Combined-Semantics Equivalence Is Decidable For a Practical Class of Conjunctive Queries

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

ABSTRACT

The problems of query containment and equivalence are fundamental problems in the context of query processing and optimization. In their classic work [2] published in 1977, Chandra and Merlin solved the two problems for the language of conjunctive queries *(CQ queries)* on relational data, under the "set-semantics" assumption for query evaluation. Alternative semantics, called *bag* and *bag-set semantics*, have been studied since 1993; Chaudhuri and Vardi in [5] 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 [8, 9] 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 [9] 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 [9].

Our goal in this paper is to continue the study of equivalence of CQ queries. We focus on the problem of determining whether two CQ queries are combined-semantics equivalent. We continue the tradition of [2, 5, 9] of studying this problem using the tool of containment between queries. This paper introduces a syntactic necessary and sufficient condition for equivalence of queries belonging to a large natural language of "explicit-wave" combined-semantics CQ queries; this language encompasses (but is not limited to) all set, bag, and bag-set queries, and appears to cover all combined-semantics CQ queries that are expressible in SQL. Our result solves in the positive the decidability problem of determining combined-semantics equivalence for pairs of explicit-wave CQ queries. That is, for an arbitrary pair of combined-semantics CQ queries, it is decidable (i) to determine whether each of the queries is explicit wave, and (ii) to determine, in case both queries are explicit wave, whether or not they are combined-semantics equivalent, by using our syntactic criterion. (The problem of determining equivalence for general combined-semantics CQ queries remains open. Even so, our syntactic sufficient containment condition could still be used to determine that two general CQ queries are combined-semantics equivalent.) Our equivalence test, as well as our general sufficient condition for containment of combined-semantics CQ queries, reduce correctly to the special cases reported in [2, 5] for set, bag, and bag-set semantics. Our containment and equivalence conditions also properly generalize the results of [9], provided that the latter are restricted to the language of (combined-semantics) CQ queries.

1. INTRODUCTION

Query containment and equivalence are recognized as fundamental problems in evaluation and optimization of database queries. 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 SQL have prompted researchers [5, 12] 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 [5] published in PODS in 1993, Chaudhuri and Vardi focused on the hard problem of bag containment for CQ queries. The paper [5] formulates containment and equivalence results, including equivalence tests,

for bag and bag-set queries. However, the full version of the paper [5] 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 [8, 9] 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*. The focus of the papers [8, 9] was on formulating conditions for *combined-semantics equivalence* of queries expressed using that syntax. Intuitively, two queries are combined-semantics equivalent, denoted $\equiv_C$, if on each database, the two queries return the same answers, with the same multiplicity of each answer tuple. To show the practical value of the combined-semantics formalism, Cohen exhibited in [9] a number of real-life SQL queries that can be expressed as combined-semantics CQ queries, but cannot be expressed using set, bag, or bag-set semantics. In the following example, we show three realistic combined-semantics queries, which we then use to illustrate the contributions of this paper.

EXAMPLE 1.1. *The application domain that we use here is based on a data-warehousing example from [11]. Consider a retailer that has multiple stores. The retailer carries many items and has an elaborate relational database/wa- rehouse for analysis, marketing, and promotion purposes. One of the tables in the database has the schema* `Pos(trans- actionID, storeID, itemID, 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.*

The business-development division of this store chain would typically analyze general trends and high-level aggregates over the `Pos` *relation. We assume that for this reason, the data analysts in that division access data through a* view, *with the schema* `Sales(storeID, itemID, date, amount)`*, rather than working with the "raw" data in the* `Pos` *relation. To ensure correctness of the data-analyses results, the* `Sales` *relation is a* bag-valued *relation; that is, for each tuple with specific values of the attributes* `storeID`*,* `itemID`*,* `date`*, and* `amount`*, the* `Sales` *relation makes available as many copies of this tuple as there are rows with this information in the base relation* `Pos`*.*

Suppose that this business-development division of the 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 a fixed date,[1] *January 1, 2012. Consider a SQL query* `Qc` *that could be used for the purpose of this analysis.*

```
(Qc) SELECT storeID, amount FROM Sales S
     WHERE EXISTS
        (SELECT * FROM Sales
         WHERE storeID = S.storeID AND itemID = S.itemID
         AND amount = S.amount AND date = '01/01/12')
```

For each store ID, query `Qc` *returns separately the amount for each transaction that took place in that store on the date* `01/01/12`*. Moreover, for each item that was sold in the store on the date* `01/01/12`*, the query* `Qc` *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. 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* `01/01/12`*, then all they have to do to write such a query is to (i) add to the query* `Qc` *the clause* `GROUP BY storeID`*, and to (ii) replace* `amount` *by* `sum(amount)` *in the* `SELECT` *clause of the resulting query. Note that to evaluate the resulting analysis query, the query processor would first evaluate the query* `Qc`*, and would then apply to the answer to* `Qc` *the above grouping and aggregation.*

Due to the large size of the relation `Sales` *(one million rows), the self-join of* `Sales` *in the query* `Qc`*, via a correlated subquery, should be avoided if at all possible. For some SQL queries, such direct removal of a subquery is indeed possible. Consider, for instance, a query* `Qc2`*, whose definition is almost the same as that of the query* `Qc`*:*

```
(Qc2) SELECT storeID, amount FROM Sales S
      WHERE date = '01/01/12'
      AND EXISTS
         (SELECT * FROM Sales
          WHERE storeID = S.storeID AND itemID = S.itemID
          AND amount = S.amount)
```

The only difference between the queries `Qc` *and* `Qc2` *is that the equality comparison with the date* `'01/01/12'`*, which can be found in the subquery of the query* `Qc`*, is a condition in the main body of the query* `Qc2`*.*

It is easy to argue informally, based on the "intended meaning" of the query `Qc2`*, that* `Qc2` *always returns the same number of the same answers as the query* `Qc2min`*, which is obtained by removing the correlated subquery from* `Qc2`*:*

```
(Qc2min) SELECT storeID, amount FROM Sales S
         WHERE date = '01/01/12'
```

Indeed, by the results of this paper, the queries `Qc2` *and* `Qc2min` *are combined-semantics equivalent.*

Thus, one question we could pose is whether the queries `Qc` *and* `Qc2` *(or alternatively the queries* `Qc` *and* `Qc2min`*) are combined-semantics equivalent. If we can answer this question in the positive, then the query* `Qc2min`*, which is very efficient to execute, can be evaluated on the relation* `Sales` *to obtain the correct answer to the query of interest* `Qc`*. Another question we could of course pose is whether the query* `Qc` *is combined-semantics equivalent to the SQL query that results from removing the correlated subquery of the query* `Qc`*. Intuitively, this is unlikely, as the latter query would lack the equality comparison with the date* `'01/01/12'`*.* □

Combined-semantics CQ queries such as the query `Qc` of Example 1.1, with grouping and aggregation added, arise naturally in on-line analytical processing *(OLAP)* applications [14, 15]. 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, 4], which are the only well-understood joins in the literature on OLAP query optimization. See [14, 15, 16, 17] for more detailed discussions of why queries with nested subqueries and with "non-star joins" are natural and frequent in OLAP. (For additional extended illustrations of such queries, see the online version [6] of this paper.)

[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.

It turns out that the equivalence tests reported in [2, 5, 8, 9] do not apply to the SQL queries `Qc` or `Qc2` of Example 1.1, even though both queries are expressible in the syntax of [8, 9] for combined-semantics conjunctive queries. At the same time, we can use the results of this paper and of [7] (the latter results are also available online in [6]) to establish the following:

1. $Qc \equiv_C Qc2$ does not hold, and neither does $Qc \equiv_C Qc2min$;
2. Qc is not combined-semantics equivalent to a CQ set, bag, or bag-set query (intuitively, a CQ query belonging to each of the three classes would be expressible as a SQL query without subqueries);
3. $Qc2 \equiv_C Qc2min$ does hold (observe that $Qc2min$ is a conjunctive *bag-semantics* query);
4. $Qc2min$ is a minimized version of $Qc2$; and
5. Qc is the only minimized version of itself.

To the best of our knowledge, none of the above inferences 1–5 can be made using formal results reported in prior work. We use the results of [7] in making inferences 4–5 only, and use the results of this current paper in making all the five inferences.

Our contributions.

We study equivalence of unaggregated SQL queries with equality comparisons (including comparisons with constants) and possibly with subqueries. We follow the approach of [9], 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").[2] In the remainder of this paper, all queries are expressed using the Datalog-based formalism of [9]. In the remainder of this paper, we refer to combined-semantics CQ queries as *CCQ queries.*

We focus on the problem of determining whether two CCQ queries are combined-semantics equivalent. We continue the tradition of [2, 5, 9] of studying these problems using the tool of containment between queries. Our first specific contribution is to introduce in this work "covering mappings" *(CVMs)* between CCQ queries, and to show that CVMs furnish a sufficient condition for combined-semantics containment, for all CCQ queries.

The second specific contribution of this paper is in providing a necessary condition for combined-semantics equivalence of CCQ queries. To formulate this condition, we isolate a large class of CCQ queries, which we call "explicit-wave queries".[3] 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 [9] provides its sufficient and necessary equivalence tests. Further, it appears that all combined-semantics CQ queries that are expressible in SQL are explicit-wave queries. (Please see Section 4 for the details.) Our necessary condition for query equivalence is asymmetric – it states that if for CCQ queries Q_1 and Q_2 we have the combined-semantics equivalence $Q_1 \equiv_C Q_2$, *and* Q is an explicit-wave query, then there exists a CVM from Q_2 to Q_1. We also show that this necessary condition is tight, in that we exhibit a specific CCQ query Q that is *not* explicit wave, such that our necessary equivalence condition does not hold when applied to Q and to another CCQ query Q', even though the two queries are provably combined-semantics equivalent. The problem of combined-semantics equivalence for general CCQ queries remains open.

