Gamma^{(t^*_Q)}_{\bar{S}}(Q'', D_{\bar{N}^{(i)}}(Q))$ for the query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, by all the valid assignment mappings for all the elements of the class $\mathcal{C}^{(Q'')}$) is exactly $\Pi^{(i)}_{\Phi^{\mathcal{C}^{(Q'')}}_n} \times \Pi^{(i)}_{\Phi^{\mathcal{C}^{(Q'')}}_c}$.*
$\square$

Please see Example L.3 for an illustration. he proof of Proposition L.29 is immediate from Proposition L.28. (See Proposition L.17 for the details on the $\Pi^{(i)}_{\Phi_c^{\mathcal{C}(Q'')}}$ part of the computation. The $\Pi^{(i)}_{\Phi_n^{\mathcal{C}(Q'')}}$ part of the computation follows from the construction of the database $D_{\bar{N}^{(i)}}(Q)$, specifically from the definition of the main construction cycle as described in Section L.3.2.)

We note that the expression $\Pi^{(i)}_{\Phi_n^{\mathcal{C}(Q'')}} \times \Pi^{(i)}_{\Phi_c^{\mathcal{C}(Q'')}}$ in Proposition L.29 is in terms of only the elements of the vector $\bar{N}^{(i)}$, and is uniform across all $i \in \mathbb{N}_+$. Thus, we obtain the following result as an easy corollary of Proposition L.29.

Proposition L.30. *Let Ξ be a G-ary vector of (not necessarily distinct) elements of the set $\mathcal{S}_{C(Q)}$, such that the monomial class $\mathcal{C}^{(Q'')}$ with atom-signature Ξ is not empty. Then, for all $i \in \mathbb{N}_+$, the cardinality of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ (that is, the number of distinct tuples contributed, to the set $\Gamma^{(t^*_Q)}_{\bar{S}}(Q'', D_{\bar{N}^{(i)}}(Q))$ for the query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$, by all the valid assignment mappings for all the elements of the class $\mathcal{C}^{(Q'')}$) can be computed by substituting the values in the vector $\bar{N}^{(i)}$ (specifically value $N_j^{(i)}$ as value of variable N_j, for each $j \in \{1, \ldots, m+w\}$) into the formula $\Pi_{\Phi_n^{\mathcal{C}(Q'')}} \times \Pi_{\Phi_c^{\mathcal{C}(Q'')}}$.* □

For a monomial class $\mathcal{C}^{(Q'')}$ with noncopy signature $\Phi_n^{\mathcal{C}(Q'')}$ and with copy signature $\Phi_c^{\mathcal{C}(Q'')}$, such that $\mathcal{C}^{(Q'')}$ is not empty, we call the expression (of Proposition L.30) $\Pi_{\Phi_n^{\mathcal{C}(Q'')}} \times \Pi_{\Phi_c^{\mathcal{C}(Q'')}}$, in terms of the variables in the vector $\bar{N}$, *the multiplicity monomial of the monomial class $\mathcal{C}^{(Q'')}$.*

L.7 The Wave Monomial of the Query Q

In this section we obtain results that are instrumental in proving Theorem 4.1. Namely, we show that:

(1) There exists a (nonempty) monomial class $\mathcal{C}^{(Q)}$ *for the query* Q, such that the multiplicity monomial of $\mathcal{C}^{(Q)}$ is "the wave of the query Q." (See Proposition L.33.) The wave of the query Q is defined in this section (see Definition L.10) based on the vector $\bar{N}$ and on the mapping ν_Q^{copy} defined in Section L.3.2.

(2) Suppose that for a CCQ query Q'', there exists a (nonempty) monomial class $\mathcal{C}^{(Q'')}$, such that the multiplicity monomial of $\mathcal{C}^{(Q'')}$ is "the wave of the query Q." Then there exists a SCVM from the query Q'' to the query Q. (See Proposition L.34.)

In Section L.10 we will see that whenever (a) $Q' \equiv_C Q$ for a CCQ query Q' and (b) Q is an explicit-wave CCQ query, then there exists a (nonempty) monomial class $\mathcal{C}_*^{(Q')}$ for the query Q', such that the multiplicity monomial of $\mathcal{C}_*^{(Q')}$ is "the wave of the query Q." The proof of Theorem 4.1 is immediate from that result and from Propositions L.33 and L.34 of this section. (We remind the reader that throughout the proof of Theorem 4.1, all monomial classes of all queries are defined w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ for the fixed input query Q.)

We begin the exposition by defining "the wave of the query Q." We introduce some notation to make the definition concise:

(A) Denote by $\mathcal{P}^{(Q)}_{noncopy}$ (i) the constant 1 in case $m = 0$, and (ii) the product $\Pi_{j=1}^m N_j$ in case $m \geq 1$.

(B) Denote by $\mathcal{P}^{(Q)}_{copy}$ (i) the constant 1 in case $r = 0$, and (ii) the product $\Pi_{j=1}^r \nu_Q^{copy}(Y_{m+j})$ in case $r \geq 1$. (For the notation ν_Q^{copy}, see Section L.3.2.)

Definition L.10. *(***The wave of CCQ query** *Q) For a CCQ query Q,* the wave *$\mathcal{P}_*^{(Q)}$ of Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ is the product $\mathcal{P}_*^{(Q)} = \mathcal{P}^{(Q)}_{noncopy} \times \mathcal{P}^{(Q)}_{copy}$.* □

The intuition for the wave $\mathcal{P}_*^{(Q)}$ of a CCQ query Q is that $\mathcal{P}_*^{(Q)}$ reflects (i) the association of each multiset non-copy variable of Q (in case $m \geq 1$) with a separate variable among $N_1, \ldots, N_m$, and (ii) the association, in case $r \geq 1$, of each copy-sensitive subgoal, call it s, of Q (via the copy variable of the subgoal) with the unique element, call it s', of the set $\mathcal{S}_{C(Q)}$ (see Section L.2.1) such that the subgoal s and the element s' have the same relational template. The provenance of each association will become explicit in the proof of Proposition L.33. As an illustration of the definition, in Example L.5, he wave of query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ is the product $\mathcal{P}_*^{(Q)} = \Pi_{j=1}^4 N_j$. In the same example, the wave of query Q' w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q')\}$ is the product $\mathcal{P}_*^{(Q')} = N_1 \times N_2 \times (N_3)^2$. The component $(N_3)^2$ of the expression $\mathcal{P}_*^{(Q')}$ comes from the fact that the set $\mathcal{S}_{C(Q')}$, for the query Q' of Example L.5, s a singleton set, and both subgoals of the query Q' agree on the relational template with the only element of the set $\mathcal{S}_{C(Q')}$.

Proposition L.31. *Given a CCQ query Q and the vector $\bar{N} = [\ N_1\ N_2\ \ldots\ N_{m+w}]$ that is used to construct the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ for the query Q. Then each element of the vector $\bar{N}$ occurs in the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$.* □

The proof of Proposition L.31 is immediate from the definition of the products $\mathcal{P}^{(Q)}_{noncopy}$ and $\mathcal{P}^{(Q)}_{copy}$ used in Definition L.10. (In case where $r \geq 1$, the less obvious part of the proof, that is the presence of each of $N_{m+1}, N_{m+2}, \ldots, N_{m+w}$ in the product $\mathcal{P}^{(Q)}_{copy}$, is immediate from the definition of the mapping ν_Q^{copy}, see Section L.3.2.)

Our next result is immediate from the definition of the wave of the query Q. (For each expression of the form N_j^k, such that $N_j \in \{N_1, N_2, \ldots, N_{m+r}\}$ and $k \geq 1$, we say that the expression N_j^k has k occurrences of the variable N_j.)

Proposition L.32. *Given a CCQ query Q and the vector $\bar{N} = [\ N_1\ N_2\ \ldots\ N_{m+w}]$ that is used to construct the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ for the query Q. Then the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$ has exactly $m + r$ occurrences of the variables from the set $\{N_1, N_2, \ldots, N_{m+r}\}$.* □

Proposition L.33. *Given a CCQ query Q, there exists a nonempty monomial class, call it $\mathcal{C}_*^{(Q)}$, for the query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of $\mathcal{C}_*^{(Q)}$ is the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$.* □

Proof. The proof has three parts:

(1) We first show that for each $i \in \mathbb{N}_+$, there exists an association for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$, call this association $\mathcal{A}_*^{(i)}$, such that:

  (i) The association $\mathcal{A}_*^{(i)}$ has a t_Q^*-valid assignment mapping for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$;

  (ii) In case $m \geq 1$, we have that the noncopy-signature[16] $\Phi_n[\mathcal{A}_*^{(i)}]$ of the association $\mathcal{A}_*^{(i)}$ is the vector $[Y_1 \ \ldots \ Y_m]$; and, finally,

  (iii) In case $r \geq 1$, we have that the copy-signature $\Phi_c[\mathcal{A}_*^{(i)}]$ of the association $\mathcal{A}_*^{(i)}$ is the vector $[\nu_Q^{copy}(Y_{m+1}) \ \ldots \ \nu_Q^{copy}(Y_{m+r})]$.

(2) We then show that the query Q, w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, has a nonempty monomial class whose noncopy-signature (whose copy-signature, respectively) is the noncopy-signature (the copy-signature, respectively) of the associations $\mathcal{A}_*^{(i)}$, for all $i \in \mathbb{N}_+$, of item (1) of this proof. We denote this monomial class by $\mathcal{C}_*^{(Q)}$.

(3) Finally, we show that the multiplicity monomial of the monomial class $\mathcal{C}_*^{(Q)}$ of item (2) is the wave of the query Q.

In fact, items (2) and (3) are straightforward: Item (2) is immediate from the definition of monomial classes and from item (1), and item (3) is immediate from item (2) and from Proposition L.30. Hence, in the remainder of this proof we prove parts (i) through (iii) of item (1) above.

Recall that we assume $m + r \geq 1$. Hence the set $\mathcal{S}_{C(Q)}$ (see Section L.2.1) for the query Q is not empty.

Fix an $i \in \mathbb{N}_+$. Recall the set $\mathcal{S}^{(i)} \neq \emptyset$ introduced in Section L.3.2 to construct the database $D_{\bar{N}^{(i)}}(Q)$. Fix an arbitrary tuple $t \in \mathcal{S}^{(i)}$. For the tuple $t \in \mathcal{S}^{(i)}$, Section L.3.2 defined a mapping, ν_t, from all the terms of the query Q to constants in the set $adom(D_{\bar{N}^{(i)}}(Q)) \bigcup \mathbb{N}_+$. By definition, for the $t \in \mathcal{S}^{(i)}$ we have that the restriction of ν_t to all the terms of the query Q occurring in the elements of the set $\mathcal{S}_{C(Q)}$ induces a bijection from the subset $\mathcal{S}_{C(Q)}$ of the condition of Q to a set, call it D_t, of ground atoms of the database $D_{\bar{N}^{(i)}}(Q)$. By construction of the database $D_{\bar{N}^{(i)}}(Q)$, the set D_t (i) was generated from $\mathcal{S}_{C(Q)}$ using the mapping ν_t, and (ii) is an extended canonical database for the query Q.

We now construct the association $\mathcal{A}_*^{(i)}$. We begin by associating each atom $s \in \mathcal{S}_{C(Q)}$ with its image (in the set of ground atoms $D_t \subseteq D_{\bar{N}^{(i)}}(Q)$) under ν_t. Now there are two cases: (a) In case all the subgoals of the (regularized version of the) query Q are elements of the set $\mathcal{S}_{C(Q)}$, we are done with the construction of the association $\mathcal{A}_*^{(i)}$. We now consider the remaining case (b), where there exist subgoals of the (regularized version of the) query Q that are not elements of the set $\mathcal{S}_{C(Q)}$. Consider an arbitrary subgoal s of Q such that $s \in (L - \mathcal{S}_{C(Q)})$, where L is the condition of the regularized version of the query Q. By definition of the set $\mathcal{S}_{C(Q)}$, s is a copy-sensitive atom, such that there exists a unique element, call it s', of the set $\mathcal{S}_{C(Q)}$, such that s and s' have the same relational template.

[16]Observe that the noncopy-signature of the association $\mathcal{A}_*^{(i)}$ is well defined, by $\mathcal{A}_*^{(i)}$ having a valid assignment mapping for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$.

Then in our construction of the association $\mathcal{A}_*^{(i)}$, for each such subgoal s of the query Q, $s \in (L - \mathcal{S}_{C(Q)})$, we associate (in $\mathcal{A}_*^{(i)}$) the atom s with the atom $\nu_t(s') \in D_{\bar{N}^{(i)}}(Q)$, for the s' as determined in the previous paragraph. This completes the construction of the association $\mathcal{A}_*^{(i)}$. Observe that by construction, in both cases (a) and (b) as in the preceding paragraphs, the association $\mathcal{A}_*^{(i)}$ associates all the elements of the condition of the query Q with exactly the set of ground atoms $D_t \subseteq D_{\bar{N}^{(i)}}(Q)$.

We now prove claim (1)(i) of the beginning of this proof. We first show that the association $\mathcal{A}_*^{(i)}$ has a valid assignment mapping for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$. Indeed, by definition of ν_t it holds that ν_t assigns values to *all* terms of the query Q, consistently across all the pairs in the association $\mathcal{A}_*^{(i)}$. Denote by $\theta_*^{(Q)}$ the resulting valid assignment mapping for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$. Now, it is immediate from the definition of ν_t that the restriction of the mapping $\theta_*^{(Q)}$ to the head vector $[X_1 \ \ldots \ X_l]$ of the query Q is the tuple t_Q^*. Thus, $\theta_*^{(Q)}$ is a t_Q^*-valid assignment mapping for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$, which completes our proof of claim (1)(i).

We now prove claim (1)(ii) of the beginning of this proof. This claim requires the assumption that $m \geq 1$. Under this assumption, by definition of ν_t we have that ν_t maps the variable Y_j, for each $j \in \{1, \ldots, m\}$, into an element of the set $S_j^{(i)}$. (See Section L.3.1 for the definition of $S_j^{(i)}$.) Hence, by definition of the vector $\Phi_n[\mathcal{A}_*^{(i)}]$ (see Section L.4.1), the jth element of $\Phi_n[\mathcal{A}_*^{(i)}]$ is the variable Y_j, for each $j \in \{1, \ldots, m\}$. Q.E.D.

To complete the proof of Proposition L.33, it remains to prove claim (1)(iii) of the beginning of this proof. This claim requires the assumption that $r \geq 1$. Under this assumption, the claim is immediate from the construction of the association $\mathcal{A}_*^{(i)}$ and from the definition of the vector $\Phi_c[\mathcal{A}_*^{(i)}]$ (see Section L.4.1). Q.E.D. $\square$

PROPOSITION L.34. *Given CCQ queries $Q(\bar{X}) \leftarrow L, M$ and $Q''(\bar{X}'') \leftarrow L'', M''$, such that (i) Q and Q'' have the same (positive-integer) head arities, (ii) $|M_{copy}| = |M''_{copy}|$, and (iii) $|M_{noncopy}| = |M''_{noncopy}|$. Suppose that there exists a nonempty monomial class $\mathcal{C}^{(Q'')}$ for the query Q'' w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of $\mathcal{C}^{(Q'')}$ is the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$. Then there exists a SCVM from the query Q'' to the query Q.* $\square$

The proof of Proposition L.34 is constructive: That is, the proof generates a SCVM from the query Q'' to the query Q of the statement of Proposition L.34.

PROOF. We are given that there exists a nonempty monomial class $\mathcal{C}^{(Q'')}$ for the query Q'' w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of $\mathcal{C}^{(Q'')}$ is the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$. Then, by definition of multiplicity monomials (Section L.6.3) and of copy-/noncopy-sigature vectors (Section L.4.1), we have that:

- In case $m \geq 1$, the vector $\Phi_n[\mathcal{C}^{(Q'')}]$ must be a permutation of the vector $[Y_1 \ \ldots \ Y_m]$; and
- In case $r \geq 1$, the vector $\Phi_c[\mathcal{C}^{(Q'')}]$ must be a permutation of the vector $[\nu_Q^{copy}(Y_{m+1}) \ \ldots \ \nu_Q^{copy}(Y_{m+r})]$.

By definition of monomial classes and from the fact that the monomial class $\mathcal{C}^{(Q'')}$ for the query Q'' w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ is not empty (see Section L.5 for the relevant results), we have that for each $i \in \mathbb{N}_+$, the monomial class $\mathcal{C}^{(Q'')}$ has an association with at least one t^*_Q-valid assignment mapping for Q'' and $\{D_{\bar{N}^{(i)}}(Q)\}$. Fix an arbitrary $i \in \mathbb{N}_+$ and consider an arbitrary such association in $\mathcal{C}^{(Q'')}$. Denote the association by $\mathcal{A}^{(init)}_*$, and denote by $\theta^{(init)}_*$ its unity t^*_Q-valid assignment mapping for Q'' and $\{D_{\bar{N}^{(i)}}(Q)\}$; the mapping $\theta^{(init)}_*$ exists by Proposition L.14.

By Proposition L.20, in case the association $\mathcal{A}^{(init)}_*$ is such that the mapping $\theta^{(init)}_*$ induces a mapping from the subgoals of the query Q'' into two or more extended canonical databases (in $D_{\bar{N}^{(i)}}(Q)$) for the query Q, there must exist an association, call it $\mathcal{A}_*$, that has the same atom-signature as $\mathcal{A}^{(init)}_*$ and such that the unity valid assignment mapping for $\mathcal{A}_*$, call this mapping θ_*, induces a mapping from the subgoals of the query Q'' into a single extended canonical database (in $D_{\bar{N}^{(i)}}(Q)$) for the query Q, call this database D_*. Observe that from the fact that $\mathcal{A}^{(init)}_*$ and $\mathcal{A}_*$ have the same atom-signature, we have that $\mathcal{A}_*$ belongs to the monomial class $\mathcal{C}^{(Q'')}$, just as $\mathcal{A}^{(init)}_*$ does. In addition, $\mathcal{A}^{(init)}_*$ and $\mathcal{A}_*$ have the same copy-signature (which is $\Phi_c[\mathcal{C}^{(Q'')}]$), as well as the same noncopy-signature (which is $\Phi_n[\mathcal{C}^{(Q'')}]$).

If, on the other hand, the association $\mathcal{A}^{(init)}_*$ is such that the mapping $\theta^{(init)}_*$ induces a mapping from the subgoals of the query Q'' into a single extended canonical database (in $D_{\bar{N}^{(i)}}(Q)$) for the query Q, call this database D_*, then for the remainder of the proof we refer to $\mathcal{A}^{(init)}_*$ as $\mathcal{A}_*$, and refer to $\theta^{(init)}_*$ as θ_*.

Now denote by $\nu_{Q''}$ the mapping (i) whose domain in the set of all the terms of the query Q'' that are not copy variables of Q'', and (ii) such that on the entire domain of $\nu_{Q''}$, the mapping $\nu_{Q''}$ coincides with the mapping θ_*. Further, define $\mu'_{Q''}$ as the composition $\nu^{(i)}_Q \circ \nu_{Q''}$ of the mapping $\nu_{Q''}$ with the mapping $\nu^{(i)}_Q$ defined in Section L.3.1. By definition, $\mu'_{Q''}$ is a mapping from all the terms of the query Q'' that are not copy variables of Q'' to terms of the query Q. Finally, define $\mu_{Q''}$ as the mapping (i) whose domain is the set of all terms of the query Q'' (that is, including all the copy variables of Q''), and (ii) such that on the entire domain of $\mu'_{Q''}$, the mapping $\mu_{Q''}$ coincides with the mapping $\mu'_{Q''}$.

Finally, (iii) in case $r \geq 1$, for each $j \in \{1, \ldots, r\}$, we define $\mu_{Q''}(Y''_{m+j})$ as follows: Suppose the jth element of the vector $\Phi_c[\mathcal{C}^{(Q'')}]$, being the variable (in the vector $\bar{N}$) N_{m+k} for some $1 \leq k \leq w$,[17] occurs in the vector $\Phi_c[\mathcal{C}^{(Q'')}]$ a total of n times, where $1 \leq n \leq r$. Suppose further that out of these n positions in which variable N_{m+k} occurs in the vector $\Phi_c[\mathcal{C}^{(Q'')}]$, our fixed position j is the lth such position from the left, $1 \leq l \leq n$. Then, by definition of the wave of the query Q, it must be that for the equivalence class, call it $\mathcal{C}(Y_{m+k})$, for the *same* value k as above (i.e., the k in Y_{m+k} is the same as in the N_{m+k}), of the copy-sensitive subgoal s of Q where the copy variable of s is Y_{m+k}, the class $\mathcal{C}(Y_{m+k})$ has exactly n copy-sensitive subgoals of the query Q. All these n subgoals of the query Q have the same relational template, but have distinct copy variables, call the assortment of these copy variables Y_{i1}, Y_{i2}, …, Y_{in}, with $i1 < i2 < \ldots < in$. (Naturally, the variable Y_{m+k} is one of these n copy variables Y_{i1}, Y_{i2}, …, Y_{in}.) Then define $\mu_{Q''}(Y''_{m+j})$ to be the copy variable Y_{il} of the query Q, where Y_{il} is the lth variable in the list ($Y_{i1}, Y_{i2}, \ldots, Y_{in}$). Note that this assignment algorithm terminates and results in the same assignments, in $\mu_{Q''}$, for all copy variables of the query Q'' independently of the order in which we choose the positions j out of the set $\{1, \ldots, r\}$. Observe also that $\mu_{Q''}$ is still a mapping once we are done with all the assignments in (iii). (Indeed, each copy variable of the query Q'', in case $r \geq 1$, is assigned by $\mu_{Q''}$ to a distinct copy variable of the query Q.) Finally, observe that in case $r = w \geq 1$, the mapping $\mu_{Q''}(Y''_{m+j})$ is defined by (iii), for each $j \in \{1, \ldots, r\}$, as the copy variable Y_{m+k} of query Q, where k is such that N_{m+k} is the jth element of the vector $\Phi_c[\mathcal{C}^{(Q'')}]$.

We now show that properties (1) through (6) of same-scale coverings mappings in Definition 3.1 are satisfied by the mapping $\mu_{Q''}$. (Hence, we conclude that the mapping $\mu_{Q''}$ is a same-scale coverings mapping from the query Q'' to the query Q.)

(1) This property in Definition 3.1 is satisfied by the mapping $\mu_{Q''}$ due to the fact that the association $\mathcal{A}_*$ has a (unity) valid assignment mapping, by definition of valid assignment mappings, and by definition of the mapping $\nu^{(i)}_Q$ used in the construction of the mapping $\mu_{Q''}$.

(2) This property in Definition 3.1 is satisfied by the mapping $\mu_{Q''}$ due to the fact that the association $\mathcal{A}_*$ has a (unity) t^*_Q-valid assignment mapping, by definition of t^*_Q-valid assignment mappings, and by definition of the mapping $\nu^{(i)}_Q$ used in the construction of the mapping $\mu_{Q''}$.

(3) This property in Definition 3.1 is satisfied by the mapping $\mu_{Q''}$ due to the facts that:
   - The noncopy-signature of the association $\mathcal{A}_*$ is (in case $m \geq 1$) a permutation of the list $[Y_1 \ \ldots \ Y_m]$, and by definition of the mapping $\mu'_{Q''}$ (and hence also of the mapping $\mu_{Q''}$) on the set of multiset noncopy variables of the query Q''.
   - The copy-signature of the association $\mathcal{A}_*$ is (in case $r \geq 1$) a permutation of the list $[\nu^{copy}_Q(Y_{m+1}) \ \ldots \ \nu^{copy}_Q(Y_{m+r})]$, and by definition of the mapping $\mu_{Q''}$ on the set of copy variables of the query Q''.

(4) This property in Definition 3.1 is satisfied by the mapping $\mu_{Q''}$ due to the fact that, by its definition, mapping $\mu_{Q''}$ maps each relational subgoal of the query Q'' into a unique element of the set $\mathcal{S}_{C(Q)}$ of subgoals of the query Q.

(5) This property in Definition 3.1 is satisfied by the mapping $\mu_{Q''}$ due to the facts that:
   - by its definition, mapping $\mu_{Q''}$ maps each copy-sensitive subgoal of the query Q'' into a subgoal of the query Q; and
   - the copy-signature of the association $\mathcal{A}_*$ does not have occurrences of the constant 1, hence mapping $\mu_{Q''}$ maps each copy-sensitive subgoal of the query Q'' into a *copy-sensitive* subgoal of the query Q.

[17]By definition of the wave of the query Q, the vector $\Phi_c[\mathcal{C}^{(Q'')}]$ may not contain the constant 1.

(6) Finally, this property in Definition 3.1 is satisfied by the mapping $\mu_{Q''}$ due to the definition of $\mu_{Q''}$ on the set of copy variables of the query Q'', see item (iii) in the definition of the mapping.

□

L.8 Extended Example: Basic Notation and Constructs

In this section we provide an extended example that illustrates the notions and constructions introduced in Sections L.2 through L.7 of the proof of Theorem 4.1. The example uses three CCQ queries, Q, Q', and Q''; each of the queries is an explicit-wave query by part (1) of Definition 4.1. y the results in this paper, for the queries Q and Q' of Example L.3 we have that $Q \equiv_C Q'$. In the beginning of the example, we exhibit a SCVM from Q' to Q. (The existence of the mapping is stipulated by Theorem 4.1.) At the same time, it is easy to ascertain that there does not exist a SCVM from the query Q'' to the query Q of Example L.3. Thus, by Theorem 4.1, $Q \equiv_C Q''$ cannot hold for the queries Q'' and Q of Example L.3. We build on this example a little later (see Example L.4 in Section L.9.3), to show how to use the proof of Theorem 4.1 to construct a counterexample database to $Q \equiv_C Q''$. At the end of Example L.3, we also illustrate the constructs of Section L.7, by discussing "the wave" of the query Q (see Definition L.10 in Section L.7) and the monomial classes of the queries Q and Q' that "have the wave" of Q. We also show that query Q'' does not "have the wave" of the query Q, and discuss the implications of this fact.

EXAMPLE L.3. *Let CCQ queries Q, Q', and Q'' be as follows.*

$Q(X_1) \leftarrow p(X_1, Y_1), p(X_1, X_2; Y_2), \{Y_1, Y_2\}$.
$Q'(X_1') \leftarrow p(X_1', Y_1'), p(X_1', X_2'; Y_2'), p(X_1', X_3'), \{Y_1', Y_2'\}$.
$Q''(X_1'') \leftarrow p(X_1'', X_2''), p(X_1'', Y_1''; Y_2''), \{Y_1'', Y_2''\}$.

Observe that by each of the three queries having exactly one copy-sensitive subgoal, each of Q, Q', and Q'' is an explicit-wave query. (See part (1) of Definition 4.1.)

By the results in this paper, e have that $Q \equiv_C Q'$. A SCVM μ from Q' to Q, as stipulated by Theorem 4.1, is $\mu = \{X_1' \rightarrow X_1, Y_1' \rightarrow Y_1, X_2' \rightarrow X_2, Y_2' \rightarrow Y_2, X_3' \rightarrow X_2\}$.

It is easy to see that there does not exist a SCVM from Q'' to Q. (Indeed, for each mapping from the terms of Q'' to the terms of Q, the mapping violates at least one of conditions (3) through (5) of Definition 3.1.) Thus, by Theorem 4.1, $Q \equiv_C Q''$ cannot hold. Later, we build on this example (see Example L.4 in Section L.9.3) to show how to use the proof of Theorem 4.1 to construct a counterexample database to $Q \equiv_C Q''$.

We now use queries Q and Q'' to illustrate the notation and constructions of the proof of Theorem 4.1, sequentially by subsections of the proof.

Constructing Database $D_{\bar{N}^{(i)}}(Q)$ for $\bar{N}^{(i)} = [2\ 3]$.