The importance of the contributions of this paper is in solving in the positive the decidability problem of determining combined-semantics equivalence, for pairs of explicit-wave CCQ queries. That is, the results reported in this paper imply immediately that for an arbitrary pair of CCQ queries, it is decidable (i) to determine whether each of the queries is explicit wave, and (ii) to determine, in case both queries are explicit wave, whether or not they are combined-semantics equivalent, by using our syntactic CVM-based criterion. Note that even in case where one or both of the input queries are not explicit wave, our syntactic (CVM-based) sufficient condition for combined-semantics equivalence could still be used to determine that the two queries are equivalent.

The results of this paper combined with those of [7] can be used directly in query optimizers for database-management systems, as well as for developing optimization 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.

The remainder of this paper is organized as follows. In Section 1.1 we review related work. Section 2 formulates the background notions and results. In Section 3, we introduce covering mappings (CVMs) and present our sufficient CVM-based condition for combined-semantics equivalence, which is applicable to all CCQ queries. Section 4 formalizes the notion of explicit-wave queries, presents in Theorem 4.1 the CVM-based necessary condition for equivalence of explicit-wave CCQ queries, and shows that this condition does not necessarily apply to general CCQ queries. Section 4 also contains the decidability test, which is the main result of this paper. Finally, Section 5 contains the proof of Theorem 4.1.

1.1 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. In this current paper we extend the containment and equivalence results of [2] to general CQ combined-semantics queries, and show the limitations of each extension.

Equivalence tests for CQ bag and bag-set queries were formulated by Chaudhuri and Vardi in [5]; correctness of the tests follows from the results of [9]. Our equivalence results for CQ combined-semantics queries reduce correctly to the special cases of CQ bag and bag-set queries, as given in [5]. 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 [5].

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 [13] for original

[2]Section 1 in [9] provides some details of the translations.

[3]The term "explicit-wave query" is due to the structures generated by the proof of the necessary equivalence condition reported in this paper.

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 [10]. Please see [5, 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 [8, 9], 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 [9] 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 [9] 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 [9] 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 [9], including of its implicit sufficient condition for query containment, provided that the results of [9] 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 [9], which contains an excellent overview of the literature in this direction.

2. PRELIMINARIES

2.1 Combined semantics: The framework [9]

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 $adom(D)$ if and only if there exists in D an atom $r(c'_1, \ldots, c'_m; N')$ (for some copy number $N' \geq 1$ and 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 the 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 $\bar{X}$ is a (possibly empty) vector, 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 $M_{noncopy} := M - M_{copy}$.* □

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. Consider an illustration.

EXAMPLE 2.1. *Let CCQ query Qc be as follows.*

$$Qc(X, Y) \leftarrow sales(X, Z, U, Y; i), sales(X, Z, c, Y), \{Z, U, i\}.$$

Suppose that we interpret c as the constant value '01/01/12.' Then this query is the Datalog version of the SQL query `Qc` *of Example 1.1 (in Section 1).*

The condition L of the query Qc is the conjunction of the two subgoals of Q with the predicate `sales`. *The variables X and Y are the head variables of the query Qc. The set $\{Z, U, i\}$ is the set M of multiset variables of Qc; the set M_{copy} of Qc comprises the (only) copy variable i of Qc, and the set $M_{noncopy}$ of Qc has the multiset noncopy variables Z and U of Qc. By definition, $M = M_{copy} \cup M_{noncopy}$. This query does not have set variables.* □

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 $k \geq 0$ elements, we say that a CCQ query with head $Q(\bar{X})$ is a *CCQ k-ary query*. In the special case where $k = 0$, we say that Q is a *CCQ Boolean query,* and denote its head by $Q()$.

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 $p(\gamma\bar{T}; N) \in D$.

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* $\Gamma_{\bar{S}}(Q, D)$.

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*

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

(In the special case where Q is a Boolean query, each $\gamma(\bar{T})$ as above is interpreted as a separate copy of the empty tuple.) 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. Consider an illustration.

EXAMPLE 2.2. *Let CCQ query Qc be as in Example 2.1, and consider a database D whose Sales relation has two distinct tuples:* $(85, 433, 01/01/12, 264; 2)$ *and* $(85, 433, 03/15/12, 264)$. *(Note that the first tuple of the Sales relation is present in two copies in the database D.) Then $Res_C(Qc, D)$ is a bag of exactly three copies of the tuple* $(85, 264)$. □

Under certain circumstances, combined semantics coincides with set, bag, or bag-set semantics. Please see [9] 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.

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

Homomorphisms and set queries. Given two conditions $\phi(\bar{U})$ and $\psi(\bar{V})$, 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, (2) for each relational atom $r(U_1, \ldots, U_n)$ of ϕ we have that $r(h(U_1),\ \ldots,\ h(U_n))$ is in ψ, and (3) for each copy-sensitive atom $p(W_1, \ldots, W_n; i)$ of ϕ we have that $p(h(W_1),\ \ldots,\ h(W_n); h(i))$ is in ψ. Given two CCQ k-ary queries $Q_1(\bar{X}) \leftarrow \phi(\bar{X}, \bar{Y}), M_1$ and $Q_2(\bar{X}') \leftarrow \psi(\bar{X}', \bar{Y}'), M_2$, 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 $Q \equiv_S Q'$.

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 2.2. *[5] 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.* □

(Two CCQ queries Q_1 and Q_2, with respective sets of multiset variables M_1 and M_2, are *isomorphic* if there exists a one-to-one containment mapping from Q_1 *onto* Q_2 such that the mapping induces a bijection from M_1 to M_2, and there exists another containment mapping from Q_2 onto Q_1, with (symmetrically) the same properties.)

Combined-semantics queries. The next result is a sufficient condition of [9] for equivalence of two queries under combined semantics. In [9], Cohen formulates each of Definition 2.3 and Theorem 2.3 for CCQ queries that may also contain negation and inequality comparisons. (In condition (3) of Definition 2.3 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 2.3. *(**Multiset-homomorphism [9]**) Let $Q(\bar{X}) \leftarrow L, M$ and $Q'(\bar{X}') \leftarrow L', M'$ be two k-ary CCQ queries, for $k \geq 0$. Let φ be a mapping from the terms of Q' to the terms of Q.*[4] *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 variables $Y \neq Y' \in M'$.* □

For every mapping φ that satisfies conditions 1–3 of Definition 2.3, 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 2.3. *[9] Given CCQ queries Q and Q'. If Q and Q' are multiset homomorphic then $Q \equiv_C Q'$.* □

Note 1. Theorem 2.3 is proved in [9] via showing that for two (generalized) CCQ queries Q and Q', the existence of a multiset-homomorphism from Q' to Q implies $Q \sqsubseteq_C Q'$.

It is shown in [9] that the sufficient equivalence condition of Theorem 2.3 is not necessary for the query classes considered in [9].

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

3. CONTAINMENT AND MAPPINGS

In this section, for combined-semantics containment of CCQ queries, we outline two necessary conditions, Theorems 3.1 and 3.2, which are proved in [7], and introduce a sufficient condition, Theorem 3.3. The latter result properly generalizes both (i) the sufficient condition outlined in [5] for bag containment of CQ queries, and (ii) the general sufficient containment condition for CCQ queries that can be obtained from [9]. 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 outline here two necessary conditions for a CCQ query Q being combined-semantics contained in CCQ query Q', Theorems 3.1 and 3.2. Both results are proved in [7]; we will need them in this current paper in our discussion of Theorem 4.1.

THEOREM 3.1. *Let Q and Q' be two k-ary CCQ queries, for a $k \geq 0$. 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.)

We call a pair (Q, Q') of CCQ queries a *containment-compatible CCQ pair* if (i) The (nonnegative) 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.

The next result, Theorem 3.2 (also proved in [7]), generalizes 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 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}. A formalization is given in [7].

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}.* □

Neither Theorem 3.1 nor 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 introduce covering mappings *(CVMs)* between CCQ queries, and study properties of CVMs.

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 in [7] 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 2.3, there does not exist a multiset homomorphism [9], or even a GCM, from Q to Q'. At the same time, by the results of [7], Q' is a minimized version of Q. We can ascertain this fact by using a CVM, μ, from Q to Q': μ = { $X \to X$, $Y \to Y$, $i \to i$, $Z \to X$ } . □

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 2.3); 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.* □

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.

PROOF. (Proposition 3.2) 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 $\mu(Q')$ be 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 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 $\mu = \{X \to X, Y \to Y, Z \to X, i \to i\}$ 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 $\mathcal{T}(Q)$.) 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. □

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 [5] stating that a containment mapping[5] 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 [5].

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 [5] (see Appendix B), claims containment $Q \sqsubseteq_C Q'$, but no CVM exists from Q' to Q.

We now prove Theorem 3.3. Remarkably, the proof is almost verbatim the proof, in [9], of the result that is given as Theorem 2.3 in this current paper.

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 $\bar{t}$ be a tuple of constants in $adom(D)$. Suppose that $Res_C(Q_d, D)$ contains $k > 0$ occurrences of $\bar{t}$. (The case where the head vectors of the two queries are empty, and hence $\bar{t}$ is the empty tuple, is proved in the same way as the case where $\bar{t}$ is a tuple with at least one constant. We omit the proof of the former case.) We show that $Res_C(Q', D)$ contains at least k occurrences of $\bar{t}$. 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 $\bar{t}$. 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, $\gamma_i^* \circ \varphi(\bar{X}') = \bar{t}$, 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 $\bar{t}$.

Our arguments apply for all D and for all $\bar{t}$. Therefore, $Q_d \sqsubseteq_C Q'$ and (by $Q_d \equiv_C Q$) we have $Q \sqsubseteq_C Q'$, as required. □

3.3 CVMs and multiset homomorphisms of [9]

In this subsection we compare CVMs with multiset homomorphisms [9], see Definition 2.3. 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 C.1 in Appendix C.) 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 C.)

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

[5]The "containment mapping" terminology of [5] results from the use in that paper of a syntax for bag queries that does not coincide with the syntax of [9] used in this current paper. See Appendix A for a detailed discussion.

result and from Example 3.2, we obtain that the restriction of Theorem 2.3 (due to [9]) 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. (By Theorem 3.3, 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 [9], provided that the latter is applied to CCQ queries only; see Note 1 in Section 2.2. (By Definition 2.3 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 equivalence of CCQ queries, 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. As we will see in Section 5, establishing this result is not trivial. We also formulate and prove the main result of this paper, Theorem 4.2, and show how it gives rise to an algorithm for determining whether two explicit-wave CCQ queries are or are not combined-semantics equivalent.

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. Further, a polynomial-time checkable sufficient condition for a CCQ query to be explicit wave is that each subgoal of the query not contain both a set variable and a copy variable:

PROPOSITION 4.1. *Given a CCQ query Q such that each copy-sensitive subgoal of Q has no set variables. Then Q is an explicit-wave query.* □

(This result is straightforward. For completeness, a proof can be found in Appendix D.)

By Proposition 4.1, each CCQ set query is an explicit-wave query, and so is each CCQ bag query and each CCQ bag-set query. In addition, Example 1.1 in Section 1 exhibits SQL queries that are explicit-wave CCQ queries, and such that none of the previously known tests for combined-semantics equivalence are applicable to them.

Further, again by Proposition 4.1, we posit that explicit-wave queries include all those CCQ queries that are expressible in SQL. (That is, it appears that all "practical" CCQ queries are explicit wave.) Indeed, we are not aware of a way in SQL to enforce, for some relation in the main `FROM` clause of a query without the `DISTINCT` keyword, that the multiplicity of one individual argument of the relation not contribute to the multiplicity of the query answers.

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 E) 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 F). 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, as we have just argued, implicit-wave CCQ queries cannot occur under set, bag, or bag-set semantics. By Theorem 4.1 in this section, 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.

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, see Theorem 4.2.

Due to the well-known example of [5] (Appendix B), 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. 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 shown, under the three traditional semantics all CCQ queries are explicit-wave queries. In [9], Cohen explores query classes that properly subsume the class of CCQ queries. When restricted to CCQ queries, all the necessary and sufficient conditions of [9] for combined-semantics query equivalence require the queries to be explicit-wave queries. (We note that none of the necessary and sufficient conditions of [9] 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 G for all the details.)

We now formulate the main result of this paper, our decidability result for combined-semantics equivalence of CCQ explicit-wave queries.

THEOREM 4.2. *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.* □

The result of Theorem 4.2 is immediate from Theorems 4.1 and 3.3.

We use Theorem 4.2 to develop the following algorithm for determining whether two explicit-wave CCQ queries Q and Q' are or are not combined-semantics equivalent. (The correctness of the algorithm is by Theorems 3.1 and 4.2.)

Input: Pair (Q, Q') of CCQ queries.

Output: Determination whether $Q \equiv_C Q'$ does or does not hold, in case both Q and Q' are explicit-wave queries.

Procedure:

1. If (Q, Q') is not an equivalence-compatible CCQ pair, then stop and return "not combined-semantics equivalent."
2. Use Definition 4.1 to ascertain that both Q and Q' are explicit-wave queries.
3. Whenever both Q and Q' are explicit-wave queries, use the test of Theorem 4.2 to report whether $Q \equiv_C Q'$ does or does not hold.

We note that instead of using in step 2 of the algorithm the procedure of Definition 4.1, which is in co-NP, we can alternatively apply the polynomial-time sufficient condition of Proposition 4.1 for a CCQ query to be explicit wave. As discussed earlier in this section, this sufficient condition appears to cover all practical cases of SQL queries. Further, step 3 of the algorithm is NP-complete; this is easy to see when we recall that containment mappings of [2] are a special case of CVMs. (It is straightforward to argue that finding, for an equivalence-compatible CCQ pair (Q, Q'), a CVM from Q to Q' when both queries are *set* queries is at least as hard as finding a CVM from Q to Q' in the general case. This claim follows from the fact that, in case where (Q, Q') is an equivalence-compatible CCQ pair, each CVM from Q to Q' is by definition a bijection from the set M of multiset variables of Q to the set M' of multiset variables of Q'. Intuitively, as CCQ set queries do not have multiset variables, when looking for the existence of a CVM from CCQ set query Q to CCQ set query Q' we "have to consider the maximal possible number of options" when enumerating candidate CVMs from the terms of Q to the terms of Q'.)

Finally, we observe that even if one or more of the queries Q and Q' is an implicit-wave CCQ query, then still Theorem 3.3 could be used in at least some cases, to ascertain that $Q \equiv_C Q'$. (Recall that Theorem 3.3 applies to all CCQ queries, rather than just explicit wave.) Similarly, step 1 of the algorithm would determine correctly nonequivalence for pairs of CCQ queries where neither query in the pair has to be explicit wave.

5. PROOF OF THEOREM 4.1

In this section we provide a proof of Theorem 4.1.

5.1 Intuition for the Proof and Extended Example

5.1.1 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 [2] 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. (Observe that based on the conditions of Theorem 4.1, 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 (suitable modifications of) "canonical databases" for Q. (See Section 5.2.1 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 5.3 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, ..., 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 $N_1^{(i)}$, ..., $N_n^{(i)}$ 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, ..., 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 $Res_C(Q'', D_{\bar{N}^{(i)}}(Q))$. See Sections 5.4 through 5.6 and Section 5.9 for the construction of the function. Sections 5.8 and 5.9 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 $\Pi_{j=1}^n N_j$. 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 $\nu_Q^{(i)} \circ \gamma$ is a SCVM from Q to itself. (Note the presence in the product $\Pi_{j=1}^n N_j$ 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 $\Pi_{j=1}^n N_j$, regardless of the relationship between the structures of Q and Q'. We show that whenever $\mathcal{F}_{(Q)}^{(Q')} = \Pi_{j=1}^n N_j$, 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 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 $\Gamma_{\bar{S}}^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$, of all generative assignments from Q to databases $D_{\bar{N}^{(i)}}(Q)$. (The set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$ is the set projection, on the vector $\bar{S}(Q)$, of the set, denoted $\Gamma^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$, of all satisfiable assignments for Q and $D_{\bar{N}^{(i)}}(Q)$ that contribute the tuple t_Q^* to the answer to Q on $D_{\bar{N}^{(i)}}(Q)$.) We call this monomial, $\mathcal{P}_*^{(Q)}$, *the wave of the query Q* w.r.t. $\{D_{\bar{N}^{(i)}}(Q)\}$. (See Section 5.7 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 $\nu_Q^{(i)}$ 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 5.1 in Section 5.1 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 5.10, with its main result Proposition 5.47.)

5.1.2 The main propositions that imply the result of Theorem 4.1

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 5.6; Section 5.8 contains an extended illustration of these constructs. Intuitively, a monomial class for a query and database is a set of certain-format assignments from the query to the database; the multiplicity monomial of a monomial class is a monomial that expresses how many tuples of a certain form the assignments for that monomial class contribute to the bag of answers to the query on the database.) A high-level map of the entire proof of Theorem 4.1 can be found in Section 5.1.4.

- Proposition 5.33 of Section 5.7 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 5.34 of Section 5.7 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 5.47 of Section 5.10 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. $\{D_{\bar{N}^{(i)}}(Q)\}$.

5.1.3 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 5.6 in Section 5.9.6 is an extended variant of this Example 5.1. In addition, Example 5.6 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 5.1. *For the query Q of Example 4.1, we use the following notation for is variables:*

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

This notation makes it clear that the variables Y_1 through Y_4 are all multiset (copy or noncopy) variables of the query.

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 $N_1 := 1$, $N_2 := 1$, $N_3 := 2$, 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 $X_1 \to a$, $Y_1 \to b$, $Y_2 \to c$, $X_2 \to d$, and $X_3 \to e$. By definition of combined-semantics query evaluation, these generative assignments together contribute to the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$, for $t_Q^* = (a)$, exactly $\Pi_{j=1}^4\ N_j^{(i)} = 6$ distinct tuples. (The set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$ is the set projection, on the vector $\bar{S}(Q)$, of the set, denoted $\Gamma^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$, of all satisfiable assignments for Q and $D_{\bar{N}^{(i)}}(Q)$ that contribute the tuple t_Q^* to the answer to Q on $D_{\bar{N}^{(i)}}(Q)$.) 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 $\Gamma_{\bar{S}}^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$. (Recall that the set $\Gamma_{\bar{S}}^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$ is the result of* set*-projecting out the columns for the* set *variables of the query Q from the set $\Gamma^{(t_Q^*)}(Q, D_{\bar{N}^{(i)}}(Q))$ (i.e., from the set of all assignments for Q and $D_{\bar{N}^{(i)}}(Q)$ that contribute the tuple t_Q^* to the answer to Q on $D_{\bar{N}^{(i)}}(Q)$). This* set *projection "bundles together," into the same extendible assignment, possibly multiple distinct satisfying assignments from the query to the database.) 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 of Example 4.1, we show in Example 5.6 in Section 5.9.6 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 $\mathcal{P}_*^{(Q)}$. (Intuitively, such a class*

of assignments from Q' to the databases cannot exist because the query Q' has the same set variable in both subgoals, whereas Q has different set variables in the two subgoals.) It is easy to verify that for the queries of Example 4.1, there does not exist a SCVM from Q' to Q. □

5.1.4 A Map of the Proof

The remainder of the proof of Theorem 4.1 is structured as follows. In Section 5.2, we lay out the notation, assumptions, and basic results that run through the entire proof of Theorem 4.1. In Section 5.3, we show how to construct the family of databases $\mathcal{D}_{\bar{N}}(Q)$ for an arbitrary CCQ query Q. Then, Sections 5.4 through 5.7 furnish all the building blocks for the construction of the function $\mathcal{F}_{(Q)}^{(Q'')}$, for the family of databases $\mathcal{D}_{\bar{N}}(Q)$ and for a CCQ query Q'', such that Q and Q'' are an equivalence-compatible CCQ pair. (By Theorem 3.1, whenever two CCQ queries are combined-semantics equivalent, they must constitute an equivalence-compatible CCQ pair, please see Section 3 for the details.)