We first use the notation introduced in Section L.2 of the proof of Theorem 4.1. We have that $m = |M_{noncopy}| = |\{Y_1\}| = 1$, and that $r = |M_{copy}| = |\{Y_2\}| = 1$. The set $\mathcal{S}_{C(Q)}$ of the representative-element subgoals of the query Q is $\mathcal{S}_{C(Q)} = \{p(X_1, Y_1), p(X_1, X_2; Y_2)\}$, with $w = 1$. The reason is, the only relational subgoal of Q, call this subgoal h_1, is the representative element of the equivalence class $\{h_1\}$, and the only copy-sensitive subgoal of Q, call this subgoal h_2, is the representative element of the equivalence class $\{h_2\}$.

We now follow Section L.3 of the proof of Theorem 4.1, to illustrate the construction of a database in the family $\{D_{\bar{N}^{(i)}}(Q)\}$ for the query Q. We define mapping $\nu_0 = \{X_1 \rightarrow a, X_2 \rightarrow b\}$, for distinct constants a and b. Then we have that $S_0 = \{a, b\}$, and that $t^*_Q = (a)$. As $m + w = 2$ for the query Q, the vector $\bar{N}$ for Q comprises two variables, N_1 (intuitively for the multiset noncopy variable Y_1 of Q) and N_2 (intuitively for the copy variable Y_2 of Q). Let i be a fixed natural number (i.e., we treat i as the same constant throughout this example), and let the vector $\bar{N}^{(i)} = [2\ 3]$. That is, $N_1^{(i)} = 2$, and $N_2^{(i)} = 3$. For two distinct constants c and d, such that c and d are also distinct from the constants a and b used above to form the set S_0, let $S_1^{(i)} = \{c, d\}$; this set, of cardinality $N_1^{(i)}$, provides the domain (in the database) of the multiset noncopy variable Y_1 of Q. Then we have, by the definitions in Section L.3, that:

- $S_*^{(i)} = S_0 \bigcup S_1^{(i)}$;
- $\nu_Q^{(i)} = \{a \rightarrow X_1, b \rightarrow X_2, c \rightarrow Y_1, d \rightarrow Y_1\}$;
- $\nu^{copy}(Y_2) = N_2^{(i)} = 3$; *and*
- $\nu_Q^{copy}(Y_2) = N_2$.

For the set $\mathcal{S}^{(i)} = \{(c), (d)\}$, we have that $\nu_{(c)} = \{X_1 \rightarrow a, X_2 \rightarrow b, Y_1 \rightarrow c, Y_2 \rightarrow 3\}$ and that $\nu_{(d)} = \{X_1 \rightarrow a, X_2 \rightarrow b, Y_1 \rightarrow d, Y_2 \rightarrow 3\}$. We use mappings $\nu_{(c)}$ and $\nu_{(d)}$ each in one iteration of the main construction cycle for the database $D_{\bar{N}^{(i)}}(Q)$. The mapping $\nu_{(c)}$ applied to the two atoms in the set $\mathcal{S}_{C(Q)}$ results in ground atoms $p(a, c; 1)$ and $p(a, b; 3)$, and the mapping $\nu_{(d)}$ applied to the set $\mathcal{S}_{C(Q)}$ results in ground atoms $p(a, d; 1)$ and (again) $p(a, b; 3)$. Therefore, by construction we have the database $D_{\bar{N}^{(i)}}(Q) = \{\ p(a, c; 1), p(a, b; 3),\ p(a, d; 1)\ \}$. We will refer to the ground atom $p(a, c; 1)$ in the database $D_{\bar{N}^{(i)}}(Q)$ as d_1, to the atom $p(a, b; 3)$ as d_2, and to the atom $p(a, d; 1)$ as d_3.

Construction of the Terms for $\mathcal{F}_{(Q)}^{(Q'')}$.

We now follow Sections L.4 through L.7 of the proof of Theorem 4.1, to illustrate the construction of the terms for the function $\mathcal{F}_{(Q)}^{(Q'')}$, for the query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$ as constructed above in this example.

The number G of subgoals of the query Q'' is $G = 2$. Denote by g_1 the copy-sensitive subgoal $p(X_1'', Y_1''; Y_2'')$ of Q'', and by g_2 the relational subgoal $p(X_1'', X_2'')$ of the query. In query Q, denote by h_1 the subgoal $p(X_1, Y_1)$ and by h_2 the subgoal $p(X_1, X_2; Y_2)$.

There are nine associations between the $G = 2$ subgoals of the query Q'' and the three ground atoms (d_1, d_2, d_3) of the database $D_{\bar{N}^{(i)}}(Q)$. We list all the associations in this table:

ID	**DB**	$\Psi_a[\mathcal{A}_j]$	Φ_n	Φ_c	$\Gamma_{\bar{S}}^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$
$\mathcal{A}_1$	$[d_1, d_1]$	$[h_1, h_1]$	Y_1	1	$(a, c, 1)$
$\mathcal{A}_2$	$[d_1, d_2]$	$[h_1, h_2]$	Y_1	1	$(a, c, 1)$
$\mathcal{A}_3$	$[d_1, d_3]$	$[h_1, h_1]$	Y_1	1	$(a, c, 1)$
$\mathcal{A}_4$	$[d_2, d_1]$	$[h_2, h_1]$	X_2	N_2	$(a, b, 1), (a, b, 2), (a, b, 3)$
$\mathcal{A}_5$	$[d_2, d_2]$	$[h_2, h_2]$	X_2	N_2	$(a, b, 1), (a, b, 2), (a, b, 3)$
$\mathcal{A}_6$	$[d_2, d_3]$	$[h_2, h_1]$	X_2	N_2	$(a, b, 1), (a, b, 2), (a, b, 3)$
$\mathcal{A}_7$	$[d_3, d_1]$	$[h_1, h_1]$	Y_1	1	$(a, d, 1)$
$\mathcal{A}_8$	$[d_3, d_2]$	$[h_1, h_2]$	Y_1	1	$(a, d, 1)$
$\mathcal{A}_9$	$[d_3, d_3]$	$[h_1, h_1]$	Y_1	1	$(a, d, 1)$

The columns of the table, from left to right, refer to:

1. *Association ID, $\mathcal{A}_j$, for each of the associations $\mathcal{A}_1$ through $\mathcal{A}_9$ between query Q'' and database $D_{\bar{N}^{(i)}}(Q)$;*
2. *List of those ground atoms of the database that are associated by $\mathcal{A}_j$ with the subgoals of Q''; this list is to be read as "the first item in the list is associated by $\mathcal{A}_j$ with subgoal g_1 of Q''," and "the second item in the list is associated by $\mathcal{A}_j$ with subgoal g_2 of Q'';"*
3. *Atom-signature $\Psi_a[\mathcal{A}_j]$ of the association $\mathcal{A}_j$; this list is to be read as "the first item in the list is associated by $\mathcal{A}_j$ with subgoal g_1 of Q''," and "the second item in the list is associated by $\mathcal{A}_j$ with subgoal g_2 of Q'';"*
4. *Noncopy-signature $\Phi_n[\mathcal{A}_j]$ of the association $\mathcal{A}_j$;*
5. *Copy-signature $\Phi_c[\mathcal{A}_j]$ of the association $\mathcal{A}_j$; and*
6. *All the tuples contributed by the association $\mathcal{A}_j$ to the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$. (We assume that the columns of the relation $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$ are, from left to right, X_1'', Y_1'', and Y_2''.)*

For instance, the next-to-last row of the table is to be read as follows: Association $\mathcal{A}_8$ for the query Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$ as defined above, associates subgoal g_1 of Q'' with atom d_3 of $D_{\bar{N}^{(i)}}(Q)$, and associates subgoal g_2 of Q'' with atom d_2 of $D_{\bar{N}^{(i)}}(Q)$. Therefore, the atom-signature $\Psi_a[\mathcal{A}_8]$ associates g_1 with subgoal h_1 of the query Q, and associates g_2 with subgoal h_2 of Q. The noncopy-signature of $\mathcal{A}_8$ maps the multiset noncopy variable Y_1'' of the query Q'' to the multiset noncopy variable Y_1 of the query Q, and the copy-signature of $\mathcal{A}_8$ maps the copy variable Y_2'' of the query Q'' to the "copy value" 1 of the relational subgoal h_1 of the query Q. Finally, association $\mathcal{A}_8$ contributes tuple $(a, d, 1)$ to the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$.

The construction of the table uses the notation and definitions of Section L.4.1: The mapping $\psi_{\bar{N}^{(i)}}^{gen(Q)}$, as induced by the mapping $\nu_Q^{(i)}$, is defined as $\psi_{\bar{N}^{(i)}}^{gen(Q)} = \{d_1 \rightarrow h_1, d_2 \rightarrow h_2, d_3 \rightarrow h_1\}$. Then the atom-signature of, say, association $\mathcal{A}_8$ is computed as the vector with first element $\psi_{\bar{N}^{(i)}}^{gen(Q)}[d_3] = h_1$ and with second element $\psi_{\bar{N}^{(i)}}^{gen(Q)}[d_2] = h_2$.

The computation of the noncopy-signature $\Phi_n[\mathcal{A}_j]$ for each association $\mathcal{A}_j$ uses an arbitrary valid assignment mapping, call it θ, for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_j$, as well as the mapping $\nu_Q^{(i)}$ defined earlier in this example. Then the noncopy-signature of, say, association $\mathcal{A}_8$ is computed as the unary (because $m = 1$) vector $\Phi_n[\mathcal{A}_8] = [\nu_Q^{(i)}(\theta_8(Y_1''))] = [\nu_Q^{(i)}(d)] = Y_1$. The reason is, $\mathcal{A}_8$ generates a unique valid assignment mapping $\theta_8 = \{X_1'' \rightarrow a, Y_1'' \rightarrow d, Y_2'' \rightarrow 1, X_2'' \rightarrow b\}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$. (By definition, θ_8 is a unity t_Q^*-valid assignment mapping for Q'', $D_{\bar{N}^{(i)}}(Q)$, and $\mathcal{A}_8$.) Then we obtain for $\Phi_n[\mathcal{A}_8]$ that $[\nu_Q^{(i)}(\theta_8(Y_1''))] = [\nu_Q^{(i)}(d)] = Y_1$.

The computation of the copy-signature $\Phi_c[\mathcal{A}_j]$ for each association $\mathcal{A}_j$ uses the mapping $\nu^{copy-sig}$, which maps subgoal h_1 of the query Q to constant 1 (because h_1 is a relational atom), and maps copy-sensitive subgoal h_2 of Q, with copy variable Y_2, to variable $\nu_Q^{copy}(Y_2) = N_2$ in the vector $\bar{N}$. Then the copy-signature of, say, association $\mathcal{A}_8$ is computed as the unary (because $r = 1$) vector $\Phi_c[\mathcal{A}_8] = [\nu^{copy-sig}(\psi_{\bar{N}^{(i)}}^{gen(Q)}[d_3])] = [\nu^{copy-sig}(h_1)] = 1$.

Finally, we use all the t_Q^*-valid assignment mappings for Q'', $D_{\bar{N}^{(i)}}(Q)$, and each $\mathcal{A}_j$, to determine the contributions of each association $\mathcal{A}_j$ to the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$. For instance, for the association $\mathcal{A}_8$ we use the mapping θ_8 (which is the only t_Q^*-valid assignment mapping for $\mathcal{A}_8$) to construct the tuple $(a, d, 1)$ for the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$.

We now illustrate the construction of the set $\mathbb{A}_{Q''}^{(i)}$ for the query Q'' and database $D_{\bar{N}^{(i)}}(Q)$, as defined in Section L.5. The set comprises all the nine associations above: $\mathbb{A}_{Q''}^{(i)} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_9\}$. We use Proposition L.18 to conclude that the tuples shown in the last column of the table in this example are all and the only tuples in the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$ for the query and for the database.

Now we illustrate the construction of all the monomial classes, as equivalence classes of elements of the set $\mathbb{A}_{Q''}^{(i)}$ for the query Q'' and database $D_{\bar{N}^{(i)}}(Q)$, as defined in Section L.6. The classes are:

- *$\mathcal{C}_1^{(Q'')} = \{\mathcal{A}_1,\ \mathcal{A}_3,\ \mathcal{A}_7,\ \mathcal{A}_9\}$ (for the atom-signature $[h_1, h_1]$ of $\mathcal{A}_1$, $\mathcal{A}_3$, $\mathcal{A}_7$, and $\mathcal{A}_9$). We have that $\Phi_n^{\mathcal{C}_1^{(Q'')}} = [Y_1]$ and $\Phi_c^{\mathcal{C}_1^{(Q'')}} = [1]$.*
- *$\mathcal{C}_2^{(Q'')} = \{\mathcal{A}_2,\ \mathcal{A}_8\}$ (for the atom-signature $[h_1, h_2]$ of $\mathcal{A}_2$ and $\mathcal{A}_8$). We have that $\Phi_n^{\mathcal{C}_2^{(Q'')}} = [Y_1]$ and $\Phi_c^{\mathcal{C}_2^{(Q'')}} = [1]$.*
- *$\mathcal{C}_3^{(Q'')} = \{\mathcal{A}_4,\ \mathcal{A}_6\}$ (for the atom-signature $[h_2, h_1]$ of $\mathcal{A}_4$ and $\mathcal{A}_6$). We have that $\Phi_n^{\mathcal{C}_3^{(Q'')}} = [X_2]$ and $\Phi_c^{\mathcal{C}_3^{(Q'')}} = [N_2]$.*
- *$\mathcal{C}_4^{(Q'')} = \{\mathcal{A}_5\}$ (for the atom-signature $[h_2, h_2]$ of $\mathcal{A}_5$). We have that $\Phi_n^{\mathcal{C}_4^{(Q'')}} = [X_2]$ and $\Phi_c^{\mathcal{C}_4^{(Q'')}} = [N_2]$.*

Each of the four monomial classes has the noncopy-signature and the copy-signature of all its constituent associations. That is, for $\mathcal{C}_1^{(Q'')}$, we have above that $\Phi_n^{\mathcal{C}_1^{(Q'')}} = [Y_1]$ and $\Phi_c^{\mathcal{C}_1^{(Q'')}} = [1]$, and so on.

For those terms of the query Q that are not copy variables, we use the mapping ν_0 and the set $S_1^{(i)}$ to determine that $Dom_Q^{(i)}(X_1) = \{a\}$, $Dom_Q^{(i)}(X_2) = \{b\}$, and $Dom_Q^{(i)}(Y_1) = \{c, d\}$. Further, $DomLabel_Q(X_1) = DomLabel_Q(X_2) = 1$, and $DomLabel_Q(Y_1) = N_1$.

We now compute the multiplicity monomial for each of the four monomial classes for Q'' and for $D_{\bar{N}^{(i)}}(Q)$. For each of $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$, we have that $\Pi_{\Phi_n^{\mathcal{C}_1^{(Q'')}}} = \Pi_{\Phi_n^{\mathcal{C}_2^{(Q'')}}}$ is the product $\Pi_{j=1}^{1} DomLabel_Q(Y_j) = N_1$. Further, we have that $\Pi_{\Phi_c^{\mathcal{C}_1^{(Q'')}}} = \Pi_{\Phi_c^{\mathcal{C}_2^{(Q'')}}}$ is the product $\Pi_{j=1}^{1} 1$. Thus, the multiplicity monomial for each of $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ is the monomial $N_1 \times 1 = N_1$. Observe that $N_1^{(i)} = 2$ in our vector $\bar{N}^{(i)} = [2\ 3]$, and that the value $N_1^{(i)} = 2$ of the monomial N_1 for each of $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ is the correct count of the two tuples, $(a, c, 1)$ and $(a, d, 1)$, contributed by each of the two classes individually *to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q'', D_{\bar{N}^{(i)}}(Q))$. Note that the projection of all these tuples on Y_1'' (in $\Gamma_{\bar{S}}^{t_Q^*}(Q'', D_{\bar{N}^{(i)}}(Q))$) is exactly all the elements of the set $Dom_Q^{(i)}(Y_1)$; recall that Y_1 is the only element of the vector $\Pi_{\Phi_n^{\mathcal{C}_1^{(Q'')}}}$ and of the vector $\Pi_{\Phi_n^{\mathcal{C}_2^{(Q'')}}}$. (See Proposition L.26 for the details.)*

For each of $\mathcal{C}_3^{(Q'')}$ and $\mathcal{C}_4^{(Q'')}$, we have that $\Pi_{\Phi_n^{\mathcal{C}_3^{(Q'')}}} = \Pi_{\Phi_n^{\mathcal{C}_4^{(Q'')}}}$ is the product $\Pi_{j=1}^{1} DomLabel_Q(X_2) = 1$. Further, we have that $\Pi_{\Phi_c^{\mathcal{C}_3^{(Q'')}}} = \Pi_{\Phi_c^{\mathcal{C}_4^{(Q'')}}}$ is the product $\Pi_{j=1}^{1} N_2 = N_2$. Thus, the multiplicity monomial for each of $\mathcal{C}_3^{(Q'')}$ and $\mathcal{C}_4^{(Q'')}$ is the monomial $1 \times N_2 = N_2$. Observe that $N_2^{(i)} = 3$ in our vector $\bar{N}^{(i)} = [2\ 3]$, and that the value $N_2^{(i)} = 3$ of the monomial N_2 for each of $\mathcal{C}_3^{(Q'')}$ and $\mathcal{C}_4^{(Q'')}$ is the correct count of the three tuples, $(a, b, 1)$, $(a, b, 2)$, and $(a, b, 3)$, contributed by each of the two classes individually *to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q'', D_{\bar{N}^{(i)}}(Q))$. Note that the only element in the projection of all these tuples on Y_1'' (in $\Gamma_{\bar{S}}^{t_Q^*}(Q'', D_{\bar{N}^{(i)}}(Q))$) is exactly the only element of the set $Dom_Q^{(i)}(X_2)$; recall that X_2 is the only element of the vector $\Pi_{\Phi_n^{\mathcal{C}_3^{(Q'')}}}$ and of the vector $\Pi_{\Phi_n^{\mathcal{C}_4^{(Q'')}}}$. (See Proposition L.26 for the details.)*

For the construction of the function $\mathcal{F}_{(Q)}^{(Q'')}$ from the above multiplicity monomials, please see Example L.4 (Section L.9.3).

Construction of the Terms for $\mathcal{F}_{(Q)}^{(Q)}$.

We now follow Sections L.4 through L.7 of the proof of Theorem 4.1, to illustrate the construction of the terms for the function $\mathcal{F}_{(Q)}^{(Q)}$, for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$ as constructed above in this example. We follow steps similar to those used in the construction of the terms for the function $\mathcal{F}_{(Q)}^{(Q'')}$ for the query Q'', see preceding section of this example. As a result of the steps, we obtain four monomial classes for the query Q:

- *Monomial class $\mathcal{C}_1^{(Q)}$ has noncopy-signature $[Y_1]$ and copy-signature $[1]$; it contributes tuples $(a, c, 1)$ and $(a, d, 1)$ to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$. The multiplicity monomial for $\mathcal{C}_1^{(Q)}$ is the term N_1.*
- *Monomial class $\mathcal{C}_2^{(Q)}$ has noncopy-signature $[Y_1]$ and copy-signature $[N_2]$; it contributes tuples $(a, c, 1)$, $(a, c, 2)$, $(a, c, 3)$, $(a, d, 1)$, $(a, d, 2)$, and $(a, d, 3)$ to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$. The multiplicity monomial for $\mathcal{C}_2^{(Q)}$ is the term $N_1 \times N_2$.*
- *Monomial class $\mathcal{C}_3^{(Q)}$ has noncopy-signature $[X_2]$ and copy-signature $[1]$; it contributes tuple $(a, b, 1)$ to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$. The multiplicity monomial for $\mathcal{C}_3^{(Q)}$ is the term 1 (i.e., constant 1).*
- *Monomial class $\mathcal{C}_4^{(Q)}$ has noncopy-signature $[X_2]$ and copy-signature $[N_2]$; it contributes tuples $(a, b, 1)$, $(a, b, 2)$, and $(a, b, 3)$ to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$. The multiplicity monomial for $\mathcal{C}_4^{(Q)}$ is the term N_2.*

For the construction of the function $\mathcal{F}_{(Q)}^{(Q)}$ from the above multiplicity monomials, please see Example L.4 (Section L.9.3).

Construction of the Terms for $\mathcal{F}_{(Q)}^{(Q')}$.

The construction of the terms for the function $\mathcal{F}_{(Q)}^{(Q')}$ is almost identical to that for the function $\mathcal{F}_{(Q)}^{(Q)}$, because the only difference between Q and Q' is in the presence of an extra subgoal $p(X_1', X_3')$ in Q', and this subgoal does not introduce any multiset variables (of Q'). Thus, we obtain the same monomial classes for Q' and for Q (modulo renaming all the variables of Q into "same-name" variables of Q', for instance variable X_1 of Q corresponds to variable X_1' of Q'). Please see Example L.4 (Section L.9.3) for the construction of the function $\mathcal{F}_{(Q)}^{(Q')}$ from the above multiplicity monomials.

The Wave of Query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$.

We now use the monomial classes of the queries Q, Q', and Q'' to illustrate the notion of the "wave of CCQ query," which was introduced in Section L.7. By Definition L.10, the wave of the query Q of this example, w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, is the product $N_1 \times N_2$. By Proposition L.33, the query Q has a nonempty monomial class w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$, specifically the monomial class $\mathcal{C}_2^{(Q)}$, such that the multiplicity monomial of the class $\mathcal{C}_2^{(Q)}$ is exactly the wave of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$.

Now the query Q' of this example also has a nonempty monomial class w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of that monomial class is the wave of the query Q. (Recall that in this example we obtained the same monomial classes for Q' and for Q, modulo renaming all the variables of Q into "same-name" variables of Q'.) Thus, by Proposition L.34, there must exist a SCVM from the query Q' to the query Q. Indeed, the same-scale covering mapping μ of the beginning of this example is built as specified in the proof of Proposition L.34.

Finally, observe that for the query Q'' of this example and for each nonempty monomial class of Q'' w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$, the multiplicity monomial of the monomial class is not *the wave of the query Q. Thus, Proposition L.34 does not apply. Indeed, as we showed in the beginning of this example, there does not exist a SCVM from Q'' to Q. Then from Theorem 4.1 we conclude that $Q \equiv_C Q''$ does not hold for the queries Q and Q'' of this example. Please see Example L.4 (Section L.9.3) for a discussion of how the database $D_{\bar{N}^{(i)}}(Q)$ constructed earlier here (in Example L.3) is a counterexample database for $Q \equiv_C Q''$. In addition, for the wave $\mathcal{P}_*^{(Q)} = N_1 \times N_2$ of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$, Example L.4 points out the presence of the monomial $\mathcal{P}_*^{(Q)}$ in the functions $\mathcal{F}_{(Q)}^{(Q)}$ and $\mathcal{F}_{(Q)}^{(Q')}$, for the queries Q and Q' of this example, and also points out the absence of the monomial $\mathcal{P}_*^{(Q)}$ in the function $\mathcal{F}_{(Q)}^{(Q'')}$, for the query Q'' of this example.* □

L.9 Putting Together the Function $\mathcal{F}_{(Q)}^{(Q'')}$

In this section we define the function $\mathcal{F}_{(Q)}^{(Q'')}$ outlined in the beginning of Section L.6. The only entities that we use to specify the function $\mathcal{F}_{(Q)}^{(Q'')}$ are (a) the multiplicity monomials defined in Section L.6, and (b) the noncopy-signatures and the copy-signatures of the monomial classes introduced in Section L.6. Recall that each of the multiplicity monomials, as well as each of the copy-signatures, is in terms of the variables in the vector $\bar{N}$; we will show in this section how to "convert" the noncopy-signatures into collections of variables in the vector $\bar{N}$.

In this section we specify the function $\mathcal{F}_{(Q)}^{(Q'')}$ for an arbitrary CCQ query Q, for the family $\{D_{\bar{N}^{(i)}}(Q)\}$ of databases defined using Q (as outlined in Section L.3), and for an arbitrary CCQ query Q'' that satisfies the restrictions (w.r.t.

the query Q) of Section L.2.

L.9.1 Notation, Definitions, Basic Results

For CCQ queries Q and Q'' satisfying the requirements of Section L.2, suppose that Q and Q'' are also such that (as discussed in the beginning of Section L.6) the set of all nonempty monomial classes for Q'' and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ is not empty. That is, suppose that $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$ is the set of all nonempty monomial classes for Q'' and for $\{D_{\bar{N}^{(i)}}(Q)\}$, with $n^* \geq 1$.

We begin the exposition by making a few straightforward observations. Recall the notation $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ of Section L.6. Then the following proposition is immediate from Proposition L.26, for the cases where $n^* \geq 2$.

Proposition L.35. *Given CCQ queries Q and Q'', suppose that the set $\mathcal{C}[Q''] = \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$ of all nonempty monomial classes for Q'' and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ has $n^* \geq 2$ elements. Further, let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$, for some $n, p \in \{1, \ldots, n^*\}$, be two monomial classes in the set $\mathcal{C}[Q'']$, such that the noncopy-signatures of $\mathcal{C}_n^{(Q'')}$ and of $\mathcal{C}_p^{(Q'')}$ are not identical vectors. Then for each $i \in \mathbb{N}_+$, we have that $\Gamma^{(i)}[\mathcal{C}_n^{(Q'')}] \bigcap \Gamma^{(i)}[\mathcal{C}_p^{(Q'')}] = \emptyset$.* □

For the other observations in this subsection, we will need the following notation. For an arbitrary monomial class $\mathcal{C}_n^{(Q'')} \neq \emptyset$, $n \in \{1, \ldots, n^*\}$, in case $r \geq 1$, we denote the elements of the copy-signature vector $\Phi_c[\mathcal{C}_n^{(Q'')}]$ as $[V_{j1[n]}, \ldots, V_{jr[n]}]$. Recall that, by definition of copy-signature, for each $k \in \{1, \ldots, r\}$ we have that $V_{jk[n]} \in \{1, N_{m+1}, \ldots, N_{m+w}\}$, where the N-values are variables in the vector $\bar{N}$. (In case $r = 0$, $\Phi_c[\mathcal{C}_n^{(Q'')}]$ is the empty vector by definition.)

Definition L.11. *(**Unconditional dominance for monomial classes**) Let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ be two (not necessarily distinct) monomial classes in the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$. (That is, $n, p \in \{1, \ldots, n^*\}$.) Further, let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ have the same noncopy-signature. Then we say that* monomial class $\mathcal{C}_n^{(Q'')}$ unconditionally dominates monomial class $\mathcal{C}_p^{(Q'')}$ *if:*

- *We have the case $r = 0$; or*
- *We have the case $r \geq 1$, and for the pair $(V_{jk[p]}, V_{jk[n]})$ for each $k \in \{1, \ldots, r\}$, we have that either $V_{jk[p]} = 1$, or $V_{jk[p]} = V_{jk[n]}$.*

□

We observe that the unconditional-dominance relation is reflexive by definition.

The following important property of unconditional-dominance holds by the results of Section L.6.

Proposition L.36. *Let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ be two monomial classes in the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$. (That is, $n, p \in \{1, \ldots, n^*\}$.) Suppose that $\mathcal{C}_n^{(Q'')}$ unconditionally dominates $\mathcal{C}_p^{(Q'')}$. Then for each $i \in \mathbb{N}_+$, we have that $\Gamma^{(i)}[\mathcal{C}_p^{(Q'')}] \subseteq \Gamma^{(i)}[\mathcal{C}_n^{(Q'')}]$.* □

The following result is immediate from Definition L.11.