Among these sections, Section 5.7 introduces "the wave" of a CCQ query Q; that "wave" notion will play a major role in our reasoning in Section 5.10 to complete the proof of Theorem 4.1. In addition, in Section 5.7 we prove Propositions 5.33 and 5.34, which are two of the three main results toward the proof of Theorem 4.1 (see Section 5.1.2).

Further in the proof of Theorem 4.1, Section 5.8 provides an extended example that illustrates in a single flow the contents of Sections 5.2 through 5.7. Then, Section 5.9 puts together the function $\mathcal{F}_{(Q)}^{(Q'')}$ based on the results of the preceding sections.

Finally, Section 5.10 answers the following question: For two CCQ queries Q and Q' such that $Q \equiv_C Q'$, when does Q' have the wave of Q? The answer to this question completes the proof of Theorem 4.1. In particular, the proof of Proposition 5.47, which is the third and last main result toward the proof of Theorem 4.1 (see Section 5.1.2), can be found in Section 5.10.

5.2 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. To make referencing the notation easier, every effort has been made to introduce all of the notation into these initial subsections, which are separate from sections for (portions of) the proof of Theorem 4.1.

5.2.1 Canonical databases of CCQ queries

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: Extended canonical databases: 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 $\{ X_1 \to a, Y_1 \to b, Y_2 \to c, X_2 \to d, i \to 1, j \to 1 \}$.

General CCQ queries: Copy-neutral canonical databases: A database is called a *copy-neutral canonical database* for a CCQ query Q if it can be obtained from an extended canonical database for Q by changing, in an unrestricted way, the values of zero or more copy numbers in the latter database. That is, in a copy-neutral canonical database for Q, the copy number of each ground atom can (but does not have to) (i) coincide with the copy number of another ground atom in the database, and (ii) coincide with an element of the active domain of the database (in case the active domain includes natural numbers). Clearly, each CCQ query can be associated with an infinite number (up to isomorphism) of copy-neutral canonical databases.

5.2.2 The Queries

The basics.

For the fixed (input) CCQ queries Q and Q', as well as for the CCQ query Q'' that we use 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 $l \geq 0$ 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 5.1. *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 5.2. *(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 5.3. *Let $M = \emptyset$. Then Theorem 4.1 holds.* □

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. (For an illustration, see query Q' in Example 4.1.)

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 5.4. *(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) $m + w \geq 1$.* □

Notation for query variables.

For ease of exposition, we assume that in case where $l \geq 1$, each of Q, Q', and Q'' has l distinct head variables. (Recall that $l \geq 0$ denotes the head arity of the query Q. Handling the cases where ($l \geq 1$ and) 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 (in case $l \geq 1$) 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 $u \geq 1$) 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 5.4 (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 $Y'_1, \ldots, Y'_m$ (by $Y''_1, \ldots, Y''_m$, 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, in case $l \geq 1$ we denote by $X'_1, \ldots, X'_l$ (by $X''_1, \ldots, X''_l$, 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.

5.2.3 *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.

5.3 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 copy-neutral canonical databases for the query Q. In the exposition in this section we use the notation introduced in Section 5.2.

Throughout the proof of Theorem 4.1, whenever we discuss or use "the family of databases as constructed in Section 5.3", we always refer to the family of databases built based on the fixed input query Q (see Section 5.2.2), *regardless of the context.* (This convention is lifted in part of Example 5.5 in Section 5.9.4.)

5.3.1 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 $X_1, \ldots, X_{l+u}$ ($l + u \geq 0$), 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 $t_Q^* = \nu_0[\bar{X}]$ in case $l \geq 1$, and as the empty tuple in case $l = 0$. (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 5.2.) We assume that the vector $\bar{N}$ accepts values from the Cartesian product of $m+w$ copies of the set $\mathbb{N}_+$ of natural numbers; we denote by $\mathcal{N}$ this domain from which $\bar{N}$ accepts values. 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 $N_1^{(i)}, N_2^{(i)}, \ldots, N_{m+w}^{(i)}$ for the natural-number values, from left to right, in the vector $\bar{N}^{(i)}$. That is, the natural number $N_j^{(i)}$, $1 \leq j \leq m + w$, 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 $N_j^{(i)}$ such that $1 \leq j \leq m$, define $S_j^{(i)}$ as a set of $N_j^{(i)}$ distinct constants such that all the sets $S_j^{(i)}$ ($1 \leq j \leq m$) are pairwise disjoint and such that $S_j^{(i)} \bigcap P = \emptyset$ and $S_j^{(i)} \bigcap S_0 = \emptyset$ 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 $\bigcup_{j=1}^{m} S_j^{(i)}$ in case $m \geq 1$.

Finally, for the vector $\bar{N}^{(i)}$, for each $i \in \mathbb{N}_+$, we define mapping $\nu_Q^{(i)}$, to be used in Section 5.3.3 and in other parts of the proof of Theorem 4.1. Define the domain of the mapping $\nu_Q^{(i)}$ as the union $P \bigcup S_0 \bigcup S_*^{(i)}$. Define mapping $\nu_Q^{(i)}$ as follows:

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

5.3.2 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 $\nu_t^{noncopy}$. Now consider the case where the query Q is such that we have $m \geq 1$. Let $\mathcal{S}^{(i)}$ be the cross product $S_1^{(i)} \times S_2^{(i)} \times \ldots \times S_m^{(i)}$. For each tuple t in $\mathcal{S}^{(i)}$ in this case $m \geq 1$, we treat t as an assignment $\nu_t^{noncopy}$ 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 $\nu^{copy}(Y_j) := N_j^{(i)}$ 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 $Y_{k(j)}$, with $k(j) \in \{m+1, \ldots, m+w\}$, 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_Q^{copy}, to be used in Section 5.4 and beyond, again for the case where $r \geq 1$. The mapping ν_Q^{copy} uses "the same logic" as the mapping ν^{copy}, except that ν_Q^{copy} 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 $N_{m+1}^{(i)}, \ldots, N_{m+w}^{(i)}$ 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 $\nu_Q^{copy}(Y_j) := N_j$ 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 $\nu_Q^{copy}(Y_j) := \nu_Q^{copy}(Y_{k(j)})$. 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 $C'_{k(j)}$.

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 $\nu_t^{noncopy}[Y_j]$ 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.

5.3.3 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 5.2.

PROPOSITION 5.5. *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 $\nu_Q^{(i)}$ 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 from its domain T to the set $\nu_Q^{(i)}(T)$.* □

PROPOSITION 5.6. *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 5.6 (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 5.7. *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 5.8. *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 $D_{\bar{N}^{(i)}}(Q)$ a copy-neutral canonical database for the 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 copy-neutral canonical database for query Q.

(iii) For each pair (D_1, D_2) of copy-neutral 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 $D_{\bar{N}^{(i)}}(Q)$ a copy-neutral canonical database of the query Q such that the active domain of that copy-neutral 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 5.9. *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.*[6] *Then there exists in $D_{\bar{N}^{(i)}}(Q)$ a copy-neutral 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 5.6 (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 to 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 $D_{\bar{N}^{(i)}}(Q)$ a copy-neutral canonical database for Q. (See Proposition 5.8 for the justifications.) It follows that 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. □

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, whenever $l \geq 1$) is present in $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$, for each $i \geq 1$. (Recall that in case $l = 0$, we have that t_Q^* is the empty tuple. For ease of exposition, in the remainder of this proof of Theorem 4.1, we omit the discussion of the case $l = 0$ and of its implications for the definition of the tuple t_Q^*.)

PROPOSITION 5.10. *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 $Res_C(Q, D_{\bar{N}^{(i)}}(Q))$. □

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

5.4 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 5.3. Let Q'' be an arbitrary CCQ query that satisfies all the requirements (for Q'' specifically) of Section 5.2.2. 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"[7] the fixed tuple t_Q^* (as defined in Section 5.3) 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 $i \in \mathbb{N}_+$.

The notions introduced in this and previous sections are illustrated by a detailed Example 5.3 in Section 5.8.

5.4.1 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 5.2.2 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 5.1. *(**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. □

DEFINITION 5.2. *(**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 $\bar{Z}^{(k)}$

[7] That is, each of these assignments generates a separate element of the set $\Gamma^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$.

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 5.2 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 5.3. *(**(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 5.4. *(**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 5.2. *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 $\mathcal{A}_2$, $\theta_{21} = \{X \to 1, Y \to 1, i \to 1\}$ is a unity valid assignment mapping from the terms of Q'' to the elements of $adom(D) \cup \mathbb{N}_+$.*
- *Given $\mathcal{A}_3$, $\theta_3 = \{X \to 3, Y \to 3, i \to 7\}$ 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 $\theta_{22} = \{X \to 1, Y \to 1, i \to 2\}$ and $\theta_{23} = \{X \to 1, Y \to 1, i \to 3\}$. 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 5.6 (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 5.3.1. 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 5.5. *(**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 $D_{\bar{N}^{(i)}}(Q)$.* □

The intuition for Definition 5.5 is as follows. Suppose that for an association $\mathcal{A}$ for a query Q'' and for a database $D_{\bar{N}^{(i)}}(Q)$, the association $\mathcal{A}$ contributes at least one valid assignment mapping to the set $\Gamma(Q'', D_{\bar{N}^{(i)}}(Q))$. Then, intuitively, the atom-signature of $\mathcal{A}$ shows the pattern in which the condition of the query Q'' can be mapped into the (subset $\mathcal{S}_{C(Q)}$ of the) condition of the query Q.

Recall the notation $Y_1'', \ldots, Y_{m+r}''$ of Section 5.2.2 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_Q^{copy} defined in Section 5.3.2: ν_Q^{copy} 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 use ν_Q^{copy} to 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 $Z \in M_{copy}$, $\nu^{copy-sig}(h) := \nu_Q^{copy}(Z)$.

Recall (see beginning of Section 5.4.1) 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 5.6. *(**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 $[\ \nu^{copy-sig}(\psi_{\bar{N}^{(i)}}^{gen(Q)}[d_{j1}]), \ldots, \nu^{copy-sig}(\psi_{\bar{N}^{(i)}}^{gen(Q)}[d_{jr}])\ ]$.*

□

DEFINITION 5.7. *(**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 $[\ \nu_Q^{(i)}(\theta(Y_1'')), \ldots, \nu_Q^{(i)}(\theta(Y_m''))\ ]$.*

□

The intuition for Definitions 5.6 and 5.7 parallels the intuition for atom-signatures, see the discussion, above, of Definition 5.5. That is, suppose that for an association $\mathcal{A}$ for a query Q'' and for a database $D_{\bar{N}^{(i)}}(Q)$, the association $\mathcal{A}$ contributes at least one valid assignment mapping to the set $\Gamma(Q'', D_{\bar{N}^{(i)}}(Q))$. Then, intuitively, we have that:

- The copy-signature of $\mathcal{A}$ shows how the *copy variables* of the query Q'' could be mapped, via all the valid assignments associated with $\mathcal{A}$, into copy variables of (the subset $\mathcal{S}_{C(Q)}$ of subgoals of) the query Q;

  some of the "copy variables" in the vector could equal unity, to indicate those cases where the image, in the sense of $\mathcal{A}$ via $\Phi_c[\mathcal{A}]$, of a copy-sensitive subgoal of Q'' is a relational subgoal of Q;

  and
- The noncopy-signature of $\mathcal{A}$ shows how the *multiset noncopy variables* of the query Q'' could be mapped, via all the valid assignments associated with $\mathcal{A}$, into terms in (the subset $\mathcal{S}_{C(Q)}$ of subgoals of) the query Q.

In the remainder of this proof, we will use the atom-signatures, the copy-signatures, and the noncopy-signatures of associations for Q'' and $D_{\bar{N}^{(i)}}(Q)$, for each natural number i, to classify all the t_Q^*-valid assignment mappings from Q'' to the database $D_{\bar{N}^{(i)}}(Q)$. We will use the classification to construct the closed-form expressions, $\mathcal{F}_{(Q)}^{(Q'')}$, for the multiplicity of the tuple t_Q^* in the answer to the query Q'' on each such database $D_{\bar{N}^{(i)}}(Q)$. (For an introduction to $\mathcal{F}_{(Q)}^{(Q'')}$, please refer back to Section 5.1.1.) More specifically, in Section 5.6 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 $\Gamma_{\bar{S}}^{(t_Q^*)}(Q'', D_{\bar{N}^{(i)}}(Q))$, that (entries) are contributed by the (valid) mappings of $\mathcal{A}$ for Q'' and $D_{\bar{N}^{(i)}}(Q)$. Please see Example 5.3 for an extended illustration of copy-signatures and of noncopy-signatures of associations.

The following result is immediate from the definitions.

PROPOSITION 5.11. *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 5.11 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.

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

The results of Propositions 5.12 through 5.17 are immediate from the definitions (unless discussed further in this subsection).

PROPOSITION 5.12. *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 5.2.3, as well as to our definition of the databases $D_{\bar{N}^{(i)}}(Q)$ using the subset $\mathcal{S}_{C(Q)}$ of subgoals of the query Q. (Recall that all the elements of the set $\mathcal{S}_{C(Q)}$ have pairwise distinct relational templates.) Note that the associations $\mathcal{A}_1$ and $\mathcal{A}_2$, in the statement of Proposition 5.13, are not restricted to have either the same atom-signature or different atom-signatures.

PROPOSITION 5.13. *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 5.14. *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 5.15. *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 5.16. *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 $V^{(i)}_{jk}$ 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 $V^{(i)}_{jk} := 1$ 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 5.16, 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 5.17. *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* $1 \leq n_k \leq V^{(i)}_{jk}$ *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* $\Pi_{k=1}^{r} V^{(i)}_{jk}$ *in case* $r \geq 1$*; and*

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

□

5.5 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 5.4, 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, $\mathbb{A}^{(i)}_{Q''}$, which includes some of the above associations, and "captures", in a very precise sense (see Proposition 5.18), all of the t^*_Q-valid assignment mappings for Q'' and $D_{\bar{N}^{(i)}}(Q)$. We will use the set $\mathbb{A}^{(i)}_{Q''}$ in Section 5.6, 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 5.8. *(***Set** $\mathbb{A}^{(i)}_{Q''}$ **of** t^*_Q**-valid assignment mappings for** Q'' **and** $D_{\bar{N}^{(i)}}(Q)$*) Let* $i \in \mathbb{N}_+$*.* The set $\mathbb{A}^{(i)}_{Q''}$ 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* $\mathcal{A} \in \mathbb{A}^{(i)}_{Q''}$ *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 $\mathbb{A}^{(i)}_{Q''}$ by $R^{(i)}_{Q''}$. Further, whenever $R^{(i)}_{Q''} \geq 1$, we refer to the

individual elements of the set $\mathbb{A}^{(i)}_{Q''}$ as $A^{(i)}_j$, for $1 \leq j \leq R^{(i)}_{Q''}$. (We will avoid the confusion as to which query $A^{(i)}_j$ "refers to", by always using the notation $A^{(i)}_j$ 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 and only if we have that $\mathcal{A} \in \mathbb{A}^*_{Q''}$. (See Proposition 5.13 for a justification of this definition.)

PROPOSITION 5.18. *Let $i \in \mathbb{N}_+$. Then (i) The set $\mathbb{A}^{(i)}_{Q''}$ 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 $\mathbb{A}^{(i)}_{Q''}$ captures θ, θ is a t^*_Q-valid assignment mapping for Q'' and $D_{\bar{N}^{(i)}}(Q)$.* □

PROPOSITION 5.19. *Suppose that there exists an $i^* \in \mathbb{N}_+$ such that for some association $\mathcal{A}^{(i^*)}_{j^*} \in \mathbb{A}^{(i^*)}_{Q''}$, a valid assignment mapping for $\mathcal{A}^{(i^*)}_{j^*}$ induces a mapping from all the subgoals of Q'' to a* single *copy-neutral canonical database for query Q.*[8] *Then for* each *$i \in \mathbb{N}_+$, there exists a j, $1 \leq j \leq R^{(i)}_{Q''}$, such that a valid assignment mapping for the association $\mathcal{A}^{(i)}_j$ (exists and) induces a mapping from all the subgoals of the query Q'' to a* single *copy-neutral canonical database for query Q (within database $D_{\bar{N}^{(i)}}(Q)$). Moreover, $\mathcal{A}^{(i)}_j$ and $\mathcal{A}^{(i^*)}_{j^*}$ have the same atom-signature.* □

PROOF. We are given that there exists a pair (i^*, j^*), with $i^* \in \mathbb{N}_+$ and with $1 \leq j^* \leq R^{(i^*)}_{Q''}$, such that the association $\mathcal{A}^{(i^*)}_{j^*}$ generates a valid assignment mapping from query Q'' to a single copy-neutral canonical database, call it D^*, for query Q (within database $D_{\bar{N}^{(i^*)}}(Q)$). By Proposition 5.14 in Section 5.4.2, 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 5.8 in Section 5.3.3, database $D_{\bar{N}^{(i)}}(Q)$ has at least one copy-neutral canonical database for query Q. Choose and fix in $D_{\bar{N}^{(i)}}(Q)$ one arbitrary such copy-neutral canonical database for Q, call this database D. By definition of copy-neutral 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 $\psi^{gen(Q)}_{\bar{N}^{(i)}}[d^*]$ is the same (atom in set $\mathcal{S}_{C(Q)}$) as $\psi^{gen(Q)}_{\bar{N}^{(i)}}[\iota^*(d^*)]$. 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 copy-neutral 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 5.18, the association represented by φ must be in the set $\mathbb{A}^{(i)}_{Q''}$ for database $D_{\bar{N}^{(i)}}(Q)$. By construction, this association and $\mathcal{A}^{(i^*)}_{j^*}$ have the same atom-signature. By the fact that i has been chosen arbitrarily, Q.E.D. □

PROPOSITION 5.20. *Suppose that, for some $i \in \mathbb{N}_+$, there exists an association $\mathcal{A}^{(i)}_j$ in the set $\mathbb{A}^{(i)}_{Q''}$ such that the unity valid assignment mapping, θ_j, for $\mathcal{A}^{(i)}_j$ induces a mapping from the subgoals of Q'' into elements of two or more copy-neutral canonical databases of Q in database $D_{\bar{N}^{(i)}}(Q)$. Then there exists an association $\mathcal{A}^{(i)}_{j'}$ in $\mathbb{A}^{(i)}_{Q''}$, and there exists in $D_{\bar{N}^{(i)}}(Q)$ a copy-neutral canonical database of Q, call this database D^*, such that the unity valid assignment mapping, $\theta_{j'}$, for $\mathcal{A}^{(i)}_{j'}$ induces a mapping from all the subgoals of Q'' into ground atoms belonging to D^* only. Moreover, $\mathcal{A}^{(i)}_j$ and $\mathcal{A}^{(i)}_{j'}$ have the same atom-signature.* □

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

Thus we assume for the remainder of this proof that $m = |M_{noncopy}| \geq 1$. For the association $\mathcal{A}^{(i)}_j$ 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 5.9, the only way the valid assignment mapping θ_j for $\mathcal{A}^{(i)}_j$ can map the subgoals of Q'' into elements of two or more copy-neutral canonical databases of Q in database $D_{\bar{N}^{(i)}}(Q)$ is when the result of intersecting $adom(\mathcal{T})$ with $S^{(i)}_l$, for at least one $l \in \{1, \ldots, m\}$, has size two or more. (For the notation $adom(\mathcal{T})$, see note before Proposition 5.9.)