Proposition L.37. *Let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ be two monomial classes in the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$. (That is, $n, p \in \{1, \ldots, n^*\}$.) Further, let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ have the same noncopy-signature. Then we have that (i) $\mathcal{C}_n^{(Q'')}$ unconditionally dominates $\mathcal{C}_p^{(Q'')}$* and *$\mathcal{C}_p^{(Q'')}$ unconditionally dominates $\mathcal{C}_n^{(Q'')}$, if and only if (ii) $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ have the same copy-signature.* □

From reflexivity of unconditional-dominance and from Propositions L.36 and L.37, we obtain the following result.

Proposition L.38. *Let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ be two monomial classes in the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$. (That is, $n, p \in \{1, \ldots, n^*\}$.) Further, let $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ have the same noncopy-signature and the same copy-signature. Then for each $i \in \mathbb{N}_+$, we have that $\Gamma^{(i)}[\mathcal{C}_p^{(Q'')}] = \Gamma^{(i)}[\mathcal{C}_n^{(Q'')}]$.* □

In Example L.3 in Section L.8, monomial class $\mathcal{C}_1^{(Q'')}$ unconditionally dominates a *nonidentical* [18] (to $\mathcal{C}_1^{(Q'')}$) monomial class $\mathcal{C}_2^{(Q'')}$, and vice versa (that is, monomial class $\mathcal{C}_2^{(Q'')}$ unconditionally dominates monomial class $\mathcal{C}_1^{(Q'')}$). Similarly, monomial class $\mathcal{C}_3^{(Q'')}$ of the same Example unconditionally dominates a nonidentical (to $\mathcal{C}_3^{(Q'')}$) monomial class $\mathcal{C}_4^{(Q'')}$, and vice versa.

We now outline an algorithm template that we call Removal of duplicate monomial classes. The input is the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$, $n^* \geq 1$, for CCQ query Q'' and for family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$; the output is a subset (denoted by $\mathbb{C}(Q'')$) of the input. The algorithm template involves three steps:

(1) Partition all elements of the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$ into equivalence classes, where two distinct (in case $n^* \geq 2$) monomial classes $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$, for $n \neq p \in \{1, \ldots, n^*\}$, belong to the same equivalence class if and only if $\mathcal{C}_n^{(Q'')}$ and $\mathcal{C}_p^{(Q'')}$ have identical noncopy-signatures and identical copy-signatures.

(2) Use an arbitrary algorithm, call it Choose-Representative-Element, to choose one element of each of the equivalence classes as the representative element of the equivalence class.

(3) Return the set $\mathbb{C}(Q'')$ of representative elements (only) of all of the equivalence classes of the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$.

A specific algorithm instantiating the algorithm template Removal of duplicate monomial classes is obtained by specifying the algorithm Choose-Repres- entative-Element. Observe that $\mathbb{C}(Q'') \neq \emptyset$ for all nonempty inputs to Removal of duplicate monomial classes and for all choices of the algorithm Choose-Representative-Element.

Clearly, in general, the contents of the set $\mathbb{C}(Q'')$ depend on the algorithm, Choose-Representative-Element, for choosing the representative element of each equivalence class, within the algorithm template Removal of duplicate monomial classes. (For instance, given as input the four monomial classes of Example L.3 in Section L.8, the algorithm

[18]Recall (see Section L.6) that the identity of a monomial class is determined by its atom-signature.

template could produce four different outputs.) At the same time, the following two results, Proposition L.39 and Proposition L.40, hold regardless of the choice of the algorithm CHOOSE-REPRESENTATIVE-ELEMENT when instantiating the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES.

PROPOSITION L.39. *Given the set $\mathcal{C}[Q'']$ of all nonempty monomial classes for CCQ query Q'' and for family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, and given an algorithm* CHOOSE-REPRESENTATIVE-ELEMENT *to instantiate the algorithm template* REMOVAL OF DUPLICATE MONOMIAL CLASSES. *Then for the output $\mathbb{C}(Q'')$ of the resulting algorithm given the input $\mathcal{C}[Q'']$, the following two facts hold:*

(i) For each pair (e_1, e_2) of distinct (i.e., $e_1 \neq e_2$) elements of the set $\mathbb{C}(Q'')$, e_1 and e_2 either have different noncopy-signatures or have different copy-signatures; and

(ii) For all $i \in \mathbb{N}_+$, we have that:

$$\bigcup_{\mathcal{C} \in \mathcal{C}[Q'']} \Gamma^{(i)}[\mathcal{C}] = \bigcup_{\mathcal{C}' \in \mathbb{C}(Q'')} \Gamma^{(i)}[\mathcal{C}'].$$

□

Proposition L.39 is immediate from Proposition L.38 and from the construction of the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES.

In the next result, Proposition L.40, we denote by $|S|$ the cardinality of set S. Proposition L.40 holds by construction of the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES.

PROPOSITION L.40. *Let $\mathcal{C}[Q'']$ be the set of all nonempty monomial classes for CCQ query Q'' and for family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$. Let a_1 and a_2 be two instantiations of the algorithm template* REMOVAL OF DUPLICATE MONOMIAL CLASSES*, where a_1 and a_2 may use different ways of choosing the representative element of each equivalence class generated by the algorithm. Let $\mathbb{C}_j(Q'')$ be the output of algorithm a_j on the input $\mathcal{C}[Q'']$, for $j \in \{1, 2\}$. Then we have that:*

(1) $|\mathbb{C}_1(Q'')| = |\mathbb{C}_2(Q'')|$; and

(2) There exists an isomorphism, call it μ, from the set $\mathbb{C}_1(Q'')$ to the set $\mathbb{C}_2(Q'')$, such that for each element e of the set $\mathbb{C}_1(Q'')$, e and $\mu(e)$ have the same copy-signature as well as the same noncopy-signature.

□

For our purpose of constructing a function that would return the multiplicity of the tuple t^*_Q in the bag Res_C $(Q'', D_{\bar{N}^{(i)}}(Q))$, for all $i \in \mathbb{N}_+$, Propositions L.39 and L.40 let us refer to the output of an arbitrary instantiation of the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES as *the* output, $\mathbb{C}(Q'')$, of the algorithm (template). We can show that the unconditional-dominance relation of Definition L.11 is reflexive, antisymmetric, and transitive on the set $\mathbb{C}(Q'')$. As such, the unconditional-dominance relation is a partial order on that set.

We now use any standard algorithm[19] for removal of all those monomial classes from the set $\mathbb{C}(Q'')$ that (monomial classes) are unconditionally dominated by some other monomial class in the set $\mathbb{C}(Q'')$. Clearly, the output of that algorithm is a unique subset, call it $\mathbb{C}^{nondom}(Q'')$, of the set $\mathbb{C}(Q'')$. We say that the set $\mathbb{C}^{nondom}(Q'')$ is *the result of dropping unconditionally-dominated monomial classes from the set $\mathbb{C}(Q'')$.*

[19]We use the observation that the unconditional-dominance relation of Definition L.11 is a partial order on the set $\mathbb{C}(Q'')$.

Using Propositions L.39 and L.40, it is straightforward to show the following.

PROPOSITION L.41. *Let $\mathcal{C}[Q'']$ be the set of all nonempty monomial classes for CCQ query Q'' and for family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$. Let a_1 and a_2 be two instantiations of the algorithm template* REMOVAL OF DUPLICATE MONOMIAL CLASSES*, where a_1 and a_2 may use different ways of choosing the representative element of each equivalence class generated by the algorithm. Let $\mathbb{C}_j(Q'')$ be the output of algorithm a_j on the input $\mathcal{C}[Q'']$, for $j \in \{1, 2\}$. Further, let $\mathbb{C}^{nondom}_j(Q'')$ be the result of dropping unconditionally-dominated monomial classes from the set $\mathbb{C}_j(Q'')$, for $j \in \{1, 2\}$. Then for all $i \in \mathbb{N}_+$, we have that:*

$$\bigcup_{\mathcal{C} \in \mathcal{C}[Q'']} \Gamma^{(i)}[\mathcal{C}] = \bigcup_{\mathcal{C}' \in \mathbb{C}^{nondom}_1(Q'')} \Gamma^{(i)}[\mathcal{C}']$$

$$= \bigcup_{\mathcal{C}'' \in \mathbb{C}^{nondom}_2(Q'')} \Gamma^{(i)}[\mathcal{C}''].$$

□

As a result of Proposition L.41, for our purpose (of constructing a function that would return the multiplicity of the tuple t^*_Q in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q)))$ we can refer to each set $\mathbb{C}^{nondom}(Q'')$ as *the* set $\mathbb{C}^{nondom}(Q'')$ for the set of all nonempty monomial classes for CCQ query Q'' and for family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, regardless of the identity of the exact set $\mathbb{C}(Q'')$ as discussed above.

L.9.2 The Easy Case of Constructing $\mathcal{F}^{(Q'')}_{(Q)}$

The following observation lets us finalize the construction of the function $\mathcal{F}^{(Q'')}_{(Q)}$ for the case where all elements of the set $\mathbb{C}^{nondom}(Q'')$ have different noncopy-signatures. In this case, we have that the function $\mathcal{F}^{(Q'')}_{(Q)}$ is always a multivariate polynomial in terms of the variables in the vector $\bar{N}$ and with integer coefficients, on the entire domain $\mathcal{N}$ of the function. The result of Proposition L.42 is immediate from Propositions L.35, L.39, and L.40. For the definition of multiplicity monomial for monomial class, see Section L.6.3.

PROPOSITION L.42. *Given the set $\mathbb{C}^{nondom}(Q'')$ for a CCQ query Q'' and for a family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the elements of the set $\mathbb{C}^{nondom}(Q'')$ have $|\mathbb{C}^{nondom}(Q'')|$ distinct noncopy-signatures. Then, for each $i \in \mathbb{N}_+$, the cardinality of the set $\Gamma^{(t^*_Q)}_{\bar{S}}$ $(Q'', D_{\bar{N}^{(i)}}(Q))$, can be computed exactly, by substituting the values in the vector $\bar{N}^{(i)}$ (specifically value $N^{(i)}_j$ as value of variable N_j, for each $j \in \{1, \ldots, m+w\}$) into the formula*

$$\Sigma_{\mathcal{C} \in \mathbb{C}^{nondom}(Q'')} \mathcal{M}[\mathcal{C}]$$

where $\mathcal{M}[\mathcal{C}]$ is the multiplicity monomial of monomial class $\mathcal{C}$. □

That is, under the conditions of Proposition L.42, the function $\mathcal{F}^{(Q'')}_{(Q)}$ is given by the formula

$$\Sigma_{\mathcal{C} \in \mathbb{C}^{nondom}(Q'')} \mathcal{M}[\mathcal{C}]$$

in the Proposition.

For instance, if we choose the set $\{\mathcal{C}_1(Q''), \mathcal{C}_3(Q'')\}$ as the set $\mathbb{C}^{nondom}(Q'')$ for Example L.3 in Section L.8, then, for query Q'' of the Example, the function $\mathcal{F}^{(Q'')}_{(Q)}$ is the following multivariate polynomial in terms of the variables in the vector $\bar{N}$: $\mathcal{F}^{(Q'')}_{(Q)} = N_1 + N_2$. For the vector $\bar{N}^{(i)}$ of Example L.3, $\mathcal{F}^{(Q'')}_{(Q)}$ returns the correct multiplicity, 5, of the tuple $t^*_Q = (a)$ in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$. (For the details, see Example L.4 in Section L.9.3.)

Corollary L.1. *In case $r \leq 1$, given a CCQ query Q'' and a family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$. Then, for each $i \in \mathbb{N}_+$, the cardinality of the set $\Gamma^{(t^*_Q)}_{\bar{S}}(Q'', D_{\bar{N}^{(i)}}(Q))$, can be computed exactly, by substituting the values in the vector $\bar{N}^{(i)}$ (specifically value $N^{(i)}_j$ as value of variable N_j, for each $j \in \{1, \ldots, m + w\}$) into the formula*

$$\Sigma_{\mathcal{C} \in \mathbb{C}^{nondom}(Q'')} \mathcal{M}[\mathcal{C}]$$

where $\mathcal{M}[\mathcal{C}]$ is the multiplicity monomial of monomial class $\mathcal{C}$. □

Proof. (sketch) The reason that Corollary L.1 of Proposition L.42 holds is that in case $r \leq 1$, either $r = 0$ holds and then the copy-signature of each monomial class for Q'' is an empty vector, or $r = 1$ holds and then the copy-signature of each monomial class for Q'' is either the vector $[1]$ or the vector $[N]$, for exactly one variable name N across all the copy-signatures. Then the unconditional-dominance relation of Definition L.11 holds for each pair of the monomial classes for the query Q'' such that the classes in the pair have the same noncopy-signature, and hence all elements of the set $\mathbb{C}^{nondom}(Q'')$ have different noncopy-signatures. □

Observe that it is not obvious how to generalize the statement of Corollary L.1 to the case $r \geq 2$. Indeed, even when $w \leq 1$ (and hence we still have exactly one variable name N as the only possible variable across all the noncopy-signatures),[20] the case $r = 2$ already presents us with the (theoretical) possibility where two monomial classes for Q'', with the same noncopy-signature, might have respective copy-signatures $[1\ N]$ and $[N\ 1]$, for which unconditional-dominance does not hold in either direction.

L.9.3 Illustration

In this subsection we build on Example L.3 (of Section L.8), to show the construction of the functions $\mathcal{F}^{(Q)}_{(Q)}$, $\mathcal{F}^{(Q')}_{(Q)}$, and $\mathcal{F}^{(Q'')}_{(Q)}$, for the three queries of Example L.3 and for the database constructed in Example L.3. We also show how that database is a counterexample to $Q \equiv_C Q''$, for the queries Q and Q'' of Example L.3. Finally, we continue our discussion (started in Example L.3) of "the wave of" the query Q, and explore the relationship between that entity and the multivariate polynomials for $\mathcal{F}^{(Q)}_{(Q)}$, $\mathcal{F}^{(Q')}_{(Q)}$, and $\mathcal{F}^{(Q'')}_{(Q)}$.

[20]By Proposition L.4, $r \geq 1$ implies $w \geq 1$, hence from $r \geq 2$ and $w \leq 1$ we have the exact equality $w = 1$.

Example L.4. *Recall the queries Q, Q', and Q'' of Example L.3 (of Section L.8). Recall also the database $D_{\bar{N}^{(i)}}(Q)$ that we constructed in Example L.3 for the query Q. In this example we build functions $\mathcal{F}^{(Q)}_{(Q)}$, $\mathcal{F}^{(Q')}_{(Q)}$, and $\mathcal{F}^{(Q'')}_{(Q)}$, for the three queries Q, Q', and Q'' and for the database $D_{\bar{N}^{(i)}}(Q)$. We also show how the database $D_{\bar{N}^{(i)}}(Q)$ is a counterexample to $Q \equiv_C Q''$. Finally, we continue our discussion (started in Example L.3) of "the wave of" the query Q, and explore the relationship between that entity and the multivariate polynomials for $\mathcal{F}^{(Q)}_{(Q)}$, $\mathcal{F}^{(Q')}_{(Q)}$, and $\mathcal{F}^{(Q'')}_{(Q)}$.*

Construction of Function $\mathcal{F}^{(Q)}_{(Q)}$.

For the query Q and database $D_{\bar{N}^{(i)}}(Q)$, we came up in Example L.3 with four monomial classes $\mathcal{C}^{(Q)}_1$, $\mathcal{C}^{(Q)}_2$, $\mathcal{C}^{(Q)}_3$, and $\mathcal{C}^{(Q)}_4$. By Definition L.11, monomial class $\mathcal{C}^{(Q)}_2$ unconditionally-dominates monomial class $\mathcal{C}^{(Q)}_1$. The reason is, $\mathcal{C}^{(Q)}_1$ and $\mathcal{C}^{(Q)}_2$ have identical noncopy-signatures, the copy-signature of the monomial class $\mathcal{C}^{(Q)}_1$ is $[1]$, and the copy-signature of the monomial class $\mathcal{C}^{(Q)}_2$ is $[N_2]$. (See Section L.9.1 for further details on unconditional-dominance.) Similarly, monomial class $\mathcal{C}^{(Q)}_4$ unconditionally-dominates monomial class $\mathcal{C}^{(Q)}_3$. Thus, the set $\{\mathcal{C}^{(Q)}_2, \mathcal{C}^{(Q)}_4\}$ is the *set $\mathbb{C}^{nondom}(Q)$ as defined in Section L.9.1.*

Then, by Proposition L.42, the function $\mathcal{F}^{(Q)}_{(Q)}$ is the following multivariate polynomial in terms of the variables in the vector $\bar{N}$: $\mathcal{F}^{(Q)}_{(Q)} = N_1 \times N_2 + N_2$. For the vector $\bar{N}^{(i)} = [2\ 3]$ that we fixed in Example L.3, $\mathcal{F}^{(Q)}_{(Q)}$ returns the correct multiplicity, 9, of the tuple $t^*_Q = (a)$ in the bag $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$.

Construction of Function $\mathcal{F}^{(Q')}_{(Q)}$.

For the query Q' and database $D_{\bar{N}^{(i)}}(Q)$ of Example L.3, we use the reasoning similar to that for constructing the function $\mathcal{F}^{(Q)}_{(Q)}$ earlier in this example, to obtain the function $\mathcal{F}^{(Q')}_{(Q)} = N_1 \times N_2 + N_2$. As the multivariate polynomials $\mathcal{F}^{(Q)}_{(Q)}$ and $\mathcal{F}^{(Q')}_{(Q)}$ are identical to each other, they output the same answer for each $\bar{N}^{(i)} \in \mathcal{N}$.

Construction of Function $\mathcal{F}^{(Q'')}_{(Q)}$.

For the query Q'' and database $D_{\bar{N}^{(i)}}(Q)$, we came up in Example L.3 with four monomial classes $\mathcal{C}^{(Q'')}_1$, $\mathcal{C}^{(Q'')}_2$, $\mathcal{C}^{(Q'')}_3$, and $\mathcal{C}^{(Q'')}_4$. By Definition L.11, monomial classes $\mathcal{C}^{(Q'')}_1$ and $\mathcal{C}^{(Q'')}_2$ unconditionally-dominate each other. (The reason is, $\mathcal{C}^{(Q'')}_1$ and $\mathcal{C}^{(Q'')}_2$ have identical noncopy-signatures, and have identical copy-signatures.) Similarly, monomial classes $\mathcal{C}^{(Q'')}_3$ and $\mathcal{C}^{(Q'')}_4$ unconditionally-dominate each other. Suppose that we choose the set $\{\mathcal{C}^{(Q'')}_1, \mathcal{C}^{(Q'')}_3\}$ as the set $\mathbb{C}^{nondom}(Q'')$ as defined in Section L.9.1. (See Section L.9.1 for the discussion of possible choices for the set $\mathbb{C}^{nondom}(Q'')$.)

Then, by Proposition L.42, the function $\mathcal{F}^{(Q'')}_{(Q)}$ is the following multivariate polynomial in terms of the variables in the vector $\bar{N}$: $\mathcal{F}^{(Q'')}_{(Q)} = N_1 + N_2$. For the vector $\bar{N}^{(i)} = [2\ 3]$ that we fixed in Example L.3, $\mathcal{F}^{(Q'')}_{(Q)}$ returns the correct multiplicity, 5, of the tuple $t^*_Q = (a)$ in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$.

Database $D_{\bar{N}^{(i)}}(Q)$ Is a Counterexample to $Q \equiv_C Q''$.

From the different sizes of the sets $\Gamma^{t^*_Q}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$ and $\Gamma^{t^*_Q}_{\bar{S}}(Q'', D_{\bar{N}^{(i)}}(Q))$ on the database $D_{\bar{N}^{(i)}}(Q)$, as discussed earlier in this example, we have that the database $D_{\bar{N}^{(i)}}(Q)$ is a counterexample to $Q \equiv_C Q''$.

The Wave of Q in the Functions $\mathcal{F}^{(Q)}_{(Q)}$, $\mathcal{F}^{(Q')}_{(Q)}$.

Recall from Example L.3 (Section L.8) our discussion of "the wave of" the query Q of that example. By Definition L.10, the wave of that query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ is the monomial $N_1 \times N_2$.

For the queries Q, Q' and Q'' as given in this example (and, with respectively identical definitions, in Example L.3), we now contrast the functions $\mathcal{F}^{(Q)}_{(Q)}$ and $\mathcal{F}^{(Q')}_{(Q)}$, on the one hand, with the function $\mathcal{F}^{(Q'')}_{(Q)}$, on the other hand. Recall that $\mathcal{F}^{(Q)}_{(Q)} = \mathcal{F}^{(Q')}_{(Q)} = N_1 \times N_2 + N_2$. Observe that each of $\mathcal{F}^{(Q)}_{(Q)}$ and $\mathcal{F}^{(Q')}_{(Q)}$ has a term that is exactly the wave of the query Q (w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$), that is the term $N_1 \times N_2$. In contrast, the function $\mathcal{F}^{(Q'')}_{(Q)} = N_1 + N_2$ clearly does not have a term that is the wave $N_1 \times N_2$ of the query Q w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$. □

L.9.4 Beyond the Easy Case: Example

In this subsection we exhibit a CCQ query Q, such that the function $\mathcal{F}^{(Q)}_{(Q)}$, w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, cannot be computed using the results of Section L.9.2, specifically using Proposition L.42. This example motivates the development, in Section L.9.5, of a more general (as compared to that of Section L.9.2) approach toward constructing the function $\mathcal{F}^{(Q)}_{(Q)}$ for CCQ query Q'' and family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$.

EXAMPLE L.5. *Let CCQ query Q be as follows.*

$$Q(X_1) \leftarrow r(X_1, Y_1, Y_2, X_2; Y_3), r(X_1, Y_1, Y_2, X_3; Y_4), \{Y_1, Y_2, Y_3, Y_4\}.$$

Toward Constructing Function $\mathcal{F}^{(Q)}_{(Q)}$ for the Databases $\{ D_{\bar{N}^{(i)}}(Q) \}$.

We show how to develop a database in the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ for the query Q, and start constructing function $\mathcal{F}^{(Q)}_{(Q)}$ w.r.t. the databases in the family. (See Example L.6 for the completion of the construction.)

We begin by following Section L.3 of the proof of Theorem 4.1. Fix an $i \in \mathbb{N}_+$. Let the vector $\bar{N}^{(i)}$, for this fixed i, of values of the variables in the vector $\bar{N} = [N_1\ N_2\ N_3\ N_4]$, be $\bar{N}^{(i)} = [1\ 2\ 3\ 5]$. Here, each N_j in $\bar{N}$ is generated for the variable Y_j of Q, for $j \in \{1, 2, 3, 4\}$. We use $\nu_0(X_1) = a$ (hence $t^*_Q = (a)$), $\nu_0(X_2) = b$, and $\nu_0(X_3) = c$. Let $S^{(i)}_1 = \{e\}$, and let $S^{(i)}_2 = \{f, g\}$. These setting generate, for the fixed i, the database $D_{\bar{N}^{(i)}}(Q) = \{\ r(a, e, f, b; 3), r(a, e, g, b; 3),\ r(a, e, f, c; 5),\ r(a, e, g, c; 5)\ \}$. We will refer to the ground atoms in the set $D_{\bar{N}^{(i)}}(Q)$, from left to right, as d_1 through d_4. Denote by h_1 the first subgoal of the query Q, and by h_2 its second subgoal. By construction of $D_{\bar{N}^{(i)}}(Q)$, we have that $\psi^{gen(Q)}_{\bar{N}^{(i)}}[d_1] = \psi^{gen(Q)}_{\bar{N}^{(i)}}[d_2] = h_1$, and that $\psi^{gen(Q)}_{\bar{N}^{(i)}}[d_3] = \psi^{gen(Q)}_{\bar{N}^{(i)}}[d_4] = h_2$.

We now follow Sections L.4 through L.9.1 of the proof of Theorem 4.1, to construct the monomial classes for the function $\mathcal{F}^{(Q)}_{(Q)}$, for the query Q and for the database $D_{\bar{N}^{(i)}}(Q)$ as generated above in this example. As a result of the construction steps,[21] *we obtain four monomial classes for the query Q:*

- *Monomial class $\mathcal{C}^{(Q)}_1$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_3\ N_3]$; it contributes to the set $\Gamma^{t^*_Q}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$, with columns (from left to right) $X_1\ Y_1\ Y_2\ Y_3\ Y_4$, nine tuples $(a, e, f, 1, 1)$ through $(a, e, f, 3, 3)$ (that is, tuples $(a, e, f, 1, 1)$, $(a, e, f, 1, 2)$, $(a, e, f, 1, 3)$, $(a, e, f, 2, 1)$, $\ldots$, $(a, e, f, 3, 2)$, $(a, e, f, 3, 3)$), as well as nine tuples $(a, e, g, 1, 1)$ through $(a, e, g, 3, 3)$.*
- *Monomial class $\mathcal{C}^{(Q)}_2$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_3\ N_4]$; it contributes to the set $\Gamma^{t^*_Q}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$ fifteen tuples $(a, e, f, 1, 1)$ through $(a, e, f, 3, 5)$, as well as fifteen tuples $(a, e, g, 1, 1)$ through $(a, e, g, 3, 5)$.*
- *Monomial class $\mathcal{C}^{(Q)}_3$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_4\ N_3]$; it contributes to the set $\Gamma^{t^*_Q}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$ fifteen tuples $(a, e, f, 1, 1)$ through $(a, e, f, 5, 3)$, as well as fifteen tuples $(a, e, g, 1, 1)$ through $(a, e, g, 5, 3)$.*
- *Monomial class $\mathcal{C}^{(Q)}_4$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_4\ N_4]$; it contributes to the set $\Gamma^{t^*_Q}_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$ twenty five tuples $(a, e, f, 1, 1)$ through $(a, e, f, 5, 5)$, as well as twenty five tuples $(a, e, g, 1, 1)$ through $(a, e, g, 5, 5)$.*

While all four of the above monomial classes have the same noncopy-signature, none of the classes unconditionally dominates (see Definition L.11) any other monomial class in the set $\{\mathcal{C}^{(Q)}_1, \mathcal{C}^{(Q)}_2, \mathcal{C}^{(Q)}_3, \mathcal{C}^{(Q)}_4\}$. □

L.9.5 The General Case of Constructing $\mathcal{F}^{(Q'')}_{(Q)}$

In this subsection we address the construction of the function $\mathcal{F}^{(Q'')}_{(Q)}$ for the general case, as opposed to the case considered in Section L.9.2. That is, we introduce an approach to computing, for a query Q'' and database $D_{\bar{N}^{(i)}}(Q)$, the cardinality of the set $\Gamma^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$ – and therefore the multiplicity of the tuple t^*_Q in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$ – for those cases where at least two distinct elements of the set $\mathbb{C}^{nondom}\ (Q'')$ could have the same noncopy-signature. The approach introduced in this subsection is applicable to constructing the function $\mathcal{F}^{(Q'')}_{(Q)}$ for *all* cases, including the special case of Section L.9.2.

Consider Example L.5 of Section L.9.4: For the CCQ query Q and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ of the example, the set $\mathcal{C}(Q) = \{\mathcal{C}^{(Q)}_1, \mathcal{C}^{(Q)}_2, \mathcal{C}^{(Q)}_3, \mathcal{C}^{(Q)}_4\}$ has four monomial classes with the same noncopy-signature $[Y_1\ Y_2]$ and with the respective copy-signatures $[N_3\ N_3]$, $[N_3\ N_4]$, $[N_4\ N_3]$, and $[N_4\ N_4]$. Clearly, no unconditional-dominance of Definition L.11 holds for any pair of monomial classes in the set $\mathcal{C}(Q)$. Hence the set $\mathbb{C}^{nondom}(Q)$ (see Section L.9.1) is the set $\mathcal{C}(Q)$. Further, as the set $\mathbb{C}^{nondom}(Q)$ does not satisfy the conditions of Proposition L.42, the function $\mathcal{F}^{(Q)}_{(Q)}$ for the example *cannot* be constructed using Proposition L.42.

[21]These steps are outlined in significant detail in Example L.3, albeit using queries that are different from the queries of the current example.

Indeed, it is easy to see that $\mathcal{F}^{(Q)}_{(Q)}$ for Example L.5 is *not* the sum of the multiplicity monomials for the elements of the above set $\mathbb{C}^{nondom}(Q)$. Specifically, Example L.5 shows that w.r.t. the fixed database $D_{\bar{N}^{(i)}}(Q)$ used in the example, each element of the set $\mathbb{C}^{nondom}(Q)$ contributes to the set $\Gamma^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$ *the same* tuple $(a, e, f, 1, 1)$.

We summarize that the problem with the general case considered in this subsection is that the multiplicity of the tuples contributed, to the set $\Gamma^{(t^*_Q)}(\ldots)$, by distinct monomial classes for the query in question, cannot always be added up to obtain the correct total contribution of the classes to that set. At the same time, we know from Proposition L.25 that for each $i \in \mathbb{N}_+$, the size of the set $\Gamma^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$ is the size of the union $\bigcup_{j=1}^{n^*} \Gamma^{(i)}[\mathcal{C}^{(Q'')}_j]$, over all the nonempty monomial classes $\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}$ for the query Q'' w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$. We also know, from Proposition L.35, that for each pair $(\mathcal{C}^{(Q'')}_n, \mathcal{C}^{(Q'')}_p)$ of distinct monomial classes among $\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}$, such that $\mathcal{C}^{(Q'')}_n$ and $\mathcal{C}^{(Q'')}_p$ have distinct noncopy signatures, it holds that the intersection of the sets $\Gamma^{(i)}[\mathcal{C}^{(Q'')}_n]$ and $\Gamma^{(i)}[\mathcal{C}^{(Q'')}_p]$ is empty for each $i \in \mathbb{N}_+$. Thus, to obtain the function $\mathcal{F}^{(Q'')}_{(Q)}$ for the general case, it remains to consider the (perhaps nonempty) intersections of the sets $\Gamma^{(i)}[\mathcal{C}^{(Q'')}_n]$ and $\Gamma^{(i)}[\mathcal{C}^{(Q'')}_p]$ *only* for those pairs $(\mathcal{C}^{(Q'')}_n, \mathcal{C}^{(Q'')}_p)$ where $\mathcal{C}^{(Q'')}_n$ and $\mathcal{C}^{(Q'')}_p$ have *the same* noncopy signature.

Thus, the two last missing links in (finally) constructing the function $\mathcal{F}^{(Q'')}_{(Q)}$ for the general case, are based on the following two results, Propositions L.43 and L.44. Example L.6 in Section L.9.6 provides an illustration of the construction of the function $\mathcal{F}^{(Q)}_{(Q)}$ for the query Q and for the database of Example L.5 in Section L.9.4.

Proposition L.43. *Suppose the monomial classes in the set $\mathcal{C}[Q''] = \{\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}\}$ are indexed (by 1, 2, ..., n^*) in such a way that for all triples $(\mathcal{C}^{(Q'')}_{j_1}, \mathcal{C}^{(Q'')}_{j_2}, \mathcal{C}^{(Q'')}_{j_3})$, with $1 \leq j_1 < j_2 < j_3 \leq n^*$, it cannot be that (a) $\mathcal{C}^{(Q'')}_{j_1}$ and $\mathcal{C}^{(Q'')}_{j_3}$ have the same noncopy signature, and (b) $\mathcal{C}^{(Q'')}_{j_1}$ and $\mathcal{C}^{(Q'')}_{j_2}$ have different noncopy signatures. Further, let $n \in \{1, \ldots, n^*\}$ be such that n is the number of distinct noncopy signatures of all the elements of the set $\mathcal{C}[Q'']$. Finally, let $k_0 = 0$ and, for this value of n, let $1 \leq k_1 < k_2 < \ldots < k_n = n^*$ be such that for each $j \in \{1, 2, \ldots, n\}$, all monomial classes $\mathcal{C}^{(Q'')}_{k_{j-1}+1}, \mathcal{C}^{(Q'')}_{k_{j-1}+2}, \ldots, \mathcal{C}^{(Q'')}_{k_j}$ have the same noncopy signature.*[22]

Let $i \in \mathbb{N}_+$. Then the cardinality of the set $\Gamma^{(t^*_Q)}(Q''$, $D_{\bar{N}^{(i)}}(Q))$ is given exactly as the sum

$$\sum_{j=0}^{n-1} \quad \Big| \bigcup_{l=1}^{(k_{j+1})-(k_j)} \Gamma^{(i)}[\mathcal{C}^{(Q'')}_{(k_j)+l}] \Big| .$$

(Here, each $\mathcal{C}^{(Q'')}_{(k_j)+l}$ referenced in the formula is an element of the set $\mathcal{C}[Q''] = \{\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}\}$.) □

(As usual, we denote by $|S|$ the cardinality of the set S. The result of Proposition L.43 is immediate from Propositions L.25 and L.35.)

Now we will be able to compute correctly the function $\mathcal{F}^{(Q'')}_{(Q)}$ for each $i \in \mathbb{N}_+$, as soon as we are able to evaluate the formulas of the form

$$\Big| \bigcup_{l=1}^{(k_{j+1})-(k_j)} \Gamma^{(i)}[\mathcal{C}^{(Q'')}_{(k_j)+l}] \Big|, \qquad (1)$$

as introduced in Proposition L.43. We compute the value of such formulas using the basic *inclusion-exclusion principle* for computing the cardinality of the union of several sets. All that the inclusion-exclusion principle requires as inputs is the cardinalities of the *intersections* of the relevant (groups of) sets. (We handle the case of determining the size of each individual set, S, in the input to the cardinality-of-union formula, as the special case of "intersection of S with itself." As will be clear from the statement of Proposition L.44, this special case is captured correctly – as expected – by Proposition L.30.)

Thus, our next result, Proposition L.44, is the final missing link in the construction of the function $\mathcal{F}^{(Q'')}_{(Q)}$, as Proposition L.44 tells us how to compute correctly the cardinalities of the intersections of sets of the form $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$, using *only the elements of the vector $\bar{N}$,* that is only variables N_1 through N_{m+w} and nothing else. (More precisely, Proposition L.44 gives us a formula where, for each specific $i \in \mathbb{N}_+$, we can compute the cardinalities of all the requisite intersections by using the specific values, in $\bar{N}^{(i)}$ for this value i, of the respective variables in $\bar{N}$. The formula itself is in terms of $\bar{N}$ only, and does not use $\bar{N}^{(i)}$.)

For the formulation of Proposition L.44, assume that in the set $\mathcal{C}[Q''] = \{\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}\}$ there exist (at least) k monomial classes, for some $k \in \{1, \ldots, n^*\}$, whose noncopy signature is a given vector Ξ of length m. Suppose that for some fixed $i \in \mathbb{N}_+$, we want to compute the cardinality of the intersection of the sets $\Gamma^{(i)}$ (using the notation of Proposition L.43) for exactly these k elements of the set $\mathcal{C}[Q'']$. To make easier the notation in the formal results to follow, assume w.l.o.g. that the elements of the set $\mathcal{C}[Q'']$ are indexed in such a way that for all these chosen k elements of $\mathcal{C}[Q'']$ that have noncopy-signature Ξ, these k monomial classes *are the first k elements of the set $\mathcal{C}[Q'']$*. (That is, these k monomial classes are the elements $\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_k$ of the set $\mathcal{C}[Q'']$.)

Now let us refer to the vector Ξ as $\Phi_n^{\mathcal{C}^{(Q'')}_1}$. (By our indexing of the elements of the set $\mathcal{C}[Q'']$, as introduced in the previous paragraph, the noncopy-signature of the monomial class $\mathcal{C}^{(Q'')}_1$ is exactly Ξ.) The reason that we want to refer to the vector Ξ as $\Phi_n^{\mathcal{C}^{(Q'')}_1}$ is that we want, in the formal results to follow, to use the notation $\Pi_{\Phi_n^{\mathcal{C}^{(Q'')}_1}}$ introduced in

[22]All of the above conditions together just say that the elements of the set $\mathcal{C}[Q'']$ are indexed in such a way that, in the sequence $\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}$, we first have all the monomial classes with some noncopy-signature NS_1, then all the monomial classes with a different noncopy-signature NS_2, and so on. That is, for each noncopy-signature, NS, of at least one element of the set $\mathcal{C}[Q'']$, all monomial classes in $\mathcal{C}[Q'']$ that have the noncopy-signature NS are "grouped together" in the sequence $\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}$.

Section L.6.3.

We also use the following notation of Section L.9.1: For an arbitrary monomial class $\mathcal{C}_l^{(Q'')} \in \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_k^{(Q'')}\}$, for the $k \in \{1, \ldots, n^*\}$ fixed as explained above, in case where $r \geq 1$ we denote the elements of the copy-signature vector $\Phi_c[\mathcal{C}_l^{(Q'')}]$ as $[V_{j1[l]}, V_{j2[l]}, \ldots, V_{jr[l]}]$. (In case where $r = 0$, the copy-signature vector of each of $\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_k^{(Q'')}$ is the empty vector by definition.) Further, in case $r \geq 1$, for the element $V_{js[l]}$ of the vector $\Phi_c[\mathcal{C}_l^{(Q'')}]$ (for an arbitrary $s \in \{1, \ldots, r\}$) and for an $i \in \mathbb{N}_+$, we denote by $V_{js[l]}^{(i)}$ (a) the constant 1 in case $V_{js[l]} = 1$, and (b) the value $N_u^{(i)}$ from the vector $\bar{N}^{(i)}$ in case $V_{js[l]}$ is the element N_u, for an $u \in \{m+1, \ldots, m+w\}$, of the vector $\bar{N}$.

We are finally ready to phrase the final formal result needed in the construction of the function $\mathcal{F}_{(Q)}^{(Q'')}$. As has been noted earlier in this subsection, Example L.6 in Section L.9.6 provides an illustration of the construction of the function $\mathcal{F}_{(Q)}^{(Q)}$ for the query Q and for the database of Example L.5 in Section L.9.4.

Proposition L.44. *In the set $\mathcal{C}[Q''] = \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$, let (at least) the first k elements, for some $k \in \{1, \ldots, n^*\}$, have the same noncopy-signature $\Phi_n^{\mathcal{C}_1^{(Q'')}}$. Then, for an arbitrary $i \in \mathbb{N}_+$, the cardinality of the set*

$$\bigcap_{s=1}^{k} \Gamma^{(i)}[\mathcal{C}_s^{(Q'')}]$$

is provided by substituting the constants in $\bar{N}^{(i)}$ as the values of the respective variables in $\bar{N}$, into the formula:

- $\Pi_{\Phi_n^{\mathcal{C}_1^{(Q'')}}}$, *in case where $r = 0$; and*
- $\Pi_{\Phi_n^{\mathcal{C}_1^{(Q'')}}} \times \Pi_{u=1}^{r} \, min(V_{ju[1]}, V_{ju[2]}, \ldots, V_{ju[k]})$, *in case where $r \geq 1$.*

□

Proof. For the case where $r = 0$, observe that for each pair of monomial classes among $\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_k^{(Q'')}$, the monomial classes in the pair unconditionally dominate each other, by Definition L.11. Therefore, the result of Proposition L.44 is immediate from Proposition L.36.

For the case where $r \geq 1$, the result of Proposition L.44 is immediate from Lemma L.1. □

To formulate Lemma L.1, we use the following terminology. For an element $\mathcal{C}^{(Q'')}$ of the set $\mathcal{C}[Q''] = \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$, and for an $i \in \mathbb{N}_+$, consider the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$. In case where $m \geq 1$, let an m-tuple $t^{(M)}$ be an arbitrary tuple in the projection of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the multiset noncopy variables of the query Q'' (in some arbitrary fixed order of these variables). Then we say that $z \geq 1$ tuples $t_1, t_2, \ldots, t_z$ in the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ *agree on the multiset-noncopy projection* $t^{(M)}$, if we have that the set projection of the subset $\{t_1, t_2, \ldots, t_z\}$ of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on all the multiset noncopy variables of the query Q'' (in the same fixed order) is a singleton set $\{t^{(M)}\}$. In case where $m = 0$, we say that (by default) all the tuples in the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ agree on the multiset-noncopy projection that is the empty tuple.

Lemma L.1. *Suppose $r \geq 1$. In the set $\mathcal{C}[Q''] = \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$, let (at least) the first k elements, for some $k \in \{1, \ldots, n^*\}$, have the same noncopy-signature $\Phi_n^{\mathcal{C}_1^{(Q'')}}$. Let $i \in \mathbb{N}_+$. Let $t^{(M)}$ be an arbitrary tuple in the projection of the set*

$$\mathcal{S} = \bigcap_{s=1}^{k} \Gamma^{(i)}[\mathcal{C}_s^{(Q'')}]$$

on all the multiset noncopy variables of the query Q'', in case $m \geq 1$, and let $t^{(M)}$ be the empty tuple in case $m = 0$. Then, for the number K of all those tuples in the set $\mathcal{S} = \bigcap_{s=1}^{k} \Gamma^{(i)}[\mathcal{C}_s^{(Q'')}]$ that agree on the multiset-noncopy projection $t^{(M)}$, we have that the value of K is provided by substituting the constants in $\bar{N}^{(i)}$ as the values of the respective variables in $\bar{N}$, into the formula

$$K = \Pi_{u=1}^{r} \, min(V_{ju[1]}, V_{ju[2]}, \ldots, V_{ju[k]}).$$

□

Proof. (sketch) Assume a fixed $i \in \mathbb{N}_+$. The proof of Lemma L.1 is immediate from Proposition L.27, which exhibits the structure of the projection of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ (for an arbitrary monomial class $\mathcal{C}^{(Q'')}$ in the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$) on the set of all copy variables of the query Q'', and from Proposition L.28, which explores the "symmetries" of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on the databases in the family $\{D_{\bar{N}^{(i)}}(Q)\}$. Specifically, we have that:

- The values in the projection of the set $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ on the set of all copy variables of the query Q'' are natural numbers in a specified range according to the copy signature of the monomial class $\mathcal{C}^{(Q'')}$. More precisely, let the copy signature for the monomial class $\mathcal{C}^{(Q'')}$ be $[V_{j1} \; V_{j2} \; \ldots \; V_{jr}]$. Then for each $u \in \{1, \ldots, r\}$, each value in the projection of $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ onto the copy variable Y''_{m+u} of Q'' is a natural number belonging to the set $\{1, \ldots, V_{ju}^{(i)}\}$. Moreover, for each $u \in \{1, \ldots, r\}$ and for each value $v_u \in \{1, \ldots, V_{ju}^{(i)}\}$, the tuple $(v_1, v_2, \ldots, v_r)$ is in the projection of $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$ onto all the copy variables $Y''_{m+1}, Y''_{m+2}, \ldots, Y''_{m+r}$ of Q'', in this order.
- Consider now the monomial classes $\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_k^{(Q'')}$ in the statement of Lemma L.1. For the fixed $i \in \mathbb{N}_+$ and for each $u \in \{1, \ldots, r\}$, denote by Z_u the value $min(V_{ju[1]}^{(i)}, V_{ju[2]}^{(i)}, \ldots, V_{ju[k]}^{(i)})$. Then we can show that:
  - For each $u \in \{1, \ldots, r\}$ and for each value $v_u \in \{1, \ldots, Z_u\}$, the tuple $(v_1, v_2, \ldots, v_r)$ is in the projection of the set $\bigcap_{s=1}^{k} \Gamma^{(i)}[\mathcal{C}_s^{(Q'')}]$ onto all the copy variables $Y''_{m+1}, Y''_{m+2}, \ldots, Y''_{m+r}$ of Q'', in this order; and
  - Whenever, for at least one $u \in \{1, \ldots, r\}$, the value v_u is *not* an element of the set $\{1, \ldots, Z_u\}$, then we have that the tuple $(v_1, v_2, \ldots, v_r)$ is *not* in the projection of the set $\bigcap_{s=1}^{k} \Gamma^{(i)}[\mathcal{C}_s^{(Q'')}]$ onto all the copy variables $Y''_{m+1}, Y''_{m+2}, \ldots, Y''_{m+r}$ of Q'', in this order.

□

At the conclusion of this subsection, we observe that by the inclusion-exclusion principle for unions of sets, the value of the function $\mathcal{F}_{(Q)}^{(Q'')}$ for each $i \in \mathbb{N}_+$, that is the cardinality of the set $\Gamma^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$, can also be computed exactly using the set $\mathbb{C}^{nondom}(Q'')$ of Section L.9.1. That is, we can use the set $\mathbb{C}^{nondom}(Q'')$, rather than the set $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$ of *all* nonempty monomial classes for query Q'' and database $D_{\bar{N}^{(i)}}(Q)$ (cf. Proposition L.43):

PROPOSITION L.45. *Suppose the monomial classes in the set $\mathbb{C}^{nondom}(Q'') = \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_p^{(Q'')}\}$ are indexed (by 1, 2, …, p) in such a way that for all triples $(\mathcal{C}_{j_1}^{(Q'')}, \mathcal{C}_{j_2}^{(Q'')}, \mathcal{C}_{j_3}^{(Q'')})$, with $1 \leq j_1 < j_2 < j_3 \leq p$, it cannot be that (a) $\mathcal{C}_{j_1}^{(Q'')}$ and $\mathcal{C}_{j_3}^{(Q'')}$ have the same noncopy signature, and (b) $\mathcal{C}_{j_1}^{(Q'')}$ and $\mathcal{C}_{j_2}^{(Q'')}$ have different noncopy signatures. Further, let $n \in \{1, \ldots, p\}$ be such that n is the number of distinct noncopy signatures of all the elements of the set $\mathcal{C}[Q'']$. Finally, let $k_0 = 0$ and, for this value of n, let $1 \leq k_1 < k_2 < \ldots < k_n = p$ be such that for each $j \in \{1, 2, \ldots, n\}$, all monomial classes $\mathcal{C}_{k_{j-1}+1}^{(Q'')}, \mathcal{C}_{k_{j-1}+2}^{(Q'')}, \ldots, \mathcal{C}_{k_j}^{(Q'')}$ have the same noncopy signature.*[23]

Let $i \in \mathbb{N}_+$. Then the cardinality of the set $\Gamma^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$ is given exactly as the sum

$$\sum_{j=0}^{n-1} \quad | \bigcup_{l=1}^{(k_{j+1})-(k_j)} \Gamma^{(i)}[\mathcal{C}_{(k_j)+l}^{(Q'')}] \quad | \ .$$

(Here, each $\mathcal{C}_{(k_j)+l}^{(Q'')}$ referenced in the formula is an element of the set $\mathbb{C}^{nondom}(Q'') = \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_p^{(Q'')}\}$.) □

Observe that Proposition L.42, which constructs the function $\mathcal{F}_{(Q)}^{(Q'')}$ for a special "easy" case as considered in Section L.9.2, is an immediate corollary of Proposition L.45 and of the definition of the set $\mathbb{C}^{nondom}(Q'')$.

For an illustration, consider again the function $\mathcal{F}_{(Q)}^{(Q)}$ of Example L.4 in Section L.9.3. When we construct function $\mathcal{F}_{(Q)}^{(Q)}$ using all four monomial classes of the example, the inclusion-exclusion formulae of this current subsection correctly account for the fact that the $\Gamma^{(i)}()$ for the monomial class $\mathcal{C}_1^{(Q)}$ is a subset of the $\Gamma^{(i)}()$ for the monomial class $\mathcal{C}_2^{(Q)}$ on all the databases in question. We observe the similar effect when considering how the inclusion-exclusion formulae account for the relationship between the monomial classes $\mathcal{C}_3^{(Q)}$ and $\mathcal{C}_4^{(Q)}$ of the example. Hence, by the inclusion-exclusion principle for unions of sets, the construction of the function $\mathcal{F}_{(Q)}^{(Q)}$ using all four monomial classes of the example results in the same function as the construction using the set $\mathbb{C}^{nondom}(Q)$, as shown in the example.

[23]Similarly to the condition of Proposition L.43, all of the above conditions together just say that the elements of the set $\mathbb{C}^{nondom}(Q'')$ are indexed in such a way that, in the sequence $\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_p^{(Q'')}$, we first have all the monomial classes with some noncopy-signature NS_1, then all the monomial classes with a different noncopy-signature NS_2, and so on. That is, for each noncopy-signature, NS, of at least one element of the set $\mathbb{C}^{nondom}(Q'')$, all monomial classes in $\mathbb{C}^{nondom}(Q'')$ that have the noncopy-signature NS are "grouped together" in the sequence $\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_p^{(Q'')}$.

L.9.6 Illustration of the general construction

In this subsection, we provide an illustration of the results of Section L.9.5, by following the construction of the function $\mathcal{F}_{(Q)}^{(Q)}$ for the query Q and for the database of Example L.5 in Section L.9.4. As discussed in the beginning of Section L.9.5, the construction cannot be carried out correctly when using just the results of the "easy-case" Section L.9.2.

EXAMPLE L.6. *We refer to the query Q and database $D_{\bar{N}^{(i)}}(Q)$ of Example L.5 in Section L.9.4. In this example we construct the function $\mathcal{F}_{(Q)}^{(Q)}$ for that query Q and for the entire family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, $i \geq 1$. In addition, we illustrate the correctness of the construction, by using the multiplicity of the tuple t^*_Q of Example L.5 in the combined-semantics answer to the query Q on the specific database $D_{\bar{N}^{(i)}}(Q)$ of Example L.5.*

Recall that the query Q has four nonempty monomial classes, $\mathcal{C}_1^{(Q)}$, $\mathcal{C}_2^{(Q)}$, $\mathcal{C}_3^{(Q)}$, and $\mathcal{C}_4^{(Q)}$, w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$. (Refer to Example L.5 for the details.) Each of the monomial classes has noncopy-signature $[Y_1\ Y_2]$; the copy-signatures of the four monomial classes are $[N_3\ N_3]$, $[N_3\ N_4]$, $[N_4\ N_3]$, and $[N_4\ N_4]$, in this order.

To construct function $\mathcal{F}_{(Q)}^{(Q)}$ for the query Q and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, we use Proposition L.43, to establish that for each $i \in \mathbb{N}_+$, the cardinality of the set $\Gamma^{(t^*_Q)}(Q, D_{\bar{N}^{(i)}}(Q))$ is given exactly as the sum

$$| \bigcup_{l=1}^{4} \Gamma^{(i)}[\mathcal{C}_l^{(Q)}] \quad | \ .$$

For greater succinctness of the formulae to follow, we label more compactly each of the sets $\Gamma^{(i)}[\mathcal{C}_1^{(Q)}]$ through $\Gamma^{(i)}[\mathcal{C}_4^{(Q)}]$ used in the above formula, as follows: Denote $\Gamma^{(i)}[\mathcal{C}_1^{(Q)}]$ by A, $\Gamma^{(i)}[\mathcal{C}_2^{(Q)}]$ by B, $\Gamma^{(i)}[\mathcal{C}_3^{(Q)}]$ by C, and $\Gamma^{(i)}[\mathcal{C}_4^{(Q)}]$ by D. Then, by the inclusion-exclusion principle for unions of sets, we have that the above union formula can be rewritten as follows:

$$|A \cup B \cup C \cup D| = |A| + |B| + |C| + |D| - |A \cap B| - |A \cap C|$$

$$-|A \cap D| - |B \cap C| - |B \cap D| - |C \cap D| + |A \cap B \cap C| + |A \cap B \cap D|$$

$$+|A \cap C \cap D| + |B \cap C \cap D| - |A \cap B \cap C \cap D|.$$

We now use Proposition L.44 to obtain the cardinality of each of the set intersections in the right-hand side of this formula. First, observe that the multipliers $\Pi_{\Phi_n^{\mathcal{C}_l^{(Q)}}}$, for $l \in \{1, 2, 3, 4\}$, are all equal to each other, and are each the product $N_1 \times N_2$. (This is due to the fact that all the four monomial classes have the same noncopy signature.)

Thus, what remains to be done, in the construction of the formula $\mathcal{F}_{(Q)}^{(Q)}$, is to compute the products

$$\Pi_{u=1}^{k}\ min(V_{ju[1]}, V_{ju[2]}, \ldots, V_{ju[k]})$$

of Proposition L.44 for all the above set intersections, for all k between 2 (for $|A \cap B|$, $|A \cap C|$, …, $|C \cap D|$) and 4 (for $|A \cap B \cap C \cap D|$). (For convenience in the statement of Proposition L.44, the indexing in the product $\Pi_{u=1}^{k}\ min(V_{ju[1]}, V_{ju[2]}, \ldots, V_{ju[k]})$ assumes that each time we look at the cardinality of the intersection of the sets $\Gamma^{(i)}()$ for the first k consecutive *elements of the set $\{\mathcal{C}_1^{(Q)}, \ldots, \mathcal{C}_4^{(Q)}\}$. That is, the statement of the Proposition assumes reindexing of the elements of the set $\{\mathcal{C}_1^{(Q)}, \ldots, \mathcal{C}_4^{(Q)}\}$ "as needed." This assumption needs to be kept in mind when understanding the consecutive indexing by u in the formula $\Pi_{u=1}^{k}\ min(V_{ju[1]}, V_{ju[2]}, \ldots, V_{ju[k]})$ in this example.) Then, by multiplying each of these products by $N_1 \times N_2$ and by "putting the multiplication results back correctly" into our inclusion-exclusion formula for the cardinality of the union of Proposition L.43, we will obtain the expression for the function $\mathcal{F}_{(Q)}^{(Q)}$.*

We make the basic observation that each $min()$ expression for this example will result in N_3 (when the only value in the min expression is N_3 – that is, when all arguments of the min expression are the same variable N_3), in N_4 (when the only value in the min expression is N_4), or in $min(N_3, N_4)$ (in all the remaining cases, regardless of the number of times each of N_3 and N_4 is an argument of the min expression). To make the writeup more concise, we refer to the latter minimum expression as Z. (That is, we denote by Z the expression $min(N_3, N_4)$.) In addition, we denote by T the term $N_1 \times N_2$. As the union expressions of Proposition L.43 are uniform (as expressed using the elements of the vector $\bar{N}^{(i)}$) across all values of $i \in \mathbb{N}_+$, in the remainder of this example we switch to the elements of the vector $\bar{N}$ as basic blocks in the construction of the formula $\mathcal{F}_{(Q)}^{(Q)}$, and refrain from clarifying all the time that the values for specific i can be obtained by substituting the elements of the vector $\bar{N}^{(i)}$ for the respective elements of $\bar{N}$ in the expressions that we are to obtain.

By the formula of Proposition L.44, we obtain that:
$|A| = T \times (N_3)^2$; $|B| = |C| = T \times N_3 \times N_4$;
$|D| = T \times (N_4)^2$; $|A \cap B| = |A \cap C| = T \times N_3 \times Z$;
$|B \cap D| = |C \cap D| = T \times N_4 \times Z$.

Further, it is easy to check that each of the remaining cardinalities, in the inclusion-exclusion formula for $|A \cup B \cup C \cup D|$, equals $T \times Z^2$.

Thus, we obtain that

$$(|A \cup B \cup C \cup D|)/T = (N_3)^2 + 2N_3N_4 + (N_4)^2 - 2ZN_3 - 2ZN_4 + Z^2.$$

That is, we obtain that, by Propositions L.43 and L.44,

$$\mathcal{F}_{(Q)}^{(Q)} = N_1N_2 \times [(N_3)^2 + 2N_3N_4 + (N_4)^2 - 2ZN_3 - 2ZN_4 + Z^2].$$

Recall that Z here denotes the expression $min(N_3, N_4)$. Observe that in this formula for $\mathcal{F}_{(Q)}^{(Q)}$, for each of the terms

$$-2N_1 \times N_2 \times min(N_3, N_4) \times N_3,$$

$$-2N_1 \times N_2 \times min(N_3, N_4) \times N_4,\ and$$

$$+N_1 \times N_2 \times (min(N_3, N_4))^2,$$

we have that none of the three terms corresponds to monomial classes for the query Q. Thus, none of these terms is "backed up" by assignments from the query Q to any database $D_{\bar{N}^{(i)}}(Q)$.