We show an algorithm for producing the association $\mathcal{A}^{(i)}_{j'}$ and the (copy-neutral canonical) database D^*, of the statement of this Proposition, from the association $\mathcal{A}^{(i)}_j$. First, for each $l \in \{1, \ldots, m\}$ such that the result of intersecting $adom(\mathcal{T})$ with $S^{(i)}_l$ is not empty, fix a single value in that intersection. Let the result be values $v^{(i)}_{l_1}, \ldots, v^{(i)}_{l_k}$, 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 $v^{(i)}_{l_n}$ (for all $1 \leq l_n \leq m$, for all $1 \leq n \leq k$) satisfies $v^{(i)}_{l_n} \in S^{(i)}_{l_n}$. Call all values in the set $E = ((\bigcup_{n=1}^{m} S^{(i)}_n) \bigcap adom(\mathcal{T})) - \{v^{(i)}_{l_1}, \ldots, v^{(i)}_{l_k}\}$ 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 $\mathcal{A}^{(i)}_j$, by replacing in θ_j each value v belonging to $S^{(i)}_n \bigcap E$, $1 \leq n \leq m$, by the value $v^{(i)}_n$ 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 $v^{(i)}_{l_1}, \ldots, v^{(i)}_{l_k}$ fixed in the previous paragraph.) As we modify θ_j this way, we also modify the association $\mathcal{A}^{(i)}_j$, by replacing in it all occurrences of each $v \in S^{(i)}_n \bigcap E$, $1 \leq n \leq m$, by the value $v^{(i)}_n$. Denote by $\theta_{j'}$ the result of this modification of θ_j, and by $\mathcal{A}^{(i)}_{j'}$ the result of this modification of $\mathcal{A}^{(i)}_j$. By construction, we have that:

[8] That copy-neutral canonical database for Q is within database $D_{\bar{N}^{(i^*)}}(Q)$, see Proposition 5.8 in Section 5.3.3.

(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 5.9, we have that there exists in $D_{\bar{N}^{(i)}}(Q)$ a (single) copy-neutral 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 copy-neutral canonical database (in $D_{\bar{N}^{(i)}}(Q)$) for the query Q. □

PROPOSITION 5.21. *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 5.19 and 5.20. 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 copy-neutral canonical database for Q in $D_{\bar{N}^{(i)}}(Q)$. Then, by Proposition 5.19, 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 copy-neutral 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 copy-neutral canonical databases for Q in $D_{\bar{N}^{(i)}}(Q)$. Then, by Proposition 5.20, 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 copy-neutral canonical database for Q in $D_{\bar{N}^{(i)}}(Q)$. Thus we have reduced this case to Case 1. Q.E.D.

□

PROPOSITION 5.22. *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 5.21 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* $i \in \mathbb{N}_+$. □

5.6 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 5.22, $\mathcal{F}_{(Q)}^{(Q'')}$ 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 5.5, we infer from this assumption that the cardinality $R_{Q''}^{(i)}$ of the set $\mathbb{A}_{Q''}^{(i)}$ is a positive integer for each $i \in \mathbb{N}_+$.

5.6.1 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 $\mathcal{C}_1^{(Q'')(i)}$, $\mathcal{C}_2^{(Q'')(i)}$, ..., $\mathcal{C}_{n^{(i)}}^{(Q'')(i)}$, $n^{(i)} \leq R_{Q''}^{(i)}$. 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 5.12 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 5.18, 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 5.4.2 and 5.5:

PROPOSITION 5.23. *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 5.23 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 $\mathcal{C}_1^{(Q'')}$, $\mathcal{C}_2^{(Q'')}$, ..., $\mathcal{C}_{n^*}^{(Q'')}$, 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 $\mathcal{C}_1^{(Q'')}$, $\mathcal{C}_2^{(Q'')}$, ..., $\mathcal{C}_{n^*}^{(Q'')}$) to refer to the contents of the set $\mathcal{C}^{(Q'')}$ w.r.t. the set $\mathbb{A}_{Q''}^{(i)}$ for the fixed i.

5.6.2 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 5.6.3, of "multiplicity monomials" for the monomial classes $\mathcal{C}_1^{(Q'')}$, ..., $\mathcal{C}_{n^*}^{(Q'')}$.

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 5.3.1. 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 5.9. *(**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 5.24. *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 5.25. *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 5.26. *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'', ..., Y_m'' of the query Q'' is the Cartesian product of the sets $\Gamma_{(Y_1'')}^{(i)}[\mathcal{C}^{(Q'')}]$, $\Gamma_{(Y_2'')}^{(i)}[\mathcal{C}^{(Q'')}]$, ..., $\Gamma_{(Y_m'')}^{(i)}[\mathcal{C}^{(Q'')}]$.

□

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 5.27. *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 $\Gamma_{(Y_{m+j}'')}^{(i)}[\mathcal{C}^{(Q'')}]$ is the set $\{1, \ldots, Z^{(i)}\}$, where (a) $Z^{(i)}$ is 1 in case $Z = 1$, and (b) $Z^{(i)}$ is $N_k^{(i)}$ 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}'', ..., Y_{m+r}'' of the query Q'' is the Cartesian product of the sets $\Gamma_{(Y_{m+1}'')}^{(i)}[\mathcal{C}^{(Q'')}]$, $\Gamma_{(Y_{m+2}'')}^{(i)}[\mathcal{C}^{(Q'')}]$, ..., $\Gamma_{(Y_{m+r}'')}^{(i)}[\mathcal{C}^{(Q'')}]$.

□

(The proof is immediate from Proposition 5.17, 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 5.28. *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'', ..., 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''.

□

(Recall from Section 5.2.2 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 5.26. 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 5.27.)

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

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 $\Gamma_{\bar{S}}^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$. 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 5.6.2 for the notation $\Gamma^{(i)}[\mathcal{C}^{(Q'')}]$.) For each monomial class $\mathcal{C}^{(Q'')} \in \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$, 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 5.9 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 (a multiset noncopy variable of Q, i.e.,) 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 5.29 and 5.30, 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 $\Phi_n^{\mathcal{C}^{(Q'')}}$ be the noncopy-signature of the class $\mathcal{C}^{(Q'')}$. Then:

- In case $m \geq 1$, denote the elements of $\Phi_n^{\mathcal{C}^{(Q'')}}$, 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 $\Pi^{(i)}_{\Phi_n^{\mathcal{C}^{(Q'')}}}$ the value 1 in case $m = 0$, and the product $\Pi_{j=1}^m |Dom_Q^{(i)}(Z_j)|$ in case $m \geq 1$.
- Finally, denote by $\Pi_{\Phi_n^{\mathcal{C}^{(Q'')}}}$ the value 1 in case $m = 0$, and the product $\Pi_{j=1}^m DomLabel_Q(Z_j)$ in case $m \geq 1$.

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

- In case $r \geq 1$, denote the elements of $\Phi_c^{\mathcal{C}^{(Q'')}}$, 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 $W_j^{(i)}$ 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 $W_j^{(i)} = N_{m+l}^{(i)}$.) If $W_j = 1$ then let $W_j^{(i)} := 1$.
- For an $i \in \mathbb{N}_+$, denote by $\Pi^{(i)}_{\Phi_c^{\mathcal{C}^{(Q'')}}}$ the value 1 in case $r = 0$, and the product $\Pi_{j=1}^r W_j^{(i)}$ in case $r \geq 1$.
- Finally, denote by $\Pi_{\Phi_c^{\mathcal{C}^{(Q'')}}}$ the value 1 in case $r = 0$, and the product $\Pi_{j=1}^r W_j$ in case $r \geq 1$.

PROPOSITION 5.29. *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* $\Gamma_{\bar{S}}^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$ *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* $\Pi^{(i)}_{\Phi_n^{\mathcal{C}^{(Q'')}}} \times \Pi^{(i)}_{\Phi_c^{\mathcal{C}^{(Q'')}}}$. □

Please see Example 5.3 for an illustration. he proof of Proposition 5.29 is immediate from Proposition 5.28. (See Proposition 5.17 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 5.3.2.)

We note that the expression $\Pi^{(i)}_{\Phi_n^{\mathcal{C}^{(Q'')}}} \times \Pi^{(i)}_{\Phi_c^{\mathcal{C}^{(Q'')}}}$ in Proposition 5.29 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 5.29.

PROPOSITION 5.30. *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* $\Gamma_{\bar{S}}^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$ *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 5.30) $\Pi_{\Phi_n^{\mathcal{C}^{(Q'')}}(\bar{N})} \times \Pi_{\Phi_c^{\mathcal{C}^{(Q'')}}}$, in terms of the variables in the vector $\bar{N}$, *the multiplicity monomial of the monomial class* $\mathcal{C}^{(Q'')}$.

5.7 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 5.33.) The wave of the query Q is defined in this section (see Definition 5.10) based on the vector $\bar{N}$ and on the mapping ν_Q^{copy} defined in Section 5.3.2.

(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 5.34.)

In Section 5.10 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 5.33 and 5.34 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 $\mathcal{P}_{noncopy}^{(Q)}$ (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 $\mathcal{P}_{copy}^{(Q)}$ (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 5.3.2.)

DEFINITION 5.10. *(***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* $\mathcal{P}_*^{(Q)} = \mathcal{P}_{noncopy}^{(Q)} \times \mathcal{P}_{copy}^{(Q)}$. □

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 noncopy 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 5.2.2) 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 5.33. As an illustration of the definition, in Example 5.3 in Section 5.8, the wave of query Q w.r.t. the family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$ is the product $\mathcal{P}_*^{(Q)} = N_1 \times N_2$, where N_1 refers to the only multiset noncopy variable of the query Q, and N_2 refers to its only (multiset) copy variable.

PROPOSITION 5.31. *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 5.31 is immediate from the definition of the products $\mathcal{P}_{noncopy}^{(Q)}$ and $\mathcal{P}_{copy}^{(Q)}$ used in Definition 5.10. (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 $\mathcal{P}_{copy}^{(Q)}$, is immediate from the definition of the mapping ν_Q^{copy}, see Section 5.3.2.)

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 5.32. *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* $\{N_1, N_2, \ldots, N_{m+w}\}$. □

PROPOSITION 5.33. *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[9] $\Phi_n[\mathcal{A}_*^{(i)}]$ 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.

[9]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 $D_{\bar{N}^{(i)}}(Q)$.

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 5.30. 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 5.2.2) for the query Q is not empty.

Fix an $i \in \mathbb{N}_+$. Recall the set $\mathcal{S}^{(i)} \neq \emptyset$ introduced in Section 5.3.2 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 5.3.2 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 a copy-neutral 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.

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 $S^{(i)}_j$. (See Section 5.3.1 for the definition of $S^{(i)}_j$.) Hence, by definition of the vector $\Phi_n[\mathcal{A}^{(i)}_*]$ (see Section 5.4.1), 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 5.33, 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 5.4.1). Q.E.D. □

PROPOSITION 5.34. *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 (nonnegative-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.* □

The proof of Proposition 5.34 is constructive: That is, the proof generates a SCVM from the query Q'' to the query Q of the statement of Proposition 5.34.

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 5.6.3) and of copy-/noncopy-sigature vectors (Section 5.4.1), 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 $[\nu^{copy}_Q(Y_{m+1}) \quad \ldots \quad \nu^{copy}_Q(Y_{m+r})]$.

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 5.5 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 5.14.

By Proposition 5.20, 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 copy-neutral 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 copy-neutral 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 copy-neutral 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 $\nu_Q^{(i)} \circ \nu_{Q''}$ of the mapping $\nu_{Q''}$ with the mapping $\nu_Q^{(i)}$ defined in Section 5.3.1. 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 $\mu'_{Q''}$. Observe that in case $r = 0$, the mapping $\mu_{Q''}$ is fully specified and is unique.

It remains to define $\mu_{Q''}$ on the copy variables of the query Q'', in case, which we refer to as (iii), where $r \geq 1$. In this case, 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 $1 \leq k \leq w$,[10] 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 this fixed 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}, ..., 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 (5) of same-scale covering mappings (SCVMs) in Definition 3.1 are satisfied by the mapping $\mu_{Q''}$. (Hence, we conclude that the mapping $\mu_{Q''}$ is a SCVM 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 $\nu_Q^{(i)}$ 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 $\nu_Q^{(i)}$ 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''; thus, $\mu_{Q''}$ maps the set of multiset noncopy variables of the query Q'' *onto* the (same-cardinality) set of multiset noncopy variables of the query Q; and
- The copy-signature of the association $\mathcal{A}_*$ is (in case $r \geq 1$) a permutation of the list $[\nu_Q^{copy}(Y_{m+1}) \ \ldots \ \nu_Q^{copy}(Y_{m+r})]$, and by definition of the mapping $\mu_{Q''}$ on the set of copy variables of the query Q''; thus, $\mu_{Q''}$ maps the set of copy variables of the query Q'' *onto* the (same-cardinality) 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 the mapping $\mu_{Q''}$ maps each copy-sensitive subgoal of the query Q'' into a *copy-sensitive* subgoal of the query Q.

We conclude that $\mu_{Q''}$ is, by construction, a SCVM from the query Q'' to the query Q. □

5.8 Extended Example: Basic Notation and Constructs

In this section we provide an extended example that illustrates the notions and constructions introduced in Sections 5.2 through 5.7 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. By the results in this paper, for the queries Q and Q' of Example 5.3 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

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

exist a SCVM from the query Q'' to the query Q of Example 5.3. Thus, by Theorem 4.1, $Q \equiv_C Q''$ cannot hold for the queries Q'' and Q of Example 5.3. We build on this example a little later (see Example 5.4 in Section 5.9.3), 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 5.3, we also illustrate the constructs of Section 5.7, by discussing "the wave" of the query Q (see Definition 5.10 in Section 5.7) 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 5.3. *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 Theorem 3.3 and by the existence of a SCVM from Q to Q' and of another SCVM from Q' to Q, we have that $Q \equiv_C Q'$. A SCVM μ from Q' to Q is $\mu = \{X_1' \to X_1, Y_1' \to Y_1, X_2' \to X_2, Y_2' \to Y_2, X_3' \to X_2\}$.

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 5.4 in Section 5.9.3) 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 5.2 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 5.3 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 $\nu_0 = \{X_1 \to a, X_2 \to b\}$, 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 5.3, that:

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

For the set $\mathcal{S}^{(i)} = \{(c), (d)\}$, we have that $\nu_{(c)} = \{X_1 \to a, X_2 \to b, Y_1 \to c, Y_2 \to 3\}$ and that $\nu_{(d)} = \{X_1 \to a, X_2 \to b, Y_1 \to d, Y_2 \to 3\}$. 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 5.4 through 5.7 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 5.4.1: The mapping $\psi_{\bar{N}^{(i)}}^{gen(Q)}$*, as induced by the mapping* $\nu_Q^{(i)}$*, is defined as* $\psi_{\bar{N}^{(i)}}^{gen(Q)} = \{d_1 \to h_1, d_2 \to h_2, d_3 \to h_1\}$*. 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* $\theta_8 = \{X_1'' \to a, Y_1'' \to d, Y_2'' \to 1, X_2'' \to b\}$ *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 5.5. 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 5.18 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 5.6. 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 5.26 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 5.26 for the details.)*

For the construction of the function $\mathcal{F}_{(Q)}^{(Q'')}$ *from the above multiplicity monomials, please see Example 5.4 (Section 5.9.3).*

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

We now follow Sections 5.4 through 5.7 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 5.4 (Section 5.9.3).*

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 5.4 (Section 5.9.3) 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 5.7. By Definition 5.10, 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 5.33, 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 5.34, 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 5.34.*

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 5.34 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 5.4 (Section 5.9.3) for a discussion of how the database* $D_{\bar{N}^{(i)}}(Q)$ *constructed earlier (in Example 5.3) 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 5.4 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.* □

5.9 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 5.6. The reason that we construct this function in this proof of Theorem 4.1 is that we want to be able to pinpoint those queries Q'' that are associated with the wave (see Definition 5.10) of the query Q. That is, later in this proof of Theorem 4.1 (specifically in Section 5.10), for the query Q' specified in the statement of the Theorem we will use the facts that (i) $Q' \equiv_C Q$ (and thus their respective functions $\mathcal{F}_{(Q)}^{(Q')}$ and $\mathcal{F}_{(Q)}^{(Q)}$ must return the same value on each database) and that (ii) Q is an explicit-wave query, to infer that the function $\mathcal{F}_{(Q)}^{(Q')}$ for the query Q' must have as its component the wave monomial of the query Q. The claim of Theorem 4.1 will then follow from Proposition 5.34.

The only entities that we use in this section to specify the function $\mathcal{F}_{(Q)}^{(Q'')}$ are (a) the multiplicity monomials defined in Section 5.6, and (b) the noncopy-signatures and the copy-signatures of the monomial classes introduced in Section 5.6. 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 $\bar{N}$.

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 5.3), and for an arbitrary CCQ query Q'' that satisfies the restrictions (w.r.t. the query Q) of Section 5.2.

5.9.1 Notation, Definitions, Basic Results

For CCQ queries Q and Q'' satisfying the requirements of Section 5.2, suppose that Q and Q'' are also such that (as discussed in the beginning of Section 5.6) 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 that in Section 5.6, we denoted 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 proposition is immediate from Proposition 5.26, for the cases where $n^* \geq 2$.

PROPOSITION 5.35. *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 5.11. *(**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 5.6.

PROPOSITION 5.36. *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 5.11.

PROPOSITION 5.37. *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 5.36 and 5.37, we obtain the following result.

PROPOSITION 5.38. *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 5.3 in Section 5.8, monomial class $\mathcal{C}_1^{(Q'')}$ unconditionally dominates a *nonidentical* [11] (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 $\{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_{n^*}^{(Q'')}\}$.