Due to the presence of the term $min(N_3, N_4)$ in the above expression for the function $\mathcal{F}_{(Q)}^{(Q)}$, the function is not a multivariate polynomial (in terms of the elements of the vector $\bar{N}$) on the entire domain $\mathcal{N}$ of the function. At the same time:

- *For all $i \in \mathbb{N}_+$ such that $N_3^{(i)} \leq N_4^{(i)}$ in the vector $\bar{N}^{(i)}$, we have that (after we substitute $Z = min(N_3, N_4) = N_3$ and then cancel out in the resulting formula) the function $\mathcal{F}_{(Q)}^{(Q)}$ on this subdomain of $\mathcal{N}$ is the following multivariate polynomial in terms of the elements of the vector $\bar{N}$:*

$$\mathcal{F}_{(Q)}^{(Q)} = N_1 \times N_2 \times (N_4)^2.$$

- *Similarly, for all $i \in \mathbb{N}_+$ such that $N_3^{(i)} \geq N_4^{(i)}$ in the vector $\bar{N}^{(i)}$, we have that (after we substitute $Z = min(N_3, N_4) = N_4$ and then cancel out in the resulting formula) the function $\mathcal{F}_{(Q)}^{(Q)}$ on this subdomain of $\mathcal{N}$ is the following multivariate polynomial in terms of the elements of the vector $\bar{N}$:*

$$\mathcal{F}_{(Q)}^{(Q)} = N_1 \times N_2 \times (N_3)^2.$$

Specifically, for the vector $\bar{N}^{(i)} = [1\ 2\ 3\ 5]$ of Example L.5, we have that $N_3 = 3 \leq N_4 = 5$. Hence, for this i we have that $\mathcal{F}_{(Q)}^{(Q)}(\bar{N}^{(i)}) = N_1^{(i)} \times N_2^{(i)} \times (N_4^{(i)})^2$. Observe that the result of evaluating this expression $\mathcal{F}_{(Q)}^{(Q)}(\bar{N}^{(i)})$ for this i is $1 \times 2 \times (5)^2 = 50$. *This value 50 is the correct multiplicity of the tuple $t_Q^* = (a)$ of Example L.5 in the combined-semantics answer to the query Q on the specific database $D_{\bar{N}^{(i)}}(Q)$ of Example L.5. (Please refer to Example L.5 for the specific 50 tuples in the set $\Gamma_{\bar{S}}(Q, D_{\bar{N}^{(i)}}(Q))$ that generate the tuple t_Q^* in the answer to the query on the database.)* □

L.10 For the Q and Q' such that $Q \equiv_C Q'$, When Does Q' Have the Wave of Q?

In Section L.9 we learned how to construct, for CCQ queries Q and Q'' as specified in Section L.2.1, a function $\mathcal{F}_{(Q)}^{(Q'')}$. For each $i \in \mathbb{N}_+$, the function $\mathcal{F}_{(Q)}^{(Q'')}$ returns the multiplicity of the tuple t_Q^* in the bag $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$. The main result of this current section, Proposition L.47, shows that, whenever

(a) $Q \equiv_C Q'$ for CCQ queries Q and Q', and

(b) Q is an explicit-wave CCQ query (as specified by Definition 4.1),

then there exists a (nonempty) monomial class $\mathcal{C}_*^{(Q')}$ for the query Q' and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that the multiplicity monomial of $\mathcal{C}_*^{(Q')}$ is "the wave of the query Q" (as specified in Definition L.10). We show the result of Proposition L.47 using the properties of the functions $\mathcal{F}_{(Q)}^{(Q)}$ and $\mathcal{F}_{(Q)}^{(Q')}$. The proof of Theorem 4.1 is immediate from Proposition L.47 and from Propositions L.33 and L.34 of Section L.7.

As shown in Example L.5, hen the above condition (b) (of Q being an explicit-wave query) is not satisfied, then such a monomial class $\mathcal{C}_*^{(Q')}$ does not have to exist, and hence a SCVM from the query Q' to the query Q does not have to exist (as demonstrated by Example L.5 even in case $Q \equiv_C Q'$.

We begin the exposition by stating a useful auxiliary result in Section L.10.1.

L.10.1 *Equivalence of Multivariate Polynomials*

PROPOSITION L.46. *For a positive integer n, let X_1, X_2, $\ldots$, X_n be n distinct variables, where each variable accepts values from (at least) an infinite-cardinality subset of the set $\mathbb{Z}$ of all integers.*[24] *Let each of $\mathcal{P}_1$ and $\mathcal{P}_2$ be a finite-degree multivariate polynomial in terms of the variables X_1, $\ldots$, X_n and with integer coefficients. Further, assume that (w.l.o.g.) $\mathcal{P}_1 \not\equiv 0$. Then $\mathcal{P}_1 - \mathcal{P}_2 \equiv 0$ if and only if for each term $\Pi_{i=1}^{n} X_i^{l_i}$, where $l_i \in \{0\} \bigcup \mathbb{N}_+$ for all*[25] *$i \in \{1, \ldots, n\}$, the term has the same integer coefficient in $\mathcal{P}_1$ and $\mathcal{P}_2$.* $\square$

PROOF. *If:* Immediate from the definitions.

Only-If: The proof is by contradiction: Assume that for the finite-degree multivariate polynomial $\mathcal{P}_1 - \mathcal{P}_2$, call it $\mathcal{P}$, we have that $\mathcal{P} \equiv 0$. Assume further that there exists a term, call it $\mathcal{T}$, of the form $\Pi_{i=1}^{n} X_i^{l_i}$, such that the polynomial $\mathcal{P}$ has a nonzero integer coefficient for $\mathcal{T}$. We will show that in this case, $\mathcal{P} \equiv 0$ cannot hold, hence we arrive at a contradiction with the assumption $\mathcal{P} \equiv 0$.

Case 1: $\mathcal{T}$ is the only term with nonzero coefficient in the polynomial $\mathcal{P}$, and $l_i = 0$ for all $i \in \{1, \ldots, n\}$ in $\mathcal{T}$. Then $\mathcal{P}$ is equivalent to a nonzero-valued constant function, and the contradiction with the assumption $\mathcal{P} \equiv 0$ is immediate; Q.E.D.

Case 2: There exists a nonzero-coefficient term in $\mathcal{P}$, call this term $\mathcal{T}'$, such that there exists a $j \in \{1, \ldots, n\}$, where the power l_j of variable X_j in $\mathcal{T}'$ is a positive integer. Then for each X_l such that $l \in \{1, \ldots, n\} - \{j\}$, fix one arbitrary integer value $x_l \neq 0$ in the domain of X_l. (Clearly, it is possible to find a nonzero integer domain value for each X_l.) The result of substituting all the values x_l, $l \in \{1, \ldots, n\} - \{j\}$, into the polynomial $\mathcal{P}$ is a finite-degree univariate polynomial with integer coefficients, call it $\mathcal{P}_{(X_j)}$, in terms of the variable X_j and with at least one term with a nonzero (integer) coefficient. (One term with a nonzero coefficient in $\mathcal{P}_{(X_j)}$ results from $\mathcal{T}'$.) By our assumption that $\mathcal{P} \equiv 0$, the value of $\mathcal{P}_{(X_j)}$ equals zero on the entire infinite integer-valued domain of the variable X_j. This is impossible, hence we have arrived at a contradiction with the assumption that $\mathcal{P} \equiv 0$; Q.E.D. $\square$

L.10.2 *Query Q' Has the Wave of Q*

We now state and prove the main result of Section L.10.

PROPOSITION L.47. *Let Q and Q' be two CCQ queries, such that*

(a) we have that $Q \equiv_C Q'$, and

(b) Q is an explicit-wave CCQ query.

Then for the query Q' and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, there exists a nonempty monomial class $\mathcal{C}_*^{(Q')}$, such that the multiplicity monomial of $\mathcal{C}_*^{(Q')}$ is the wave of the query Q. $\square$

The proof of Proposition L.47, to be given in Section L.10.8, hinges on several results, which we now proceed to introduce.

[24]For different variables X_i, X_j, $i \neq j$, in the set $\{X_1, \ldots, X_n\}$, the domains of X_i and of X_j may include nonidentical (infinite-cardinality) subsets of the set $\mathbb{Z}$.

[25]When $l_i = 0$ for all $i \in \{1, \ldots, n\}$ in the term $\Pi_{i=1}^{n} X_i^{l_i}$, we set $\Pi_{i=1}^{n} X_i^{l_i}$ to the constant 1.

For the entire exposition, please keep in mind that throughout the proof of Theorem 4.1, all monomial classes of all queries, as well as each of the functions $\mathcal{F}_{(Q)}^{(Q)}$ and $\mathcal{F}_{(Q)}^{(Q')}$, are defined w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ for the fixed input query Q.

L.10.3 *Multivariate polynomials on total orders*

Recall that in general, for CCQ query Q'' and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ (for CCQ query Q), the function $\mathcal{F}_{(Q)}^{(Q'')}$ for Q'' and for $\{D_{\bar{N}^{(i)}}(Q)\}$ is not a multivariate polynomial on its entire domain $\mathcal{N}$. (See Example L.6 for an illustration.) At the same time, it turns out that the set $\mathcal{N}$ can be represented as a union of infinite-cardinality sets, such that for each set S in the union, the function $\mathcal{F}_{(Q)}^{(Q'')}$, for all the elements of the set S, can be rewritten equivalently as a multivariate polynomial in terms of the elements of the vector $\bar{N}$ and with integer coefficients. (That is, for each $\bar{N}^{(i)} \in S$, the value of $\mathcal{F}_{(Q)}^{(Q'')}(\bar{N}^{(i)})$ can be obtained by substituting the values in $\bar{N}^{(i)}$ into the relevant multivariate polynomial in terms of the elements of the vector $\bar{N}$ and with integer coefficients.)

In fact, as we know already for the case $r \leq 1$ (that is, for the input CCQ query Q that has $r = |M_{copy}| \leq 1$), the function $\mathcal{F}_{(Q)}^{(Q'')}$ for this case is a multivariate polynomial in terms of the elements of the vector $\bar{N}$ and with integer coefficients, on the entire domain $\mathcal{N}$ of the function. (See Section L.9.2.) Hence we proceed to prove the above claim for the case $r \geq 2$. Recall from Proposition L.4(iv) that for all $r > 0$ we have that $w > 0$. We conclude that whenever $r \geq 2$, the vector $\bar{N}$ has at least one element in the sequence N_{m+1} N_{m+2} $\ldots$ N_{m+w}. Of these cases, we first consider the special case $w = 1$, and then the general case $w \geq 1$.

The special case: $r \geq 2$ and $w = 1$.

We first consider all those cases (for the input CCQ query Q) where $r \geq 2$ and $w = 1$. In all such cases, the copy signatures of all relevant monomial classes (for both Q and Q'') are composed, by their definition, of the elements of the set $\{1, N_{m+1}\}$. Clearly, then, each *min* expression of Proposition L.44 in terms of elements of all the relevant copy signatures (in each case where $w = 1$) evaluates to either 1 or N_{m+1}, independently of the value $N_{m+1}^{(i)}$ of N_{m+1} in each vector $\bar{N}^{(i)} \in \mathcal{N}$. (Recall that $1 \leq N_{m+1}^{(i)}$ holds for all vectors $\bar{N}^{(i)}$, by definition of the set $\mathcal{N}$.) Using the results of Section L.9.5, we conclude that in all cases of query Q for which $r \geq 2$ and $w = 1$, the function $\mathcal{F}_{(Q)}^{(Q'')}$ for each such case is a multivariate polynomial in terms of the elements of the vector $\bar{N}$ and with integer coefficients, on the entire domain $\mathcal{N}$ of the function.

The general case: $r \geq 2$ (and $w \geq 1$).

Now consider all those cases (for the input CCQ query Q) where $r \geq 2$, and therefore, by Proposition L.4(iv), $w \geq 1$. We will define the sets S suggested above (such that $\mathcal{N}$ is a union of such infinite-cardinality sets) using total orders on the elements of the vector $\bar{N}_w = [\ 1\ \ N_{m+1}\ \ N_{m+2}\ \ \ldots\ \ N_{m+w}\ ]$. By $w \geq 1$, we have that the vector $\bar{N}_w$ has at least two elements. (Note: We will see that in the above special case of $r \geq 2$ and $w = 1$, the set $\mathcal{N}$ is a union of only one such set S.)

Let vector $\bar{K}_w = [\ 1\ \ K_1\ \ K_2\ \ldots\ K_w\ ]$ be an arbitrary fixed permutation of the vector $\bar{N}_w$ that satisfies the condition that the first element of $\bar{K}_w$ is always the constant 1. (That is, in each vector $\bar{K}_w$ we have that the sequence $K_1\ \ K_2\ \ldots\ K_w$ is a permutation of the sequence $N_{m+1}\ \ N_{m+2}\ \ldots\ N_{m+w}$ in the vector $\bar{N}_w$.) We refer to each such vector $\bar{K}_w$ as a *copy-variable-ordering vector for the vector* $\bar{N}$.

Let a total order $\mathcal{O}$ on the set $\{1,\ N_{m+1},\ N_{m+2},\ \ldots,\ N_{m+w}\}$ be defined as the reflexive transitive closure on the relation $\{\ (1, K_1),\ (K_1, K_2),\ (K_2, K_3),\ \ldots,\ (K_j, K_{j+1}),\ \ldots,\ (K_{w-1}, K_w)\ \}$, using the fixed vector $\bar{K}_w$. (We interpret the pair $(1, K_1)$ in $\mathcal{O}$ as $1 \leq K_1$. Further, whenever $w \geq 2$, for $1 \leq j \leq w-1$, we interpret the pair (K_j, K_{j+1}) in $\mathcal{O}$ as $K_j \leq K_{j+1}$. That is, $\mathcal{O}$ is the $\leq$ relation.) Then we say that the vector $\bar{K}_w$ *determines the total-order relation* $\mathcal{O}$ *on* $\bar{N}_w$, and use the notation $\mathcal{O}^{(\bar{K}_w)}$ for that total-order relation $\mathcal{O}$.

Now for a vector $\bar{K}_w$ as above and for an arbitrary vector $\bar{N}^{(i)} \in \mathcal{N}$, we define *the interpretation of each element of* $\bar{K}_w$ *w.r.t.* $\bar{N}^{(i)}$, as follows:

(1) We define the interpretation of the first element of $\bar{K}_w$ (that is, of the constant 1) to be the constant 1; and

(2) For each $j \in \{2, \ldots, w+1\}$, let the jth element of the vector $\bar{K}_w$ (that is, K_{j-1} in our notation for the vector $\bar{K}_w$) be a variable N_k of $\bar{N}$, for some $k \in \{m+1, \ldots, m+w\}$. Then the interpretation of K_{j-1} w.r.t. $\bar{N}^{(i)}$ is the value $N_k^{(i)}$ in $\bar{N}^{(i)}$ of the variable N_k in $\bar{N}$. We denote this interpretation of K_{j-1} w.r.t. $\bar{N}^{(i)}$ as $K_{j-1}^{(i)}$.

EXAMPLE L.7. *For $m = 1$ and for $w = 3$, the vector $\bar{N}_w$ is $\bar{N}_w = [\ 1\ \ N_2\ \ N_3\ \ N_4\ ]$. Let $\bar{K}_w := [\ 1\ \ N_3\ \ N_4\ \ N_2\ ]$. Let $\bar{N}^{(i)}$ be $[\ 5\ \ 3\ \ 7\ \ 6\ ]$. Then the interpretation of the elements of the vector $\bar{K}_w$ w.r.t. the vector $\bar{N}^{(i)}$ is as follows: 1 in $\bar{K}_w$ is interpreted as 1, the element $K_1 = N_3$ in $\bar{K}_w$ is interpreted as $K_1^{(i)} = 7$, the element $K_2 = N_4$ in $\bar{K}_w$ is interpreted as $K_2^{(i)} = 6$, and, finally, the element $K_3 = N_2$ in $\bar{K}_w$ is interpreted as $K_3^{(i)} = 3$.* □

Now suppose that we are given a vector $\bar{K}_w$ as defined above, and are given a vector $\bar{N}^{(i)} \in \mathcal{N}$. Then we say that the vector $\bar{N}^{(i)}$ *agrees with the total order* $\mathcal{O}^{(\bar{K}_w)}$ if and only if the interpretation of the elements of the vector $\bar{K}_w$ w.r.t. the vector $\bar{N}^{(i)}$ results in all (i.e., in *only*) true inequalities, on the set of natural numbers, in the reflexive transitive closure of the relation $\{\ (1, K_1^{(i)}),\ (K_1^{(i)}, K_2^{(i)}),\ (K_2^{(i)}, K_3^{(i)}),\ \ldots,\ (K_j^{(i)}, K_{j+1}^{(i)}),\ \ldots,\ (K_{w-1}^{(i)}, K_w^{(i)})\ \}$. (We interpret the pair $(1, K_1^{(i)})$ as $1 \leq K_1^{(i)}$. Further, in case $w \geq 2$, for each $j \in \{1, \ldots, w-1\}$, the pair $(K_j^{(i)}, K_{j+1}^{(i)})$ in this relation is interpreted as $K_j^{(i)} \leq K_{j+1}^{(i)}$.) The latter relation is obtained by replacing each K_j with $K_j^{(i)}$, for $j \in \{1, \ldots, w\}$, in the relation that defines the total order $\mathcal{O}^{(\bar{K}_w)}$, that is in the relation $\{\ (1, K_1),\ (K_1, K_2),\ (K_2, K_3),\ \ldots,\ (K_{w-1}, K_w)\ \}$.

EXAMPLE L.8. *For the vectors $\bar{K}_w$ and $\bar{N}^{(i)}$ of Example L.7, we have that the reflexive transitive closure of the relation $\{\ (1, K_1^{(i)}),\ (K_1^{(i)}, K_2^{(i)}),\ (K_2^{(i)}, K_3^{(i)})\ \}$, that is of the relation $\{\ (1,7),\ (7,6),\ (6,3)\ \}$, has elements violating true inequalities on natural numbers. For instance, one of the violations comes from the pair $(K_1^{(i)}, K_2^{(i)})$ in this relation, that is from the pair $(7,6)$. (Recall that we interpret $(K_j^{(i)}, K_{j+1}^{(i)})$ as $K_j^{(i)} \leq K_{j+1}^{(i)}$.) We conclude that the vector $\bar{N}^{(i)}$ does not agree with the total order $\mathcal{O}^{(\bar{K}_w)}$.*

Now consider a different vector $\bar{K}'_w := [\ 1\ \ N_2\ \ N_4\ \ N_3\ ]$. The interpretation of the elements of the vector $\bar{K}'_w$ w.r.t. the vector $\bar{N}^{(i)}$ (which is the same as before) is as follows: 1 in $\bar{K}'_w$ is interpreted as 1, the element $K'_1 = N_2$ in $\bar{K}'_w$ is interpreted as $K_1^{'\,(i)} = 3$, the element $K'_2 = N_4$ in $\bar{K}'_w$ is interpreted as $K_2^{'\,(i)} = 6$, and, finally, the element $K'_3 = N_3$ in $\bar{K}'_w$ is interpreted as $K_3^{'\,(i)} = 7$. Then we have that the reflexive transitive closure of the relation $\{\ (1, K_1^{'\,(i)}),\ (K_1^{'\,(i)}, K_2^{'\,(i)}),\ (K_2^{'\,(i)}, K_3^{'\,(i)})\ \}$, that is of the relation $\{\ (1,3),\ (3,6),\ (6,7)\ \}$, does not *have elements violating true inequalities on natural numbers. Thus, we conclude that the vector $\bar{N}^{(i)}$ agrees with the total order $\mathcal{O}^{(\bar{K}'_w)}$.* □

We now define the sets S as suggested in the beginning of this subsection. For a CCQ query Q for which $r \geq 2$ (and thus $w \geq 1$), and for the vector $\bar{N}$ for the query Q (as defined in Section L.3.1), let $\bar{K}_w$ be an arbitrary copy-variable-ordering vector for the vector $\bar{N}$. Then we define a subset $\mathcal{N}^{(\bar{K}_w)}$ of the set $\mathcal{N}$ as

$\mathcal{N}^{(\bar{K}_w)} = \{\bar{N}^{(i)} \in \mathcal{N} \mid \bar{N}^{(i)}$ agrees with the total order $\mathcal{O}^{(\bar{K}_w)}\}$.

(Note that in the case $w = 1$, there is only one possible vector $\bar{K}_w = [\ 1\ \ N_{m+1}\ ]$. Therefore, it is not hard to see that in the case $w = 1$, we have that the only possible set $\mathcal{N}^{(\bar{K}_w)}$ coincides with the entire set $\mathcal{N}$.)

The following result captures straightforward observations about the sets $\mathcal{N}^{(\bar{K}_w)}$.

PROPOSITION L.48. *Given a CCQ query Q for which $r \geq 2$, with vector $\bar{N}$ for the query Q constructed as defined in Section L.3.1. Then we have that:*

- *For each element $\bar{N}^{(i)}$ of the set $\mathcal{N}$, there exists at least one copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, such that $\bar{N}^{(i)}$ belongs to the set $\mathcal{N}^{(\bar{K}_w)}$;*
- *For each copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, the set $\mathcal{N}^{(\bar{K}_w)}$ is an infinite-cardinality subset of the set $\mathcal{N}$.*

□

We conclude from Proposition L.48 that the set $\mathcal{N}$ is a union of the infinite-cardinality sets $\mathcal{N}^{(\bar{K}_w)}$.

PROPOSITION L.49. *Given a CCQ query Q for which $r \geq 2$, with vector $\bar{N}$ for the query Q constructed as defined in Section L.3.1; and given a CCQ query Q'' satisfying the restrictions of Section L.2.1 w.r.t. the query Q. Then for each copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, there exists a multivariate polynomial $f_{(Q)}^{(Q'')}[\bar{K}_w]$ in terms of the elements of the vector $\bar{N}$ and with integer coefficients, such that:*

- *The polynomial $f_{(Q)}^{(Q'')}[\bar{K}_w]$ is defined on (at least) the set $\mathcal{N}^{(\bar{K}_w)} \subseteq \mathcal{N}$; and*
- *For each element $\bar{N}^{(i)}$ of the set $\mathcal{N}^{(\bar{K}_w)}$, we have that $f_{(Q)}^{(Q'')}[\bar{K}_w](\bar{N}^{(i)}) = \mathcal{F}_{(Q)}^{(Q'')}(\bar{N}^{(i)})$.*

□

The proof of Proposition L.49 is constructive and generalizes the intuition that we gained by considering (earlier in this subsection) the special case $r = 2$ and $w = 1$. Indeed, for each copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, each *min* expression of Proposition L.44 in terms of elements of all the relevant copy signatures evaluates to one fixed argument among all its arguments, regardless of the identity of specific elements $\bar{N}^{(i)}$ of the set $\mathcal{N}^{(\bar{K}_w)}$. Hence the result of replacing all the *min* expressions in $\mathcal{F}^{(Q'')}_{(Q)}$ with these fixed elements of the set $\{1, N_{m+1}, \ldots, N_{m+w}\}$ is the required function $f^{(Q'')}_{(Q)}[\bar{K}_w]$ for the subset $\mathcal{N}^{(\bar{K}_w)}$ of the domain $\mathcal{N}$ of the function $\mathcal{F}^{(Q'')}_{(Q)}$. Example L.6 provides an illustration.

For each vector $\bar{K}_w$ in case $r \geq 2$, in the remainder of this proof we will refer to the polynomial $f^{(Q'')}_{(Q)}[\bar{K}_w]$ as $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$; same for the respective functions for the queries Q and Q'. (See Proposition L.49 for a justification.) Further, for uniformity of notation, in the case where $r \leq 1$ we will refer to the function $\mathcal{F}^{(Q'')}_{(Q)}$ (which we showed in Section L.9.2 to be a multivariate polynomial in terms of the elements of the vector $\bar{N}$ and with integer coefficients, on the entire domain $\mathcal{N}$ of the function) as $\mathcal{F}^{(Q'')}_{(Q)}[\mathcal{N}]$. In the latter case (of $r \leq 1$), the only subdomain $\mathcal{N}^{(\bar{K}_w)}$ of the domain $\mathcal{N}$ is the set $\mathcal{N}$; hence we can use the notations $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ and $\mathcal{F}^{(Q'')}_{(Q)}[\mathcal{N}]$ interchangeably when $r \leq 1$. (Formally, in case $r = 0$, we define the vector $\bar{N}_w$ as $\bar{N}_w = [\ 1\ ]$, and in case $r = 1$, we define $\bar{N}_w$ as $\bar{N}_w = [\ 1 \quad N_{m+1}\ ]$. In either case, there exists exactly one permutation $\bar{K}_w$ of the vector $\bar{N}_w$, such that the first element of the vector $\bar{K}_w$ is the constant 1. Thus, the domain $\mathcal{N}^{(\bar{K}_w)}$ does indeed coincide with the domain $\mathcal{N}$ in all cases where $r \leq 1$.)

L.10.4 Query equivalence implies identical multivariate polynomials

We now show that for two CCQ queries Q and Q' such that $Q \equiv_C Q'$, the functions $\mathcal{F}^{(Q)}_{(Q)}$ and $\mathcal{F}^{(Q')}_{(Q)}$ can be expressed, on each well-defined (as in Sections L.9.2 and L.10.3) infinite-cardinality subdomain of the set $\mathcal{N}$, as *identical* multivariate polynomials in terms of the elements of the vector $\bar{N}$ and with integer coefficients. This result, Proposition L.50, is immediate from the results of Section L.9.5 and from Propositions L.46 and L.49.

PROPOSITION L.50. *Given a CCQ query Q, with vector $\bar{N}$ for the query Q constructed as defined in Section L.3.1; and given a CCQ query Q' such that $Q \equiv_C Q'$ holds. Then for each copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, we have that the multivariate polynomials $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}_w]$ and $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$, in terms of the elements of the vector $\bar{N}$ and with integer coefficients, are identical functions on the domain $\mathcal{N}^{(\bar{K}_w)}$.* □

L.10.5 Solid terms and phantom terms of the polynomials for the multiplicity functions

To proceed, we need to introduce technical denotations for the terms of the multivariate polynomials that we have been considering. Suppose that for CCQ queries Q and Q'', we are given the multivariate polynomial $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ for an arbitrary $\bar{K}_w$, as defined earlier in Section L.10. Consider a monomial (excluding the nonzero integer coefficient of the term) $\mathcal{T}$ in $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$. We say that:

- $\mathcal{T}$ is a *solid term of* $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ if there exists a nonempty monomial class, $\mathcal{C}^{(Q'')}$, for the query Q'' and for the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such that $\mathcal{T}$ is the multiplicity monomial for the class $\mathcal{C}^{(Q'')}$.
- Conversely, we say that $\mathcal{T}$ is a *phantom term of* $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ whenever $\mathcal{T}$ is not a solid term of $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$.

For instance, in Example L.6, the multivariate polynomial $\mathcal{F}^{(Q)}_{(Q)}[[\ 1\ N_3\ N_4\ ]]$, that is the polynomial $N_1 \times N_2 \times (N_4)^2$, has one solid term, $\mathcal{T} = N_1 \times N_2 \times (N_4)^2$. The reason for the term $\mathcal{T}$ being a solid term of the polynomial $\mathcal{F}^{(Q)}_{(Q)}[[\ 1\ N_3\ N_4\ ]]$ is that the query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ has a monomial class $\mathcal{C}^{(Q)}_4$ whose multiplicity monomial is exactly the term $\mathcal{T}$. (See Example L.5 for the details on the monomial class $\mathcal{C}^{(Q)}_4$.)

Now consider an abstract example of a phantom term.

EXAMPLE L.9. *Consider the case where $m = 0$ and $r = w = 2$. Suppose that for these values of m, r, and w, for some hypothetical CCQ query Q'' and for some hypothetical family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, it holds that the set of all monomial classes in this setting consists of two monomial classes, say $\mathcal{C}^{(Q'')}_1$ and $\mathcal{C}^{(Q'')}_2$. By $m = 0$, we have that the noncopy signature of each of $\mathcal{C}^{(Q'')}_1$ and $\mathcal{C}^{(Q'')}_2$ is the empty vector. Suppose that the copy signature of $\mathcal{C}^{(Q'')}_1$ is $[\ N_1\ N_2\ ]$, and that the copy signature of $\mathcal{C}^{(Q'')}_2$ is $[\ N_2\ N_1\ ]$.*

The noncopy signatures of the two monomial classes are identical. Thus, by our results of Section L.9.5, the function for the multiplicity of the tuple t^*_Q, in the answer to the query Q'' on the databases $\{D_{\bar{N}^{(i)}}(Q)\}$, will be computed as the cardinality of the union of the sets for the copy signatures of the two monomial classes. That is, the multiplicity function will be

$$2 \times N_1 \times N_2 - (min(N_1, N_2))^2\ .$$

Then for the vector $\bar{K}_w = [\ 1\ N_1\ N_2\ ]$, the multivariate polynomial for this multiplicity function on the domain $\mathcal{N}^{(\bar{K}_w)}$ will be

$$2 \times N_1 \times N_2 - (N_1)^2\ .$$

In this polynomial, (i) the term $N_1 \times N_2$ is a solid term of the polynomial, because the monomial class $\mathcal{C}^{(Q'')}_1$ (as well as $\mathcal{C}^{(Q'')}_2$) has the term $N_1 \times N_2$ as its multiplicity monomial. In contrast, (ii) the term $(N_1)^2$ is by definition a phantom term of the polynomial. □

L.10.6 The multiplicity function for the query Q

We now make several observations concerning the solid and phantom terms in the polynomials $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}_w]$ for the function $\mathcal{F}^{(Q)}_{(Q)}$ of the query Q. First, as usual, we will need some terminology. For CCQ queries Q and Q'' (where Q'', as usual, may or may not be the query Q) and for the vector $\bar{N}$ constructed for the query Q as defined in Section L.3.1, fix an arbitrary copy-variable-ordering vector, $\bar{K}_w$, for $\bar{N}$. In case $m \geq 1$, consider all the terms in the function $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ such that each term has the product $N_1 \times N_2 \times \ldots \times N_m$. Call all these terms collectively "the m-covering part of the function $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$." For the case $m = 0$, we say that *all* the terms of the function $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ constitute the m-covering part of the function.

The observations of this subsection, Propositions L.51 and L.52, are made for the polynomials $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}_w]$, that is for the case where the query Q'' coincides with the query Q.

Proposition L.51. *Given an explicit-wave CCQ query Q, with vector $\bar{N}$ constructed as defined in Section L.3.1. Then we have that:*

- *For each copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, the m-covering part of the function $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}_w]$ has at least one term with a nonzero coefficient; and*
- *For each pair $(\bar{K}_w, \bar{K}'_w)$ of copy-variable-ordering vectors for the vector $\bar{N}$, the m-covering part of the function $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}_w]$ and the m-covering part of the function $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}'_w]$ are identical multivariate polynomials (in terms of the elements of the vector $\bar{N}$ and with integer coefficients).*

□

The observations of Proposition L.51 are immediate from the relevant definitions, from the construction of the function $\mathcal{F}^{(Q)}_{(Q)}$, and from the definition of explicit-wave query. Specifically, the second bullet of Proposition L.51 follows from the fact that among all the monomial classes for the query Q whose (classes') noncopy signature is a permutation of the vector $[\ N_1\ N_2\ \ldots\ N_m\ ]$ in case $m \geq 1$, or is the empty vector in case $m = 0$, for each pair of such monomial classes with the same noncopy signature, there is *unconditional* dominance between the two classes. This fact holds by the definition of explicit-wave query, Definition L.10. We then use the results of Section L.9.2 (specifically of Proposition L.42) to establish that the respective m-covering polynomials do not depend on the vector $\bar{K}_w$, because the relevant sets $\mathbb{C}^{nondom}$ do not depend on the vector $\bar{K}_w$.

The results of Proposition L.51 permit us to talk about "the m-covering part of the function $\mathcal{F}^{(Q)}_{(Q)}$," for every explicit-wave CCQ query Q. We denote by $\mathcal{G}^{(Q)}_{(Q)}$ this nonempty multivariate polynomial in terms of the elements of the vector $\bar{N}$ and with integer coefficients. Notice that the reference to copy-variable-ordering vectors has been dropped from the notation for $\mathcal{G}^{(Q)}_{(Q)}$.

The next observations, in Proposition L.52, are about the properties of the function $\mathcal{G}^{(Q)}_{(Q)}$ for explicit-wave queries Q. The results of Proposition L.52 hold by the relevant definitions and by Proposition L.33.

Proposition L.52. *Given an explicit-wave CCQ query Q, we have that:*

- *In the function $\mathcal{G}^{(Q)}_{(Q)}$, each term is a solid term;*
- *Each term of the function $\mathcal{G}^{(Q)}_{(Q)}$ has a positive coefficient; and*
- *The multivariate polynomial $\mathcal{G}^{(Q)}_{(Q)}$ has a (solid) term which is the wave $\mathcal{P}^{(Q)}_*$ of the query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$.*

□

L.10.7 The m-covering part of the polynomials $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$, in case where $Q \equiv_C Q'$

In this subsection we show that for CCQ queries Q and Q' such that Q is an explicit-wave query and such that $Q \equiv_C Q'$, it holds that all the m-covering parts of all the polynomials $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ have, *as a solid term,* the wave $\mathcal{P}^{(Q)}_*$ of the query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$.

The first observation that we make follows trivially from Propositions L.50 and L.52.

Proposition L.53. *Given an explicit-wave CCQ query Q and a CCQ query Q' such that $Q \equiv_C Q'$. Then for each copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, the m-covering part of the function $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ has, with a positive-integer coefficient, the wave $\mathcal{P}^{(Q)}_*$ of the query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$.* □

The only difference between the formulations of Proposition L.53 and of Proposition L.54 is that Proposition L.54 claims that the term $\mathcal{P}^{(Q)}_*$ is a *solid* term in the m-covering part of the function $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$, for each vector $\bar{K}_w$.

Proposition L.54. *Given an explicit-wave CCQ query Q and a CCQ query Q' such that $Q \equiv_C Q'$. Then for each copy-variable-ordering vector, $\bar{K}_w$, for the vector $\bar{N}$, the m-covering part of the function $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ has, as a solid term, the wave $\mathcal{P}^{(Q)}_*$ of the query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$.* □

In the proof of Proposition L.54, for the cases where $r \geq 2$ we will be keeping track of all the occurrences of all the variables N_{m+1} through N_{m+w} in the m-covering parts of the multivariate polynomials $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ and $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}_w]$, for various vectors $\bar{K}_w$. For each possible vector $\bar{K}_w$ (for the vector $\bar{N}$ of Q) and for the m-covering part of the polynomial $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$, where Q'' is one of Q and Q', we set up a set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ defined constructively as follows:

(I) In case $m \geq 1$, drop the product $\Pi^{m}_{j=1} N_j$ from all terms of the m-covering part of the polynomial $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$. Call the resulting function $f^{(m,Q'')}[\bar{K}_w]$. In case $m = 0$, denote by $f^{(m,Q'')}[\bar{K}_w]$ the (original) m-covering part of the polynomial $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$. Observe that for all $m \geq 0$, the function $f^{(m,Q'')}[\bar{K}_w]$ is a multivariate polynomial, with integer coefficients, in terms of the variables $N_{m+1}, \ldots, N_{m+w}$ *only.*

(II) To the output $f^{(m,Q'')}[\bar{K}_w]$ of Step (I), apply the algorithm $\mathcal{B}$-construction, to obtain the set $\mathcal{B}(f^{(m,Q'')}$ $[\bar{K}_w])$. Then return the set $\mathcal{B}^{(Q'')}[\bar{K}_w] := \mathcal{B}(f^{(m,Q'')}$ $[\bar{K}_w])$.

We now specify the algorithm $\mathcal{B}$-construction.

Algorithm $\mathcal{B}$-construction.

Input: Multivariate polynomial $\mathcal{P}$ in terms of variables $N_{m+1}, \ldots, N_{m+w}$, where $w \geq 1$, with integer coefficients.

Output: Set $\mathcal{B}(\mathcal{P})$.

(1) Rewrite equivalently (syntactically) the polynomial $\mathcal{P}$, as follows:

   (a) First "expand" each power ≥ 2 in the polynomial. That is, for each expression in $\mathcal{P}$ of the form A^b, where $A \in \{N_{m+1}, \ldots, N_{m+w}\}$ and $b \geq 2$, replace the A^b by the product $\Pi_{j=1}^{b} A$.

   (b) Then "expand" each nonunity integer coefficient in the resulting polynomial $\mathcal{P}'$. That is, consider each term $\mathcal{T}$ with a nonzero and nonunity (by absolute value) integer coefficient in $\mathcal{P}'$. That is, suppose $\mathcal{T} = C \times \mathcal{T}'$, with the integer coefficient $C \in \mathbb{Z} - \{-1, 0, 1\}$, and with $\mathcal{T}'$ a product of elements of the set $\{N_{m+1}, \ldots, N_{m+w}\}$, perhaps with multiple occurrences of some of these variables (by (a) above). Then, for each such term $\mathcal{T}$, (i) in case $C > 0$, replace $\mathcal{T}$ in $\mathcal{P}'$ with a sum of C copies of the term $\mathcal{T}'$; and (ii) in case $C < 0$, replace $\mathcal{T}$ in $\mathcal{P}'$ with a sum of C copies of the term $(-1) \times \mathcal{T}'$.

(2) Set $\mathcal{B}(\mathcal{P}) := \emptyset$. For each variable $N_j \in \{N_{m+1}, \ldots, N_{m+w}\}$, count the number $M_j^{(+)}$ of positive occurrences of N_j in the output of step (1), and count the number $M_j^{(-)}$ of negative occurrences of N_j in the output of step (1); then, whenever the difference $M_j^{(+)} - M_j^{(-)}$ is not zero, add to the set $\mathcal{B}(\mathcal{P})$ the element $(M_j^{(+)} - M_j^{(-)}) \times N_j$.

(3) Output the resulting set $\mathcal{B}(\mathcal{P})$.

We provide three illustrations of the construction of sets $\mathcal{B}^{(Q'')}[\bar{K}_w]$, in Examples L.10 through L.12.

EXAMPLE L.10. *Consider the construction of the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ in the context of Example L.9. In that example, the m-covering part of the function $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$, for the vector $\bar{K}_w = [\ 1\ N_1\ N_2\ ]$, is*

$$2 \times N_1 \times N_2 - (N_1)^2 \ .$$

*(Recall that in Example L.9, $m = 0$, hence each term of the polynomial $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ of Example L.9 is also a term of the m-covering part of $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$.) Thus, Step (I) of the construction algorithm renames the above function into $f^{(m,Q'')}[\bar{K}_w]$. By Step (1)(a) of algorithm $\mathcal{B}$-*CONSTRUCTION*, we rewrite the polynomial $f^{(m,Q'')}[\bar{K}_w]$ equivalently as*

$$2 \times N_1 \times N_2 - N_1 \times N_1 \ .$$

Then, by Step (1)(b) of the algorithm, we rewrite the above expression equivalently as

$$N_1 \times N_2 + N_1 \times N_2 - N_1 \times N_1 \ .$$

*The set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ for this function is $\mathcal{B}^{(Q'')}[\bar{K}_w] = \{\ 2 \times N_2\ \}$. The reason is, the expression that we obtained as the outcome of step (1)(b) of algorithm $\mathcal{B}$-*CONSTRUCTION* has two positive-sign occurrences of N_2, two positive-sign occurrences of N_1, and two negative-sign occurrences of N_1.* □

EXAMPLE L.11. *We give an illustration of the construction of sets $\mathcal{B}^{(Q'')}[\bar{K}_w]$, using the function $\mathcal{F}^{(Q)}_{(Q)}$ of Example L.6. In that example, $m = 2$ and $r = w = 2$. We note that the query Q of Example L.6 is not an explicit-wave query. As shown in that example, the polynomial $\mathcal{F}^{(Q)}_{(Q)}[[\ 1\ N_3\ N_4\ ]]$ is*

$$\mathcal{F}^{(Q)}_{(Q)}[[\ 1\ N_3\ N_4\ ]] = N_1 \times N_2 \times (N_4)^2 \ ,$$

and the polynomial $\mathcal{F}^{(Q)}_{(Q)}[[\ 1\ N_4\ N_3\ ]]$ is

$$\mathcal{F}^{(Q)}_{(Q)}[[\ 1\ N_4\ N_3\ ]] = N_1 \times N_2 \times (N_3)^2 \ .$$

Hence, the set $\mathcal{B}^{(Q)}[[\ 1\ N_3\ N_4\ ]]$ for the case of Example L.6 is $\mathcal{B}^{(Q)}[[\ 1\ N_3\ N_4\ ]] = \{\ 2 \times N_4\ \}$, and the set $\mathcal{B}^{(Q)}[[\ 1\ N_4\ N_3\ ]]$ is $\mathcal{B}^{(Q)}[[\ 1\ N_4\ N_3\ ]] = \{\ 2 \times N_3\ \}$. □

EXAMPLE L.12. *Consider an abstract example for the case where $m = 3$ and $r = w = 2$. Suppose that for these values of m, r, and w, for some hypothetical CCQ query Q'' and for some hypothetical family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, it holds that the set of all monomial classes in this setting consists of four monomial classes, $\mathcal{C}_1^{(Q'')}$ through $\mathcal{C}_4^{(Q'')}$. Suppose that the noncopy signatures and the copy signatures of the four classes are as follows:*

- *For the monomial class $\mathcal{C}_1^{(Q'')}$, $\Phi_n[\mathcal{C}_1^{(Q'')}] = [\ Y_1\ Y_2\ Y_3\ ]$, and $\Phi_c[\mathcal{C}_1^{(Q'')}] = [\ N_4\ N_5\ ]$.*
- *For the monomial class $\mathcal{C}_2^{(Q'')}$, $\Phi_n[\mathcal{C}_2^{(Q'')}] = [\ Y_1\ Y_2\ Y_3\ ]$, and $\Phi_c[\mathcal{C}_2^{(Q'')}] = [\ N_5\ N_4\ ]$.*
- *For the monomial class $\mathcal{C}_3^{(Q'')}$, $\Phi_n[\mathcal{C}_3^{(Q'')}] = [\ Y_2\ Y_3\ Y_1\ ]$, and $\Phi_c[\mathcal{C}_3^{(Q'')}] = [\ N_4\ N_4\ ]$.*
- *Finally, for the monomial class $\mathcal{C}_4^{(Q'')}$, $\Phi_n[\mathcal{C}_4^{(Q'')}] = [\ Y_3\ Y_1\ Y_2\ ]$, and $\Phi_c[\mathcal{C}_4^{(Q'')}] = [\ N_5\ N_5\ ]$.*

The two monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ have identical noncopy signatures $[\ Y_1\ Y_2\ Y_3\ ]$. The two other monomial classes, $\mathcal{C}_3^{(Q'')}$ and $\mathcal{C}_4^{(Q'')}$, each have a unique noncopy signature. Thus, by our results of Section L.9.5, the function for the multiplicity of the tuple t_Q^*, in the answer to the query Q'' on the databases $\{D_{\bar{N}^{(i)}}(Q)\}$, will be computed as the expression for the cardinality of the union of the sets for the copy signatures of the two monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$, which (expression) is multiplied by the product $\Pi_{j=1}^{3} N_j$ and then summed up with the multiplicity monomials for the other two monomial classes. By definition, the multiplicity monomial for the monomial class $\mathcal{C}_3^{(Q'')}$ is $(\Pi_{j=1}^{3} N_j) \times (N_4)^2$, and the multiplicity monomial for the monomial class $\mathcal{C}_4^{(Q'')}$ is $(\Pi_{j=1}^{3} N_j) \times (N_5)^2$. In turn, the cardinality of the union of the sets for the copy signatures of the two monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ is computed similarly to the function of Example L.9. Thus, the overall multiplicity function will be

$$(\Pi_{j=1}^{3} N_j) \times (2 \times N_4 \times N_5 - (min(N_4, N_5))^2 + (N_4)^2 + (N_5)^2) \ .$$

Then for the vector $\bar{K}_w = [\ 1\ N_4\ N_5\ ]$, the multivariate polynomial for this multiplicity function on the domain $\mathcal{N}^{(\bar{K}_w)}$ will be

$$(\Pi_{j=1}^{3} N_j) \times (2 \times N_4 \times N_5 - (N_4)^2 + (N_4)^2 + (N_5)^2) =$$

$$= (\Pi_{j=1}^{3} N_j) \times (2 \times N_4 \times N_5 + (N_5)^2)\ .$$

Similarly, for the vector $\bar{K}'_w = [\ 1\ N_5\ N_4\ ]$, the multivariate polynomial for this multiplicity function on the domain $\mathcal{N}^{(\bar{K}'_w)}$ will be

$$(\Pi_{j=1}^{3} N_j) \times (2 \times N_4 \times N_5 - (N_5)^2 + (N_4)^2 + (N_5)^2) =$$

$$= (\Pi_{j=1}^{3} N_j) \times (2 \times N_4 \times N_5 + (N_4)^2)\ .$$

Consider the construction of the sets $\mathcal{B}^{(Q'')}[\bar{K}_w]$ and $\mathcal{B}^{(Q'')}[\bar{K}'_w]$, for the above two vectors $\bar{K}_w = [\ 1\ \ N_4\ \ N_5\ ]$ and $\bar{K}'_w = [\ 1\ \ N_5\ \ N_4\ ]$. In the construction of the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$, the m-covering part of the function $\mathcal{F}^{(Q'')}_{(Q)}[\bar{K}_w]$ is, by definition,

$$(\Pi_{j=1}^{3} N_j) \times (2 \times N_4 \times N_5 + (N_5)^2)\ .$$

Thus, Step (I) of the construction algorithm obtains the polynomial

$$f^{(m,Q'')}[\bar{K}_w] = 2 \times N_4 \times N_5 + (N_5)^2\ ,$$

by dividing the entire m-covering polynomial by the product $\Pi_{j=1}^{3} N_j$. By Step (1)(a) of algorithm $\mathcal{B}$-CONSTRUCTION, we rewrite the polynomial $f^{(m,Q'')}[\bar{K}_w]$ equivalently as

$$2 \times N_4 \times N_5 + N_5 \times N_5\ .$$

Then, by Step (1)(b) of the algorithm, we rewrite the above expression equivalently as

$$N_4 \times N_5 + N_4 \times N_5 + N_5 \times N_5\ .$$

We conclude that the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ for this function is $\mathcal{B}^{(Q'')}[\bar{K}_w]$ $= \{\ 2 \times N_4, 4 \times N_5\ \}$.

By similar reasoning, for the vector $\bar{K}'_w$ we obtain that the set $\mathcal{B}^{(Q'')}[\bar{K}'_w]$ for the m-covering part of the function

$$(\Pi_{j=1}^{3} N_j) \times (2 \times N_4 \times N_5 + (N_4)^2)\ .$$

is $\mathcal{B}^{(Q'')}[\bar{K}'_w] = \{\ 4 \times N_4, 2 \times N_5\ \}$. □

The challenge in the proof of Proposition L.54 is that the Proposition is to be proved for arbitrary CCQ queries Q' such that $Q' \equiv_C Q$. For this reason, we have to assume that while for all $\bar{K}_w$ the m-covering part of the function $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ "looks like" the all-positive-coefficient polynomial $\mathcal{G}^{(Q)}_{(Q)}$ (see Propositions L.50 and L.52), it may still be that the m-covering part of the function $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ has some negative terms hidden in there but canceled out by some positive terms. That is, it may still be that the m-covering part of the function $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ has some expressions for the cardinality of the union of two or more sets, as in, e.g., Example L.9. If this is the case then the term $\mathcal{P}^{(Q)}_*$ (i.e., the wave of the query Q) in these polynomials could be a phantom term resulting from the application of the inclusion-exclusion principle to compute the cardinality of such set unions. (Example L.12 provides an abstract illustration of how negative terms of a polynomial can get canceled out by positive terms, for *all* vectors $\bar{K}_w$, to result for each $\bar{K}_w$ in a polynomial that does not have any negative-sign terms.)

We now make three observations, in Proposition L.55 and in Corollaries L.2 and L.3, concerning the properties of the sets $\mathcal{B}^{(Q'')}[\bar{K}_w]$ for functions $\mathcal{F}^{(Q'')}_{(Q)}$, where Q'' is one of Q and Q'.

PROPOSITION L.55. *Given two multivariate polynomials, $\mathcal{P}_1$ and $\mathcal{P}_2$, in terms of variables $N_{m+1}, \ldots, N_{m+w}$, for some $m \geq 0$ and $w \geq 1$, and with integer coefficients. Let each of $N_{m+1}, \ldots, N_{m+w}$ be defined on a domain that includes (at least) an infinite-cardinality subset of the set $\mathbb{Z}$ of all integers. Then $\mathcal{P}_1 \equiv \mathcal{P}_2$ implies that the outputs of algorithm $\mathcal{B}$-CONSTRUCTION on the inputs $\mathcal{P}_1$ and $\mathcal{P}_2$ are identical sets.* □

That is, Proposition L.55 says that the sets $\mathcal{B}\ (\mathcal{P}_1)$ and $\mathcal{B}\ (\mathcal{P}_2)$, in the notation of this subsection, are identical sets. The claim of Proposition L.55 is immediate from Proposition L.46 and from the construction of the algorithm $\mathcal{B}$-CONSTRUCTION.

COROLLARY L.2. *Given CCQ query Q, with vector $\bar{N}$ constructed as defined in Section L.3.1, and given CCQ query Q' such that $Q \equiv_C Q'$. Then for each copy-variable-ordering vector $\bar{K}_w$ for the vector $\bar{N}$, the sets $\mathcal{B}^{(Q)}[\bar{K}_w]$ and $\mathcal{B}^{(Q')}[\bar{K}_w]$ are identical sets.* □

The result of Corollary L.2 is immediate from Propositions L.55 and L.50.

The proof of our next result, Corollary L.3, is immediate from Propositions L.51 and L.55.

COROLLARY L.3. *Given an explicit-wave CCQ query Q, with vector $\bar{N}$ constructed as defined in Section L.3.1. Then for each pair $(\bar{K}_w, \bar{K}'_w)$ of copy-variable-ordering vectors for the vector $\bar{N}$, the sets $\mathcal{B}^{(Q)}[\bar{K}_w]$ and $\mathcal{B}^{(Q)}[\bar{K}'_w]$ are identical sets.* □

Note that the result of Corollary L.3 does *not* hold in case where Q is *not* an explicit-wave query. See Example L.11 for an illustration.

As a final step before proving Proposition L.54, we formulate a lemma that we will use in the proof of Proposition L.54.

LEMMA L.2. *Given $n \geq 1$ natural numbers $a_1, a_2, \ldots, a_n$ such that $a_1 \leq a_2 \leq \ldots \leq a_n$. For each $j \in \{1, \ldots, n\}$, let the set $\mathcal{A}_j$ be $\mathcal{A}_j := \{1,\ 2,\ \ldots,\ a_j - 1,\ a_j\}$. Then the cardinality of the set $\bigcup_{j=1}^{n} \mathcal{A}_j$ is the natural number a_n.* □

The claim of Lemma L.2 is trivial. (Observe that for all n we have that (a) $\mathcal{A}_j \subseteq \mathcal{A}_n$ for all $j \in \{1, \ldots, n\}$, and that (b) the cardinality of the set $\mathcal{A}_n$ is exactly a_n.) Still, we take care to prove this claim. The reason is, in the proof of Proposition L.54, it will be important to us that in the expression for the cardinality of the set $\bigcup_{j=1}^{n} \mathcal{A}_j$, where the expression is obtained by the inclusion-exclusion principle, "almost all" the terms in the expression cancel each other out, while leaving "uncanceled out" only the term a_n.

PROOF. The proof is by induction.

Basis: $n = 1$. The claim of Lemma L.2 holds trivially in this case.

Induction step: Let $n \geq 2$, and assume that the claim of Lemma L.2 holds for $n-1$. Examine the expression for the cardinality of the set $\bigcup_{j=1}^{n} \mathcal{A}_j$, where the expression is obtained by the inclusion-exclusion principle. In this expression, there are two groups of terms: The first group is the sum $\sum_{j=1}^{n} |\mathcal{A}_j|$, that is the sum $\sum_{j=1}^{n} a_j$. In the second group, each term is of the form $min(a_{j_1}, a_{j_2}, \ldots, a_{j_k})$ (with either a positive or negative sign), for some k such that $2 \leq k \leq n$ and where each $a_{j_l} \in \{a_1, \ldots, a_n\}$, for $l \in \{1, \ldots, k\}$.

Assume w.l.o.g. that in the term $min(a_{j_1}, a_{j_2}, \ldots, a_{j_k})$, it holds that $a_{j_1} \leq a_{j_2} \leq \ldots \leq a_{j_k}$. (From $a_1 \leq a_2 \leq \ldots \leq a_n$ in the statement of Lemma L.2, we have a total $\leq$ order on the set $\{a_1,\ a_2,\ \ldots,\ a_n\}$.) Thus, the term $min(a_{j_1}, a_{j_2}, \ldots, a_{j_k})$ can be replaced by the equivalent term a_{j_1}. We call this replacement rule our *first equivalent-replacement rule.*

Observe that from $a_1 \leq a_2 \leq \ldots \leq a_n$ it holds that when we use this equivalent-replacement rule, for all terms of the form $min(a_{j_1}, a_{j_2}, \ldots, a_{j_k})$, with $k \geq 2$, the result of this equivalent replacement is an element of the set $\{a_1, \ldots, a_{n-1}\}$. That is, a_n is the only element of the set $\{a_1, \ldots, a_n\}$ that by our replacement rule never (equivalently) replaces the term $min(a_{j_1}, a_{j_2}, \ldots, a_{j_k})$, even in those cases where a_n is one of the a_{j_1}, a_{j_2}, $\ldots$, a_{j_k}. Hence our *second equivalent-replacement rule* is to replace the term $min(a_{j_1}, a_{j_2}, \ldots, a_{j_k})$, such that (i) $k \geq 2$, (ii) $a_{j_1} \leq a_{j_2} \leq \ldots \leq a_{j_k}$, and (iii) $a_{j_k} = a_n$, with the equivalent *min*-expression $min(a_{j_1}, a_{j_2}, \ldots, a_{j_{k-1}})$. (In the special case where $k = 2$, this equivalent *min*-expression reduces to $min(a_{j_1})$ and hence can be replaced equivalently by a_{j_1} .)

We will use these two equivalent-replacement rules to rewrite *min*-terms in the expression for the cardinality of the set $\bigcup_{j=1}^{n} \mathcal{A}_j$ by the inclusion-exclusion principle. For instance, the expression for the cardinality of the set $\bigcup_{j=1}^{n} \mathcal{A}_j$, in case $n = 2$, can be rewritten equivalently, as follows.

(a) We use our first equivalent-replacement rule to obtain an equivalent rewriting of that expression, $a_1 + a_2 - a_1$. To obtain this expression, we replace the expression for the cardinality of the set $\mathcal{A}_1 \bigcap \mathcal{A}_2$, that is the expression $min(a_1, a_2)$, with a_1, using $a_1 \leq a_2$.

(b) We use our second equivalent-replacement rule to obtain an equivalent rewriting of that expression, $a_1 + a_2 - a_1$. (This is the same rewriting as in (a).) To arrive at this expression, we replace the expression for the cardinality of the set $\mathcal{A}_1 \bigcap \mathcal{A}_2$, that is the expression $min(a_1, a_2)$, with $min(a_1)$, using the facts (i) $k = 2 \geq 2$, (ii) $a_1 \leq a_2$, and (iii) $a_2 = a_n$. We then further rewrite $min(a_1)$ equivalently as just a_1 .

Now consider, for a general $n \geq 2$, the expression, call it $\mathcal{E}_n$, for the cardinality of the set $\bigcup_{j=1}^{n} \mathcal{A}_j$ as obtained by the inclusion-exclusion principle. First, consider all the terms in the expression $\mathcal{E}_n$ such that each term mentions a_n. Call this group of terms $\mathcal{E}'_n$, and refer to all the remaining terms in $\mathcal{E}_n$ collectively as $\mathcal{E}''_n$. We analyze the expressions $\mathcal{E}'_n$ and $\mathcal{E}''_n$.

(A) It is easy to see that the expression $\mathcal{E}''_n$ evaluates to a_{n-1} by our induction assumption. (This expression is exactly the expression, which we call $\mathcal{E}_{n-1}$, for the cardinality of the set $\bigcup_{j=1}^{n-1} \mathcal{A}_j$ as obtained by the inclusion-exclusion principle.)

(B) In the expression $\mathcal{E}'_n$ we set aside the term $|\mathcal{A}_n| = a_n$. All the remaining terms in $\mathcal{E}'_n$ are each of the form $min(a_{j_1}, a_{j_2}, \ldots, a_{j_k})$, with $k \geq 2$, and such that (assuming $a_{j_1} \leq a_{j_2} \leq \ldots \leq a_{j_k}$) it holds that $a_{j_k} = a_n$. We use our second equivalent-replacement rule to rewrite each such term in $\mathcal{E}'_n$ as $min(a_{j_1}, a_{j_2}, \ldots, a_{j_{k-1}})$. (For all cases $k = 2$ we further equivalently rewrite the resulting term $min(a_{j_1})$ as a_{j_1} .) It is easy to see that after all these equivalent replacements are done, the expression $\mathcal{E}'_n$ equals $a_n - \mathcal{E}_{n-1}$. (Again, $\mathcal{E}_{n-1}$ here denotes the expression for the cardinality of the set $\bigcup_{j=1}^{n-1} \mathcal{A}_j$ as obtained by the inclusion-exclusion principle.)

(C) Finally, we put together the outputs of steps (A) and (B), to obtain that the expression $\mathcal{E}_n = \mathcal{E}''_n + \mathcal{E}'_n$ evaluates to $(\mathcal{E}_{n-1}) + (a_n - \mathcal{E}_{n-1})$. (While this is immaterial here because the two expressions $\mathcal{E}_{n-1}$ cancel each other out in this expression for $\mathcal{E}_n$, by our induction assumption we know that $\mathcal{E}_{n-1}$ evaluates to a_{n-1} .) We conclude that the expression $\mathcal{E}_n$ can be rewritten equivalently as just a_n .

End of the induction step. Q.E.D. $\square$

We now provide an illustration of the use of Lemma L.2 in the proof of Proposition L.54.

EXAMPLE L.13. *Consider an abstract example for the case where $m = 1$ and $r = w = 3$. Suppose that for these values of m, r, and w, for some hypothetical CCQ query Q'' and for some hypothetical family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, it holds that the set of all monomial classes in this setting consists of three monomial classes, $\mathcal{C}_1^{(Q'')}$ through $\mathcal{C}_3^{(Q'')}$. Suppose that the noncopy signatures and the copy signatures of the three classes are as follows.*

- *For the monomial class $\mathcal{C}_1^{(Q'')}$, $\Phi_n[\mathcal{C}_1^{(Q'')}] = [\ Y_1\ ]$ and $\Phi_c[\mathcal{C}_1^{(Q'')}] = [\ N_2\ 1\ N_3\ ]$.*
- *For the monomial class $\mathcal{C}_2^{(Q'')}$, $\Phi_n[\mathcal{C}_2^{(Q'')}] = [\ Y_1\ ]$ and $\Phi_c[\mathcal{C}_2^{(Q'')}] = [\ N_4\ 1\ N_2\ ]$.*
- *Finally, for the monomial class $\mathcal{C}_3^{(Q'')}$, $\Phi_n[\mathcal{C}_3^{(Q'')}] = [\ X_2\ ]$ (for some set variable X_2) and $\Phi_c[\mathcal{C}_3^{(Q'')}] = [\ N_2\ N_3\ N_4\ ]$.*

The two monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ have identical noncopy signatures $[\ Y_1\ ]$. The remaining monomial class, $\mathcal{C}_3^{(Q'')}$, has a unique noncopy signature. Thus, by our results of Section L.9.5, the function for the multiplicity of the tuple t^*_Q, in the answer to the query Q'' on the databases $\{D_{\bar{N}^{(i)}}(Q)\}$, will be computed as the expression for the cardinality of the union of the sets for the copy signatures of the two monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$, which (expression) is multiplied by N_1 and then summed up with the multiplicity monomial for the monomial class $\mathcal{C}_3^{(Q'')}$. By definition, the multiplicity monomial for the monomial class $\mathcal{C}_3^{(Q'')}$ is $1 \times N_2 \times N_3 \times N_4$. (Here, the 1 in the product comes from the X_2 in the noncopy signature of the monomial class.) In turn, the cardinality of the union of the sets for the copy signatures of the two monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ is computed similarly to the function of Example L.9. Thus, the overall multiplicity function will be

$$N_1 \times (N_2 \times N_3 + N_4 \times N_2 - min(N_2, N_4) \times min(N_3, N_2)) +$$

$$+1 \times N_2 \times N_3 \times N_4 \ .$$

Then for the vector $\bar{K}_w = [\ 1\ N_2\ N_3\ N_4\ ]$, the multivariate polynomial for this multiplicity function on the domain $\mathcal{N}^{(\bar{K}_w)}$ will be

$$N_1 \times (N_2 \times N_3 + N_4 \times N_2 - (N_2)^2) + 1 \times N_2 \times N_3 \times N_4 \ .$$

By definition, the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ as computed on this polynomial is $\mathcal{B}^{(Q'')}[\bar{K}_w] = \{N_3, N_4\}$. (Recall that we first discard from the polynomial the product $N_2 \times N_3 \times N_4$, which is not part of the m-covering part of the polynomial. Then we divide the remaining polynomial by N_1, and the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ results from counting the positive and negative occurrences of the variables from $\{N_2, N_3, N_4\}$ in the resulting polynomial.)

We now show that we can obtain the same set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ by using the counting shortcut furnished by Lemma L.2. Indeed, consider the polynomial, call it $f[\bar{K}_w]$,

$$f[\bar{K}_w] = N_2 \times N_3 + N_4 \times N_2 - (N_2)^2,$$

that is used as the input to the algorithm $\mathcal{B}$-CONSTRUCTION in the computation of the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$. The intuition for this polynomial is that the multiplicity of the tuples contributed by the monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ to the set $\Gamma^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$ is computed using:

(a) the value of the cardinality of the union of the sets $N_2^* = \{1, 2, \ldots, N_2\}$ and $N_4^* = \{1, 2, \ldots, N_4\}$ – these sets "arise from" all the elements in the first position *in all the relevant copy signatures, that is, in the copy signatures of monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ (recall that these copy signatures are $\Phi_c[\mathcal{C}_1^{(Q'')}] = [\ N_2\ 1\ N_3\ ]$ and $\Phi_c[\mathcal{C}_2^{(Q'')}] = [\ N_4\ 1\ N_2\ ]$);*

(b) the value of the cardinality of the union of the sets $1^* = \{1\}$ and $1^* = \{1\}$ – these sets "arise from" all the elements in the second position *in all the relevant copy signatures; and, finally,*

(c) the value of the cardinality of the union of the sets $N_3^* = \{1, 2, \ldots, N_3\}$ and $N_2^* = \{1, 2, \ldots, N_2\}$ – these sets "arise from" all the elements in the third *(the last/rightmost)* position *in all the relevant copy signatures.*

It is exactly the above intuition that explains the results of Section L.9.5, specifically the formulation and the use of the result of Proposition L.44 in the construction of the multiplicities of tuple t^*_Q by the inclusion-exclusion principle.

Now given the total order on the variables N_2 through N_4 as provided by the vector $\bar{K}_w = [\ 1\ N_2\ N_3\ N_4\ ]$, the values in (a)–(c) above can each be computed using Lemma L.2. Specifically, the cardinality of each set union in question is N_4 for (a), 1 for (b), and N_3 for (c). Observe that the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ as computed earlier in this example by definition of that set, is exactly $\mathcal{B}^{(Q'')}[\bar{K}_w] = \{N_3, N_4\}$, that is, the result of putting together the N_4 for (a) and the N_3 for (c). (We drop the 1 obtained from (b), as the computation of the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ by its definition does not account for the terms or multipliers that are the constant 1.) □

We are now ready to prove Proposition L.54.

PROOF. (Proposition L.54) We consider first the case where Q' is under the jurisdiction of Proposition L.42 (of Section L.9.2; our query Q' would be the Q'' in the statement of Proposition L.42). In that case, for each vector $\bar{K}_w$ we clearly have that all terms in the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ are solid terms. Hence, by Proposition L.53 we have the desired result of Proposition L.54.

Note that in all cases where Q' is *not* under the jurisdiction of Proposition L.42, it holds that we have $r \geq 2$. (Recall Corollary L.1 of Section L.9.2; the Corollary outlines a special case of Proposition L.42 and covers all cases where $r \leq 1$.) In the remainder of the proof, we consider this case of Q' not satisfying the conditions of Proposition L.42 (and hence $r \geq 2$ for this case).

The proof for this case is by contradiction: We assume that for some vector $\bar{K}_w$, the m-covering part of the multivariate polynomial $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ has the wave $\mathcal{P}_*^{(Q)}$ of the query Q, w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, as a *phantom* term. (The fact that the m-covering part of $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ has the wave of the query Q is by Proposition L.53.) Note that it follows immediately from the definitions of solid and phantom terms (see Section L.10.5) that for *all* vectors $\bar{K}_w$, the m-covering part of the multivariate polynomial $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ has the wave $\mathcal{P}_*^{(Q)}$ of the query Q (w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$) as a *phantom* term.

We will arrive at a contradiction with the above phantom-term assumption, by keeping track of all the occurrences of all the variables N_{m+1} through N_{m+w} in the m-covering parts of the multivariate polynomials $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ and $\mathcal{F}_{(Q)}^{(Q)}[\bar{K}_w]$, for various vectors $\bar{K}_w$. (Recall that $r \geq 2$ holds, hence by Proposition L.4 we have that $w \geq 1$ and thus the set of variables $\{N_{m+1}, \ldots, N_{m+w}\}$ is not empty.) For each possible vector $\bar{K}_w$ (for the vector $\bar{N}$ of Q) and for the m-covering part of the polynomial $\mathcal{F}_{(Q)}^{(Q'')}[\bar{K}_w]$, where Q'' is one of Q and Q', we consider the set $\mathcal{B}^{(Q'')}[\bar{K}_w]$ as defined earlier in this subsection.

Specifically, for the query Q' and for each fixed vector $\bar{K}_w$ we construct the set $\mathcal{B}^{(Q')}[\bar{K}_w]$ as follows. Observe that by definition of the set $\mathcal{B}^{(Q')}[\bar{K}_w]$, the set can be obtained by "breaking up" the relevant polynomial into a *sum* of terms where each term is a single variable from among $\{N_{m+1}, \ldots, N_{m+w}\}$, each multiplied by the coefficient $(+1)$ or by the coefficient (-1). (For instance, the polynomial

$$2 \times N_4 \times N_5 + (N_4)^2$$

of Example L.12 can be rewritten as

$$(N_4 + N_4 + N_5 + N_5) + (N_4 + N_4) \ ,$$

and the summing up of all the variable occurrences separately by variable name and by the sign of each term would result in the correct set $\mathcal{B}^{(Q')}[\bar{K}_w] = \{\ 4 \times N_4, 2 \times N_5\ \}$ for that polynomial.)

In the polynomial, call it $f(Q')$, that results from the above "breaking up" of the relevant polynomial, we now propose to (i) group together the terms that originate from monomial classes having the same noncopy signature, and, in the resulting groups, to (ii) further group together the terms that result from "the same position" in the relevant copy signatures. For instance, using the polynomial $f[\bar{K}_w]$ of Example L.13, we first "break it up" into the sum:

$$f(Q') = N_2 + N_3 + N_4 + N_2 - N_2 - N_2\ ,$$

and then group the elements of this sum further, as

$$f(Q') = (N_2 + N_4 - N_2) + (N_3 + N_2 - N_2)\ .$$

The first grouping above evaluates to item (a) in Example L.13, and the second grouping - to item (c) in the Example. (We do not obtain here groupings by distinct noncopy signatures as suggested in (i) above, because both monomial classes from Example L.13 that contribute to the polynomial $f(Q')$, have the same noncopy signature.)

Now each such grouping "by position in the copy signatures" can be evaluated by Lemma L.2, as the *maximal-value element of the group, according to the vector* $\bar{K}_w$. For instance, in the above $f(Q')$ with groupings that originates from Example L.13, the expression in the first parentheses, $(N_2 + N_4 - N_2)$, is actually $(N_2 + N_4 - min(N_2, N_4))$. Thus, under $\bar{K}_w = [\ 1\ N_2\ N_3\ N_4\ ]$, this expression evaluates to N_4 by Lemma L.2. Similarly, the expression in the second parentheses, $(N_3 + N_2 - N_2)$, which is actually $(N_3 + N_2 - min(N_3, N_2))$, evaluates under $\bar{K}_w$ to N_3 by Lemma L.2. As a result, we obtain the $\mathcal{B}$ for $f(Q')$, that is the correct set $\mathcal{B}^{(Q')}[\bar{K}_w]$, as the set having exactly these two results, N_4 and N_3, each coming from the evaluation of one of the two individual groupings.

It is easy to generalize the above observations, to show that the set $\mathcal{B}^{(Q')}[\bar{K}_w]$ can be computed correctly using the process of "breaking up" the relevant polynomial into a sum of individual variables from $\{N_{m+1}, \ldots, N_{m+w}\}$ (with some of the occurrences of the variables possibly multiplied by (-1)), and of then using Lemma L.2 to evaluate each grouping in that sum to a single variable in $\{N_{m+1}, \ldots, N_{m+w}\}$, each variable always multiplied by $(+1)$.

Note 1. From the correctness of this alternative computation of the set $\mathcal{B}^{(Q')}[\bar{K}_w]$ and by the result of Lemma L.2, we obtain the following for all CCQ queries Q and Q' and for the function $\mathcal{F}^{(Q')}_{(Q)}$ for the multiplicity of the tuple t^*_Q in the answer to Q' on the database $D_{\bar{N}^{(i)}}(Q)$ for each $i \in \mathbb{N}_+$. We obtain that the m-covering part of the function $\mathcal{F}^{(Q')}_{(Q)}$ has, for each vector $\bar{K}_w$, the set $\mathcal{B}^{(Q')}[\bar{K}_w]$ whose *all* elements have natural-number coefficients. That is, no such set $\mathcal{B}^{(Q')}[\bar{K}_w]$ has an element whose integer coefficient is a negative number.

We now arrive at a contradiction with our assumption (see beginning of this proof) that for all vectors $\bar{K}_w$, the m-covering part of the multivariate polynomial $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ has the wave $\mathcal{P}^{(Q)}_*$ of the query Q (w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$) as a *phantom* term. Denote by $F^{(Q')}_m[\bar{K}_w]$ the m-covering part of the multivariate polynomial $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ (for the query Q'), and by $F^{(Q)}_m[\bar{K}_w]$ the m-covering part of the multivariate polynomial $\mathcal{F}^{(Q)}_{(Q)}[\bar{K}_w]$ (for the query Q). Recall that in case $m \geq 1$, we have for each of $F^{(Q)}_m[\bar{K}_w]$ and $F^{(Q')}_m[\bar{K}_w]$ that each term of each of these two polynomials has the product $\Pi^m_{j=1} N_j$ (possibly multiplied by something else.) In case $m \geq 1$, we divide each of $F^{(Q)}_m[\bar{K}_w]$ and $F^{(Q')}_m[\bar{K}_w]$ by this product $\Pi^m_{j=1} N_j$, and call the resulting polynomials (each with integer coefficients and each in terms of the variables $N_{m+1}, \ldots, N_{m+w}$) $f^{(Q)}_m[\bar{K}_w]$ and $f^{(Q')}_m[\bar{K}_w]$, respectively. In case $m = 0$, we rename $F^{(Q)}_m[\bar{K}_w]$ into $f^{(Q)}_m[\bar{K}_w]$, and rename $F^{(Q')}_m[\bar{K}_w]$ into $f^{(Q')}_m[\bar{K}_w]$. Observe that by our assumption $r \geq 2$, we have that for all $m \geq 0$, each of $f^{(Q)}_m[\bar{K}_w]$ and $f^{(Q')}_m[\bar{K}_w]$ is a polynomial in terms of $w \geq 1$ variables $N_{m+1}, \ldots, N_{m+w}$.

Now for an arbitrary $\bar{K}_w$, we rewrite $F^{(Q')}_m[\bar{K}_w]$ equivalently as a sum of one or more groups, each group enclosed in parentheses, where each group is the result of applying the inclusion-exclusion principle to a cardinality-of-set-union expression that would arise in the construction of the function $\mathcal{F}^{(Q')}_{(Q)}$. (See Propositions L.43 and L.44 for the construction. See Example L.6 for an illustration of how one such group would be constructed.) The only reason we are doing the rewriting is that the original, unparenthesized, version of the polynomial $F^{(Q')}_m[\bar{K}_w]$ could have some terms from such groupings canceled out by (opposite-sign) terms, across such groupings. (See Example L.12 for an illustration of such cancellations.) In the remainder of this proof, we use this "parenthesized-and-grouped" version of the polynomial $F^{(Q')}_m[\bar{K}_w]$, as well as the "parenthesized-and-grouped" version of $F^{(Q')}_m[\bar{K}'_w]$ for a possibly different vector $\bar{K}'_w$. Note that from Proposition L.52, we have that the polynomial $F^{(Q)}_m[\bar{K}_w]$, for the query Q and for an arbitrary $\bar{K}_w$, has exactly the same set of (all-solid) terms in both its parenthesized-and-grouped version and its original (unparenthesized) version.

Note 2. By the construction in the previous papagraph, we have that all the resulting groupings are the same across all the vectors $\bar{K}_w$. That is, for any pair of vectors $(\bar{K}_w, \bar{K}'_w)$, the structure of the groupings will be the same in $F^{(Q')}_m[\bar{K}_w]$ and in $F^{(Q')}_m[\bar{K}'_w]$. The reason is, each such grouping reflects the cardinality of the union of a certain collection of sets, and each collection of sets originates from a collection of monomial classes, where the collection "belongs together" because all the monomial classes in the collection share the same noncopy signature. Clearly this grouping of monomial classes has nothing to do with the choice of the vector $\bar{K}_w$.

Fix an arbitrary $\bar{K}_w$. By our assumption, the polynomial $F^{(Q')}_m[\bar{K}_w]$ has the term $\mathcal{P}^{(Q)}_*$, the wave of the query Q, (i) as a phantom term, and (ii) (by Proposition L.53) with a positive coefficient. Then there must exist a grouping in $F^{(Q')}_m[\bar{K}_w]$, call this grouping G^*, such that $\mathcal{P}^{(Q)}_*$ (is a term in G^* and) arises in G^* from the application of the inclusion-exclusion principle to the computation of the cardinality of the union of *at least three* sets, by Propositions L.43 and L.44. (There are at least three such sets because the phantom term $\mathcal{P}^{(Q)}_*$ in G^* has the *positive* sign. This observation is immediate from the definition of the inclusion-exclusion principle.) To the monomial classes that generate these $z \geq 3$ sets in the construction of G^*, we refer as (monomial classes) $P^*_1, P^*_2, \ldots, P^*_z$.

By our assumption that $\mathcal{P}^{(Q)}_*$ is a phantom term in G^*, there must exist a $j \in \{1, \ldots, r\}$ such that the copy signatures of the monomial classes $P^*_1, \ldots, P^*_z$ have, in the jth position of their copy signatures, *at least two* distinct values from the set $\{1, N_{m+1}, \ldots, N_{m+w}\}$. (Otherwise, i.e., assuming that $\mathcal{P}_*^{(Q)}$ does not have a "multiplier" arising from a *min*-expression with at least two distinct elements of the set $\{1,\ N_{m+1},\ \ldots,\ N_{m+w}\}$ as the arguments of the *min*, $\mathcal{P}_*^{(Q)}$ must be the multiplicity monomial of one of the monomial classes $P_1^*,\ \ldots,\ P_z^*$, hence we obtain that $\mathcal{P}_*^{(Q)}$ is a solid term by definition. Therefore, we arrive at a contradiction with our assumption that $\mathcal{P}_*^{(Q)}$ is a phantom term in G^*.) Further, in that jth position, the copy signatures of the monomial classes $P_1^*,\ \ldots,\ P_z^*$ have *at least two* distinct values from the set $\{N_{m+1},\ \ldots,\ N_{m+w}\}$. (Observe that we obtained the latter set by dropping the element 1 from the preceding set.) The reason is, the term $\mathcal{P}_*^{(Q)}$ has exactly $m + r$ occurrences of the variables from the set $\{N_1,\ \ldots,\ N_{m+w}\}$ (see Proposition L.32), hence the part $\mathcal{P}_{copy}^{(Q)}$ (see Definition L.10) of $\mathcal{P}_*^{(Q)}$ must have a separate occurrence of one of $N_{m+1},\ \ldots,\ N_{m+w}$ for each of the r positions in the relevant copy-signatures. As $\mathcal{P}_*^{(Q)}$ has a variable, call it N_A^*, (N_A^* is one of $N_{m+1},\ \ldots,\ N_{m+w}$) in its jth position where $\mathcal{P}_*^{(Q)}$ has a *min*-expression involving at least two distinct elements of the set $\{1,\ N_{m+1},\ \ldots,\ N_{m+w}\}$, it must be that other members of that *min*-expression are also variables (that is, elements of the set $\{N_{m+1},\ \ldots,\ N_{m+w}\}$), with at least one variable that is distinct from N_A^*. Otherwise (i.e., in case where the only other member of the *min*-expression in the jth position of $\mathcal{P}_*^{(Q)}$ is the constant 1) the variable N_A^* cannot be the minimum. (Recall that for the set $\{N_{m+1},\ \ldots,\ N_{m+w}\}$, each element of the set is a natural number and hence cannot accept values below 1.) Denote by N_B^* the variable in this *min*-expression such that under the total order induced by the vector $\bar{K}_w$, the value of N_B^* is not less than the value of any other variable mentioned in this *min*-expression.

We summarize that for the fixed vector $\bar{K}_w$, the variable N_A^* in the (from now on fixed) jth position of $\mathcal{P}_*^{(Q)}$, in group G^*, is the result of evaluating a *min*-expression involving, as arguments, the variable N_A^* and at least one other variable, N_B^* (N_A^* and N_B^* are distinct variables), from among $N_{m+1},\ \ldots,\ N_{m+w}$. From the rule for evaluating *min*-expressions, we obtain immediately that the total order for the fixed vector $\bar{K}_w$ (see Section L.10.3 for the relevant definitions) includes the inequality $N_A^* \leq N_B^*$. It follows from this inequality that in G^*, the copy signatures of the monomial classes $P_1^*,\ \ldots,\ P_z^*$ have, in this fixed jth position of their copy signatures, *at least two* distinct variables from the set $\{N_{m+1},\ \ldots,\ N_{m+w}\}$. Denote by N_α^* the minimal-value such variable (in the jth position in G^*), under the total order induced by $\bar{K}_w$, and denote by N_β^* the maximal-value such variable (in the jth position in G^*), under the total order induced by $\bar{K}_w$. That is, under the total order induced by $\bar{K}_w$ we have that $N_\alpha^* \leq N_A^* \leq N_B^* \leq N_\beta^*$, and the jth position of the copy signatures of the monomial classes $P_1^*,\ \ldots,\ P_z^*$ does not have any variable, call it X, such that $X \in \{N_{m+1},\ \ldots,\ N_{m+w}\}$ and such that either $X \leq N_\alpha^*$ holds under the total order induced by $\bar{K}_w$ (with X and N_α^* being distinct variables), or $N_\beta^* \leq X$ holds under the total order induced by $\bar{K}_w$ (with X and N_β^* being distinct variables). On the other hand, observe that N_α^* and N_A^* could be the same variable, and that N_β^* and N_B^* could be the same variable. (As $\mathcal{P}_*^{(Q)}$ is just one term in the grouping G^*, the arguments of the *min*-expression in the jth position in $\mathcal{P}_*^{(Q)}$ may or may not include all of the variables that occur in the jth position in the copy signatures of all the monomial classes $P_1^*,\ \ldots,\ P_z^*$.)

From the previous paragraph we obtain immediately that the set $\{N_{m+1},\ \ldots,\ N_{m+w}\}$ has at least two elements (at least N_A^* and $N_B^* \neq N_A^*$). We conclude that unless $w \geq 2$, we arrive at a contradiction with our assumption that $\mathcal{P}_*^{(Q)}$ is a phantom term in the polynomial $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$. That is, for all $r \geq 2$ such that $w \leq 1$, we have that $\mathcal{P}_*^{(Q)}$ is a solid term in the polynomial $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$, for all vectors $\bar{K}_w$, which completes our proof of Proposition L.54 for all these cases. Hence in the remainder of this proof, we assume $r \geq 2$ and $w \geq 2$.

We now return to our fixed vector $\bar{K}_w$, and to the fixed group G^* in the polynomial $F_m^{(Q')}[\bar{K}_w]$. In case $m \geq 1$, divide each term of $F_m^{(Q')}[\bar{K}_w]$ and each term of $F_m^{(Q)}[\bar{K}_w]$ by $\Pi_{j=1}^m N_j$ to obtain $f_m^{(Q')}[\bar{K}_w]$ and $f_m^{(Q)}[\bar{K}_w]$ respectively, as discussed earlier in this proof. (In case $m = 0$, recall that we just rename $F_m^{(Q')}[\bar{K}_w]$ into $f_m^{(Q')}[\bar{K}_w]$ and rename $F_m^{(Q)}[\bar{K}_w]$ into $f_m^{(Q)}[\bar{K}_w]$.) Denote by g^* the group in the polynomial $f_m^{(Q')}[\bar{K}_w]$ that results (by such division/renaming) from the group G^* in the polynomial $F_m^{(Q')}[\bar{K}_w]$. Clearly, by G^* in $F_m^{(Q')}[\bar{K}_w]$ having the term $\mathcal{P}_*^{(Q)}$, the group g^* in $f_m^{(Q')}[\bar{K}_w]$ must have a term that is the part $\mathcal{P}_{copy}^{(Q)}$ (see Definition L.10) of $\mathcal{P}_*^{(Q)}$.

We now compute the set $\mathcal{B}^{(Q')}[\bar{K}_w]$, by applying to the polynomial $f_m^{(Q')}[\bar{K}_w]$ our equivalent modification of algorithm $\mathcal{B}$-CONSTRUCTION as discussed in the beginning of this proof. By definition of this modification and by Lemma L.2, each (parenthesized) grouping in the polynomial $f_m^{(Q')}[\bar{K}_w]$ will contribute to the (raw data for the) set $\mathcal{B}^{(Q')}[\bar{K}_w]$ exactly r variables (perhaps with repetitions of the same variable names among the r occurrences) from the set $\{N_{m+1},\ \ldots,\ N_{m+w}\}$, each such contribution multiplied by $(+1)$. Each such contribution corresponds to a separate position (between 1 and r) in the relevant copy signatures, for the given grouping (by cardinality-of-the-set-union). Specifically, from our earlier discussion of the *min*-expression in the fixed jth position of $\mathcal{P}_*^{(Q)}$ it follows that this fixed jth position of $\mathcal{P}_{copy}^{(Q)}$ in the grouping g^* will contribute to the set $\mathcal{B}^{(Q')}[\bar{K}_w]$ the variable N_β^* (multiplied by $(+1)$). We conclude that the set $\mathcal{B}^{(Q')}[\bar{K}_w]$ has an element $C \times N_\beta^*$, with $C \in \mathbb{N}_+$. (The reason for C possibly being greater than 1 is that, apart from the fixed jth position of $\mathcal{P}_{copy}^{(Q)}$ in the grouping g^*, other positions in this or other groupings could also contribute N_β^* to the (raw data for the) set $\mathcal{B}^{(Q')}[\bar{K}_w]$.)

Now consider (and fix) an arbitrary vector $\bar{K}'_w$ such that N_β^* is the second element of this vector, and that N_α^* is the last element of the vector. (For instance, if all the variables in all the vectors of the form $\bar{K}_w$ are N_2, N_3, N_4, N_5, and if we have $N_\beta^* = N_3$ and $N_\alpha^* = N_2$, then one possibility for $\bar{K}'_w$ is $\bar{K}'_w = [\ 1\ \ N_3\ \ N_5\ \ N_4\ \ N_2\ ]$.) Clearly, the fixed jth position of $\mathcal{P}_{copy}^{(Q)}$ in the grouping g^* will now contribute to the set $\mathcal{B}^{(Q')}[\bar{K}'_w]$ the variable N_α^* (multiplied by $(+1)$).

Observe that for the vector $\bar{K}'_w$ it holds that the polynomial $f_m^{(Q')}[\bar{K}_w]$ contributes to the (raw data for the) set $\mathcal{B}^{(Q')}[\bar{K}'_w]$ the variable N_β^* *only* for those groupings and only for those (lth) positions in those groupings where in all the copy signatures for the monomial classes for the grouping, the lth position in all the copy signatures has either the variable N^*_β (in the lth position in at least one such copy signature) or the constant 1 (in the lth position in any number, including zero, of such copy signatures). Observe also that by Note 2 (earlier in this proof), for the vector $\bar{K}_w$ (which we fixed earlier in this proof), *all* such lth positions, exactly the same as for $\bar{K}'_w$, would also each contribute the variable N^*_β to the (raw data for the) set $\mathcal{B}^{(Q')}[\bar{K}_w]$. We conclude that the set $\mathcal{B}^{(Q')}[\bar{K}'_w]$ either does not have an element for N^*_β at all (in case none such lth position exists), or has an element for N^*_β with a multiplier $C' \in \mathbb{N}_+$ such that C' is *strictly less than* C, where C is the multiplier for N^*_β in the set $\mathcal{B}^{(Q')}[\bar{K}_w]$.

We infer that for the two fixed (distinct) vectors $\bar{K}_w$ and $\bar{K}'_w$, the sets $\mathcal{B}^{(Q')}[\bar{K}_w]$ and $\mathcal{B}^{(Q')}[\bar{K}'_w]$ are not identical sets. Using Corollaries L.2 and L.3, we conclude that for the CCQ queries Q and Q', $Q \equiv_C Q'$ cannot hold. Thus, we have arrived at a contradiction with the assumption that that for at least one vector $\bar{K}_w$, the m-covering part of the multivariate polynomial $\mathcal{F}^{(Q')}_{(Q)}[\bar{K}_w]$ has the wave $\mathcal{P}^{(Q)}_*$ of the query Q (w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$) as a *phantom* term. Q.E.D. $\Box$

L.10.8 Proof of the main result of this section

We can now put together all the auxiliary results of this section, to prove Proposition L.47 of Section L.10.2. Actually the proof is immediate from Proposition L.54 and from the definition of solid terms of polynomials for our multiplicity functions (as given in Section L.10.5). We can finally conclude that the result of Theorem 4.1 holds. Q.E.D.

M. PROOFS FOR SECTION 5

M.1 Proof of Proposition 5.1

In this section we provide a proof of Proposition 5.1. Example 5.1 serves as an illustration of the proof.

Proof. *If:* This direction is straightforward. Take an arbitrary M-identity SCVM, call it μ, from Q onto Q'. The desired GCM from Q_{ce} onto Q'_{ce} is obtained by extending μ to the copy variables introduced into the condition of Q to construct Q_{ce} and Q'_{ce}. The desired extended mapping is a GCM that extends onto the subgoal of Q_{ce} and of Q'_{ce} the association that μ induces from the subgoals of the query Q to the subgoals of the query Q'. That is, the extended mapping would induce *the same* mapping as μ at the subgoal level of the two queries, modulo the copy variables added when transforming query Q (Q', respectively) into Q_{ce} (into Q'_{ce}, respectively).

Only-If: In this part of the proof, we use the following definitional notation for the four queries: $Q(\bar{X}) \leftarrow L, M$; $Q_{ce}(\bar{X}) \leftarrow L_{ce}, M_{ce}$; $Q'(\bar{X}) \leftarrow L', M'$; and $Q'_{ce}(\bar{X}) \leftarrow L'_{ce}, M'_{ce}$. By definitions of Q', Q_{ce}, and Q'_{ce}, we have that (a) $L' \subseteq L$, (b) $L'_{ce} \subseteq L_{ce}$, (c) $M = M'$, (d) $M'_{ce} \subseteq M_{ce}$, and (e) $M \subseteq M'_{ce}$.

Let ν be a GCM from Q_{ce} onto Q'_{ce}. Denote by ν_1 the mapping resulting from restricting the domain of the GCM ν to the range of ν. Observe that the range of ν is the set of all the variables (including copy variables) and constants in the atoms in the set L'_{ce}. As ν induces an *automorphism* from the subset L'_{ce} of L_{ce} (in Q_{ce}) to the condition L'_{ce} of Q'_{ce}, ν_1 is a bijection, and therefore there exists an inverse mapping ν_1^{-1}. Consider the composition $\nu_1^{-1} \circ \nu$. By construction, $\nu_1^{-1} \circ \nu$ is a well-defined mapping whose domain is the set of all terms occurring in the set L_{ce}. Further, $\nu_1^{-1} \circ \nu$ is an identity mapping when the domain of $\nu_1^{-1} \circ \nu$ is restricted to the terms occurring in the set L'_{ce}. In particular, $\nu_1^{-1} \circ \nu$ is an identity mapping on each variable in the set M. (Recall that the set L'_{ce} contains all the variables in the set M'_{ce}.) Finally, observe that the mapping $\nu_1^{-1} \circ \nu$ is a GCM from the query Q_{ce} onto the query Q'_{ce}.

Now consider the mapping, call it φ, resulting from restricting the domain of the mapping $\nu_1^{-1} \circ \nu$ to all the terms occurring in the query Q. We show that φ is an M-identity SCVM from the query Q onto the query Q'. Indeed, we have that conditions (1)–(3) and (5) of Definition 3.1 are satisfied for φ by definition of the queries Q_{ce} and Q'_{ce}, as well as by construction of the mapping $\nu_1^{-1} \circ \nu$. Further, by construction φ is an identity mapping when restricted to the domain M.

We now observe that from Q' having all the copy variables of the query Q, for each subgoal s of Q that is in the set $L - L'$ (as set difference), s must be a relational subgoal of the query Q. Hence we have, by definition of φ, the satisfaction by φ of condition (4) of Definition 3.1. (The reason is, each such subgoal s is mapped by φ into some atom, either relational or copy-sensitive, in the condition L' of the query Q'. Further, each relational subgoal s' in L' is mapped by φ into a relational subgoal, specifically into s' itself.) In addition, by each such subgoal s in $L - L'$ being a relational atom, we avoid the complication of φ mapping the copy variable of s – *in case s were a copy-sensitive atom* – to a different variable (recall that $\varphi(s)$ cannot be s, by $s \notin L'$), and of thus φ violating the requirement that the desired SCVM map each element of the set M into itself. The mapping φ could also map s, *in case s were a copy-sensitive atom*, into a relational atom, thus violating condition (5) of Definition 3.1.

Finally, φ maps each atom in $L - L'$ into an atom in L', and maps L' onto itself. We conclude that φ is an M-identity SCVM from Q onto Q'. Q.E.D. $\Box$

M.2 Proof of Proposition 5.2

In this section we provide a proof of Proposition 5.2.

In the proof of Proposition 5.2 we use a straightforward observation, as follows.

Proposition M.1. *For $k \geq 2$, let Q_1, Q_2, ..., Q_k be CCQ queries such that for each $i \in \{2, \ldots, k\}$, we have that Q_i is a reduced-condition query for Q_{i-1}. Further, for each $j \in \{1, \ldots, k-1\}$, let there exist a CVM μ_j from Q_j to Q_{j+1}. Then $\mu_{k-1} \circ \mu_{k-2} \circ \ldots \circ \mu_1$ is a CVM from Q_1 to Q_k.* $\Box$

The result of Proposition M.1 is immediate from Definition 3.1.

Proof. (Proposition 5.2) For a CCQ query Q, denote by Q^{min} the output of the algorithm Minimize-CCQ-queries when given Q as input. (Clearly, for each input the algorithm outputs a CCQ query.) We prove that Q^{min} is a minimized version of Q by proving for Q^{min} all the items of Definition 2.3 (of minimized versions of CCQ queries).

We first show that item (1) of Definition 2.3 holds for Q^{min} w.r.t. Q. That is, we show that Q^{min} is a reduced-condition query for Q. Q^{min} has been obtained by the algorithm

Minimize-CCQ-queries applying to Q (and then to its successive reduced-condition queries) one or more M-identity SCVMs. (Observe that there exists a M-identity SCVM from the input query Q to its regularized version, which is used in line 1 of the pseudocode for the algorithm.) By Proposition M.1 and by all these SCVMs being *M-identity* SCVMs, there exists an M-identity SCVM, μ, from Q to itself, such that μ is the composition of those SCVMs. By definition of M-identity SCVMs, Q^{min} is the query $\mu(Q)$ and is thus a reduced-condition query for Q.

We now show that item (2) of Definition 2.3 holds for Q^{min}. That is, we show that Q^{min} is a minimized query.

- First, we have that Q^{min} is a regularized query. (A CCQ query is called a regularized query if its regularized version is the query itself.) Indeed, line 1 of the pseudocode for algorithm Minimize-CCQ-queries obtains the regularized version of the query Q, and after that the algorithm uses that regularized version starting from line 2 of the pseudocode. Thus, for each (if any) M-identity SCMV, call it μ, applied by the algorithm to the current query Q', $\mu(Q')$ cannot be an unregularized query, by construction of $\mu(Q')$. We conclude that once the algorithm is done with the applications of all possible such M-identity SCVMs, Q^{min} is also a regularized query.
- Second, denote by Q'' the query that results from removing one or more subgoals from the *regularized* (as obtained in the previous item) query Q^{min}. There are two cases:
  - Q'' has strictly fewer multiset variables than Q^{min} does. In this case, by Theorem 3.1, $Q'' \equiv_C Q^{min}$ cannot hold.
  - Q'' has all the multiset variables of Q^{min}. Then, by Q^{min} being an output of the algorithm Minimize-CCQ-queries and by Proposition 5.1, there cannot exist a GCM from Q^{min}_{ce} into Q''_{ce}. It follows by Theorem 3.2 that $Q'' \equiv_C Q^{min}$ cannot hold.

We conclude that Q^{min} is a minimized query.

Finally, we show that item (3) of Definition 2.3 holds for Q^{min} and for Q. That is, we show that $Q^{min} \equiv_C Q$ holds. Indeed,

- $Q^{min} \sqsubseteq_C Q$ holds by the existence of an (M-identity) SCVM from Q to Q^{min}, see reasoning in the beginning of this proof, and by Theorem 3.3.
- $Q \sqsubseteq_C Q^{min}$ holds by the existence of an *identity* SCVM from Q^{min} to Q (recall that Q^{min} is a reduced-condition query for Q that retains all the multiset variables of Q) and again by Theorem 3.3.

We conclude that $Q^{min} \equiv_C Q$ holds. $\square$

M.3 Proof of Proposition 5.4

In this section we provide a proof of Proposition 5.4.

Proof. Denote by V_1 (by V_2, respectively) the number of variables in the query Q_1 (in Q_2, respectively). Similarly, denote by C_1 (by C_2, respectively) the number of constants in the query Q_1 (in Q_2, respectively). Denote by M_1 the set of multiset variables of the query Q_1, and by M_2 the set of multiset variables of the query Q_2. As usual, $|M_1|$ denotes the cardinality of M_1, and similarly for M_2.

From μ_1 being a CVM, we obtain immediately that (i) $|M_1| \geq |M_2|$, and that (ii) $C_1 \leq C_2$ (recall that a CVM maps each constant into an identical constant). Now from μ_1 being a mapping from the condition of the query Q_1 *onto* the condition of the query Q_2, we have that (iii) the condition of query Q_1 has at least as many subgoals as the condition of query Q_2, and that (iv) $C_1 + V_1 \geq C_2 + V_2$. From (ii) and (iv) we obtain immediately that (v) $V_1 \geq V_2$.

Symmetrically, from μ_2 being a CVM and a mapping from the condition of Q_2 *onto* the condition of Q_1, we have that (a) $|M_1| \leq |M_2|$, that (b) $C_1 \geq C_2$, that (c) the condition of query Q_1 has *at most* as many subgoals as the condition of query Q_2, that (d) $C_1 + V_1 \leq C_2 + V_2$, and that (e) $V_1 \leq V_2$.

It follows that:

- $|M_1| = |M_2|$;
- $C_1 = C_2$;
- The conditions of Q_1 and of Q_2 are of the same cardinality (as bags of atoms); and
- $V_1 = V_2$;

We use reasoning similar to the above to obtain that, in addition, μ_1 induces a bijection from the head vector of query Q_1 to the head vector of query Q_2. Symmetrically, μ_2 induces a bijection from the head vector of query Q_2 to the head vector of query Q_1.

Now, by definition of CVM, the mapping, call it ν_1, that results from restricting the domain of μ_1 to the set M_1 of multiset variables of the query Q_1, is a bijection from M_1 to the set M_2 of multiset variables of the query Q_2. Similarly, the mapping, call it ν_2, that results from restricting the domain of μ_2 to the set M_2, is a bijection from M_2 to M_1.

From the above observations and from the definition of SCVM we conclude that μ_1 is an isomorphism SCVM from Q_1 to Q_2, and that μ_2 is an isomorphism SCVM from Q_2 to Q_1. $\square$

M.4 Proof of Theorem 5.1

In this section we provide a proof of Theorem 5.1.

Proof. (Theorem 5.1) Let Q be a CCQ query. The existence of a minimized version of Q follows from soundness of algorithm Minimize-CCQ-queries. Now suppose there exist two distinct minimized versions of Q, Q_1 and Q_2, where each of Q_1 and Q_2 satisfies Definition 2.3. We show in this proof that there exists an isomorphism M-identity SCVM from Q_1 to Q_2, and that there exists an isomorphism M-identity SCVM from Q_2 to Q_1. (See Example 5.2 for an illustration.)

We first establish that there exists an M-identity SCVM from Q *onto* Q_1, and another from Q *onto* Q_2. Indeed, by definition of minimized version of a CCQ query, we have that Q_1 is a reduced-condition query for Q such that $Q_1 \equiv_C Q$. By Theorem 3.1 we then have that Q_1 has all multiset variables of the query Q. Then, by $Q_1 \in \mathcal{Q}_{min}(Q)$, by Theorem 3.2, and by Proposition 5.1, there must exist an M-identity SCVM from Q onto Q_1, call this SCVM μ_1. Similar reasoning provides the proof for the existence of an M-identity SCVM μ_2 from Q onto Q_2.

Consider the M-identity SCVM μ_2 from Q onto Q_2. Observe that, by Q_1 being a reduced-condition query for Q that has all the multiset variables of Q, μ_2 is an M-identity SCVM *from* Q_1 *into* Q_2.

We now prove that μ_2 is a mapping from Q_1 *onto* Q_2. (By symmetric reasoning, we obtain that μ_1 is a mapping from

Q_2 *onto* Q_1.) Indeed, assume toward a contradiction that there exists a *proper* reduced-condition query for Q_2, call this query Q_2^*, such that μ_2 is a mapping from Q_1 into Q_2^*. Then, by μ_2 being an M-identity SCVM, Q_2^* has all multiset variables of the query Q. From the existence of an identity SCVM from Q_2^* to Q (Q_2^* is a reduced-condition query for Q) and by Theorem 3.3, we have $Q \sqsubseteq_C Q_2^*$. Conversely, by $Q \equiv_C Q_1$, by the existence of SCVM μ_2 from Q_1 to Q_2^* and by Theorem 3.3, we have $Q_2^* \sqsubseteq_C Q$. We infer that a *proper* reduced-condition query for Q_2, that is Q_2^*, is combined-semantics equivalent to the query Q. Thus, Q_2 cannot be a minimized version of the query Q, and we arrive at the desired contradiction. We conclude that μ_2 is an M-identity SCVM from Q_1 *onto* Q_2.

By μ_2 being an M-identity SCVM from Q_1 *onto* Q_2, by (symmetrically) μ_1 being an M-identity SCVM from Q_2 *onto* Q_1, and by Proposition 5.4, we obtain that μ_2 is an *isomorphism* M-identity SCVM from Q_1 to Q_2, and that μ_1 is an *isomorphism* M-identity SCVM from Q_2 to Q_1. Q.E.D. □

N. NECESSARY AND SUFFICIENT CONDITIONS OF [6] FOR COMBINED-SEMANTICS QUERY EQUIVALENCE

Cohen in [6] provides necessary and sufficient conditions for combined-semantics equivalence of CQ queries, possibly with negation, comparisons, and disjunction. For these necessary and sufficient conditions to be applicable, both queries to be tested for combined-semantics equivalence are to satisfy one of the following conditions:

1. Neither of the two queries has set variables; or
2. Neither of the two queries has multiset variables; or
3. Neither of the two queries has same-name predicate twice or more in positive (i.e., nonnegated) subgoals; or
4. Each query is a join of a set (i.e., no multiset variables) subquery with a multiset (i.e., no set variables) subquery. The formal definition is that neither query may have a subgoal that would have both a multiset variable and a set variable; or
5. Neither query may have copy variables.

Now consider a restriction of the query language studied in [6] to CCQ queries. In the remainder of this section, we consider the above conditions 1–5 only as applied to the queries that satisfy this restriction. (That is, in the remainder of this section we consider CQ combined-semantics queries only, without any extensions of this query language.) Under this query-language restriction, each of the above conditions 1–5 enforces that each *CCQ* query in question be an explicit-wave query, by Definition 4.1 in this current paper. Specifically:

1. Whenever neither of the two queries has set variables, then both queries are explicit-wave queries because in each query, each copy-sensitive subgoal has no set variables. (See Section 6 in this current paper for this syntactic sufficient condition for a CCQ query to be an explicit-wave query.)
2. Whenever neither of the two queries has multiset variables, then neither query has copy-sensitive subgoals. Hence, both queries in question are explicit-wave queries by Definition 4.1 (1).
3. Whenever neither of the two queries has same-name predicate twice or more in positive (i.e., nonnegated) subgoals, then both queries are explicit-wave queries because neither (CCQ) query has self-joins. (The fact that a CCQ query without self-joins is an explicit-wave query is an easy inference from Definition 4.1 (2).)
4. Whenever neither query may have a subgoal that would have both a multiset variable and a set variable, then both queries are explicit-wave queries because in each query, each copy-sensitive subgoal has no set variables. (See Section 6 in this current paper for this syntactic sufficient condition for a CCQ query to be an explicit-wave query.)
5. Whenever neither query may have copy variables, then both queries are explicit-wave queries by Definition 4.1 (1).

We conclude that if we apply to *only* CCQ queries the necessary and sufficient conditions of [6] for query combined-semantics equivalence, then each of these conditions would be applicable exclusively to pairs of explicit-wave queries. Thus, *when all the queries in question are required to be CCQ queries,* we have that all the necessary and sufficient conditions of [6] for combined-semantics equivalence of queries are subsumed by Theorem 6.3 of this current paper and by its variant, Theorem N.1, as follows. (The result of Theorem N.1 is immediate from Theorems 4.1 and 6.1.)

THEOREM N.1. *Given explicit-wave CCQ queries Q_1 and Q_2. Then $Q_1 \equiv_C Q_2$ if and only if there exists a CVM from Q_1 to Q_2, and another from Q_2 to Q_1.* □

Observe that the CCQ query Q of Example 3.1 does not satisfy (individually) any of the conditions 1–5 of this section. Thus, none of the necessary and sufficient query-equivalence conditions of [6] would apply to a pairing of this query with an *arbitrary* query in the query language considered in [6]. (By definition, see Definition 2.1, CCQ queries do belong to the query language considered in [6].) We make the same observation about the CCQ query Q' of Example 3.1, as well as about the query CCQ Q of Example 3.2. Still, by Theorem 6.3 (or by Theorem N.1) of this current paper we obtain that (i) $Q \not\equiv_C Q'$ for the queries of Example 3.1, and that (ii) $Q \equiv_C Q'$ for the queries of Example 3.2. (See also Section 6.)

O. PROOF OF PROPOSITION 6.1

In this section we provide a proof of Proposition 6.1.

PROOF. *If:* Let ν_1 be an isomorphism SCVM from Q_1^{min} to Q_2^{min}. We show that there exists a CVM from Q_1 to Q_2. By Q_1^{min} being (up to an isomorphism M-identity SCVM, see Theorem 5.1) the output of algorithm MINIMIZE-CCQ-QUERIES, by construction of the algorithm MINIMIZE-CCQ-QUERIES, and by Proposition M.1, we have that there exists a SCVM, μ_1, from Q_1 to Q_1^{min}. Finally, there exists an identity SCVM, ι_2, from Q_2^{min} to Q_2. By another application of Proposition M.1, we obtain that $\iota_2 \circ \nu_1 \circ \mu_1$ is a (S)CVM from Q_1 to Q_2. Symmetrically, we obtain that there exists a CVM from Q_2 to Q_1. Q.E.D.

Only-If: Suppose there exists a CVM ξ_1 from Q_1 to Q_2. We show that there exists an isomorphism SCVM from Q_1^{min} to Q_2^{min}. There exists a SCVM, μ_2, from Q_2 to Q_2^{min}. (See the If part of the proof for the justification of the existence of μ_2. The justification is given there – symmetrically – for the existence of a SCVM μ_1 from Q_1 to Q_1^{min}.) By Proposition M.1, we obtain that the mapping $\mu_2 \circ \xi_1$ is a CVM from Q_1 to Q_2^{min}.

Construct a mapping ν_1, by restricting the domain of $\mu_2 \circ \xi_1$ to the set of terms of the query Q_1^{min}. By Q_1^{min} having all the multiset variables of the query Q_1 and being a reduced-condition query for Q_1, we have that ν_1 is a CVM from Q_1^{min} to Q_2^{min}.

From the existence of a CVM from Q_1 to Q_2 and of a CVM from Q_2 to Q_1, by Theorem 3.3 we obtain that $Q_1 \equiv_C Q_2$. Further, from $Q_1 \equiv_C Q_1^{min}$, from $Q_2 \equiv_C Q_2^{min}$, and by transitivity of $\equiv_C$, we obtain $Q_1^{min} \equiv_C Q_2^{min}$. From Theorem 3.1 we have that the queries Q_1^{min} and Q_2^{min} have the same number of multiset variables. Therefore, ν_1 must be a *S*CVM from Q_1^{min} to Q_2^{min}.

We now show that ν_1 is a SCVM from from Q_1^{min} *onto* Q_2^{min}. The proof is by contradiction: Assume that ν_1 is not an *onto* mapping. Then the query $\nu_1(Q_1^{min})$ is a proper reduced-condition query for Q_2^{min}, and thus also a proper reduced-condition query for Q_2. We have the following:

- $Q_1 \equiv_C Q_1^{min}$ (by definition of Q_1^{min});
- $Q_1 \equiv_C Q_2$ (by the existence of a CVM ξ_1 from Q_1 to Q_2, by the existence of a CVM from Q_2 to Q_1, and by Theorem 3.3);
- $Q_2 \sqsubseteq_C \nu_1(Q_1^{min})$ (by the existence of an identity CVM from $\nu_1(Q_1^{min})$ to Q_2 and by Theorem 3.3); and
- $\nu_1(Q_1^{min}) \sqsubseteq_C Q_1^{min}$ (by ν_1 being a CVM from Q_1^{min} to $\nu_1(Q_1^{min})$ and by Theorem 3.3).

Using transitivity of $\sqsubseteq_C$, we infer that $Q_2 \equiv_C \nu_1(Q_1^{min})$, where $\nu_1(Q_1^{min})$ is a proper reduced-condition query for Q_2^{min}. We conclude that Q_2^{min} cannot be a minimized query for Q_2, and thus arrive at the desired contradiction. Therefore, ν_1 is a SCVM from from Q_1^{min} *onto* Q_2^{min}.

Symmetrically to the above proof for ν_1, we obtain that there exists a SCVM, ν_2, from Q_2^{min} *onto* Q_1^{min}. From the existence of two "onto" CVMs ν_1 and ν_2, using Proposition 5.4, we conclude that ν_1 is an isomorphism SCVM from Q_1^{min} to Q_2^{min}, and that ν_2 is an isomorphism SCVM from Q_2^{min} to Q_1^{min}. Q.E.D. □

P. PROOF OF SUFFICIENT CONDITION FOR A CCQ QUERY TO BE AN EXPLICIT-WAVE QUERY

PROPOSITION P.1. *Given a CCQ query Q such that each copy-sensitive subgoal of Q has no set variables. Then Q is an explicit-wave query.* □

PROOF. We prove that for all queries such as Q in the statement of Proposition P.1, each such query satisfies Definition 4.1. Indeed, consider a query Q satisfying the condition that each copy-sensitive subgoal of Q has no set variables. In case Q has at most one copy-sensitive subgoal, we obtain immediately that Q satisfies Definition 4.1 (1). Thus, in the remainder of this proof we assume that Q has at least two copy-sensitive subgoals. We will show that in this case, Q always satisfies Definition 4.1 (2).

Let (μ_1, μ_2) be an arbitrary pair of noncopy-permuting GCMs from Q_{ce} to itself such that μ_1 and μ_2 agree on $M_{noncopy}$. Consider an arbitrary copy-sensitive subgoal of Q, call this subgoal s. By definition of the query Q_{ce}, s must be a subgoal of Q_{ce}. The atom s may have as arguments only constants, head variables of the query Q, and multiset variables. (Recall that no set variables of Q may be arguments of s.) Now each of μ_1 and μ_2 map each constant to itself (by each of μ_1 and μ_2 being a GCM), and by each of μ_1 and μ_2 being a mapping from Q_{ce} to itself we obtain that each of μ_1 and μ_2 maps each head variable of Q_{ce} (that is, each head variable of Q, by definition of Q_{ce}) to itself. Finally, consider each multiset *noncopy* variable, call it Y, such that Y is an argument of the atom s. By μ_1 and μ_2 agreeing on $M_{noncopy}$, we have that $\mu_1(Y)$ and $\mu_2(Y)$ are the same term of the query Q. (Recall that, by definition of Q_{ce}, we have that for all terms that are present in Q_{ce} but not in Q, each such term is a copy variable.) We conclude that atoms $\mu_1(s)$ and $\mu_2(s)$ have the same relational template. Hence, Q satisfies Definition 4.1 (2). Q.E.D. □