[11] Recall (see Section 5.6) that the identity of a monomial class is determined by its atom-signature.

A specific algorithm instantiating the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES is obtained by specifying the algorithm CHOOSE-REPRESENTATIVE-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 5.3 in Section 5.8, the algorithm template could produce four different outputs.) At the same time, the following two results, Proposition 5.39 and Proposition 5.40, hold regardless of the choice of the algorithm CHOOSE-REPRESENTATIVE-ELEMENT when instantiating the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES.

PROPOSITION 5.39. *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 5.39 is immediate from Proposition 5.38 and from the construction of the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES.

In the next result, Proposition 5.40, we denote by $|S|$ the cardinality of set S. Proposition 5.40 holds by construction of the algorithm template REMOVAL OF DUPLICATE MONOMIAL CLASSES.

PROPOSITION 5.40. *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 5.39 and 5.40 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 5.11 is reflexive, antisymmetric, and transitive on the set $\mathbb{C}(Q'')$. It follows that the unconditional-dominance relation is a partial order on that set.

We now use any standard algorithm[12] 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* $\mathbb{C}(Q'')$.

Using Propositions 5.39 and 5.40, it is straightforward to show the following.

PROPOSITION 5.41. *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 5.41, 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.

5.9.2 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 5.42 is immediate from Propositions 5.35, 5.39, and 5.40. For the definition of multiplicity monomial for monomial class, see Section 5.6.3.

PROPOSITION 5.42. *Given the set $\mathbb{C}^{nondom}(Q'')$ for a CCQ query Q'' and for a family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$, such*

[12] We use the observation that the unconditional-dominance relation of Definition 5.11 is a partial order on the set $\mathbb{C}(Q'')$.

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 $\Gamma_{\bar{S}}^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$, can be computed exactly, 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

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

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

That is, under the conditions of Proposition 5.42, 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 5.3 in Section 5.8, 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 5.3, $\mathcal{F}_{(Q)}^{(Q'')}$ 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, please see Example 5.4 in Section 5.9.3.)

COROLLARY 5.1. *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 $\Gamma_{\bar{S}}^{(t^*_Q)}(Q'', D_{\bar{N}^{(i)}}(Q))$, can be computed exactly, 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*

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

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

PROOF. (sketch) The reason that Corollary 5.1 of Proposition 5.42 holds is that in case $r \leq 1$, either $r = 0$ holds and then the copy-signature of each monomial class for Q'' is the 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 5.11 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 5.1 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),[13] 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.

[13]By Proposition 5.4, $r \geq 1$ implies $w \geq 1$, hence from $r \geq 2$ and $w \leq 1$ we have the exact equality $w = 1$.

5.9.3 Illustration

In this subsection we build on Example 5.3 (of Section 5.8), 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 5.3 and for the database constructed in Example 5.3. We also show how that database is a counterexample to $Q \equiv_C Q''$, for the queries Q and Q'' of Example 5.3. Finally, we continue our discussion (started in Example 5.3) 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 $\mathcal{F}_{(Q)}^{(Q'')}$.

EXAMPLE 5.4. *Recall the queries Q, Q', and Q'' of Example 5.3 (of Section 5.8). Recall also the database $D_{\bar{N}^{(i)}}(Q)$ that we constructed in Example 5.3 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 5.3) 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 5.3 with four monomial classes $\mathcal{C}_1^{(Q)}$, $\mathcal{C}_2^{(Q)}$, $\mathcal{C}_3^{(Q)}$, and $\mathcal{C}_4^{(Q)}$. By Definition 5.11, monomial class $\mathcal{C}_2^{(Q)}$ unconditionally-dominates monomial class $\mathcal{C}_1^{(Q)}$. The reason is, $\mathcal{C}_1^{(Q)}$ and $\mathcal{C}_2^{(Q)}$ have identical noncopy-signatures, the copy-signature of the monomial class $\mathcal{C}_1^{(Q)}$ is [1], and the copy-signature of the monomial class $\mathcal{C}_2^{(Q)}$ is $[N_2]$. (See Section 5.9.1 for further details on unconditional-dominance.) Similarly, monomial class $\mathcal{C}_4^{(Q)}$ unconditionally-dominates monomial class $\mathcal{C}_3^{(Q)}$. Thus, the set $\{\mathcal{C}_2^{(Q)}, \mathcal{C}_4^{(Q)}\}$ is the set $\mathbb{C}^{nondom}(Q)$ as defined in Section 5.9.1.

Then, by Proposition 5.42, 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 5.3, $\mathcal{F}_{(Q)}^{(Q)}$ 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 5.3, 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 5.3 with four monomial classes $\mathcal{C}_1^{(Q'')}$, $\mathcal{C}_2^{(Q'')}$, $\mathcal{C}_3^{(Q'')}$, and $\mathcal{C}_4^{(Q'')}$. By Definition 5.11, monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ unconditionally-dominate each other. (The reason is, $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ have identical noncopy-signatures,

and have identical copy-signatures.) Similarly, monomial classes $\mathcal{C}_3^{(Q'')}$ and $\mathcal{C}_4^{(Q'')}$ unconditionally-dominate each other. Suppose that we choose the set $\{\mathcal{C}_1^{(Q'')}, \mathcal{C}_3^{(Q'')}\}$ as the set $\mathbb{C}^{nondom}(Q'')$ as defined in Section 5.9.1. (See Section 5.9.1 for the discussion of possible choices for the set $\mathbb{C}^{nondom}(Q'')$.)

Then, by Proposition 5.42, 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 5.3, $\mathcal{F}_{(Q)}^{(Q'')}$ 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 $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$ and $\Gamma_{\bar{S}}^{t_Q^*}(Q'', D_{\bar{N}^{(i)}}(Q))$ 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 5.3 (Section 5.8) our discussion of "the wave of" the query Q of that example. By Definition 5.10, 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'' discussed in this example (see Example 5.3 for their definitions), 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. $\{D_{\bar{N}^{(i)}}(Q)\}$. □

5.9.4 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 5.9.2, specifically using Proposition 5.42. This example motivates the development, in Section 5.9.5, of a more general (as compared to that of Section 5.9.2) approach toward constructing the function $\mathcal{F}_{(Q)}^{(Q)}$ for CCQ query Q'' and family of databases $\{D_{\bar{N}^{(i)}}(Q)\}$.

EXAMPLE 5.5. *Let CCQ query Q be as follows.*

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

(This is the query Q of Example 4.1, rendered here using somewhat different notation.)

Toward Constructing Function $\mathcal{F}_{(Q)}^{(Q)}$ for the Databases $\{ D_{\bar{N}^{(i)}}(Q) \}$.

We show here 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. (We will complete the construction in Example 5.6 in Section 5.9.6.)

We begin by following Section 5.3 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 $S_1^{(i)} = \{e\}$, and let $S_2^{(i)} = \{f, g\}$. 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 $\psi_{\bar{N}^{(i)}}^{gen(Q)}[d_1] = \psi_{\bar{N}^{(i)}}^{gen(Q)}[d_2] = h_1$, and that $\psi_{\bar{N}^{(i)}}^{gen(Q)}[d_3] = \psi_{\bar{N}^{(i)}}^{gen(Q)}[d_4] = h_2$.

We now follow Sections 5.4 through 5.9.1 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,[14] we obtain four monomial classes for the query Q:

- *Monomial class $\mathcal{C}_1^{(Q)}$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_3\ N_3]$; it contributes to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$, 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 $(a, e, f, 1, 1)$, $(a, e, f, 1, 2)$, $(a, e, f, 1, 3)$, $(a, e, f, 2, 1)$, ..., $(a, e, f, 3, 2)$, $(a, e, f, 3, 3)$), as well as nine tuples $(a, e, g, 1, 1)$ through $(a, e, g, 3, 3)$.*

- *Monomial class $\mathcal{C}_2^{(Q)}$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_3\ N_4]$; it contributes to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$ 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 $\mathcal{C}_3^{(Q)}$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_4\ N_3]$; it contributes to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$ 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 $\mathcal{C}_4^{(Q)}$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_4\ N_4]$; it contributes to the set $\Gamma_{\bar{S}}^{t_Q^*}(Q, D_{\bar{N}^{(i)}}(Q))$ 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 5.11) any other monomial class in the set $\{\mathcal{C}_1^{(Q)}, \mathcal{C}_2^{(Q)}, \mathcal{C}_3^{(Q)}, \mathcal{C}_4^{(Q)}\}$. □

5.9.5 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 (easy) case considered in Section 5.9.2. 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

[14] These steps are outlined in significant detail in Example 5.3, albeit using queries that are different from the queries of the current example.

subsection is applicable to constructing the function $\mathcal{F}^{(Q'')}_{(Q)}$ for *all* cases, including the special case of Section 5.9.2.

Consider Example 5.5 of Section 5.9.4: 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 5.11 holds for any pair of monomial classes in the set $\mathcal{C}(Q)$. Hence the set $\mathbb{C}^{nondom}(Q)$ (see Section 5.9.1) is the set $\mathcal{C}(Q)$. Further, as the set $\mathbb{C}^{nondom}(Q)$ does not satisfy the conditions of Proposition 5.42, the function $\mathcal{F}^{(Q)}_{(Q)}$ for the example *cannot* be constructed using Proposition 5.42. Indeed, it is easy to see that $\mathcal{F}^{(Q)}_{(Q)}$ for Example 5.5 is *not* the sum of the multiplicity monomials for the elements of the above set $\mathbb{C}^{nondom}(Q)$. Specifically, Example 5.5 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 5.25 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 5.35, 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 5.43 and 5.44. Example 5.6 in Section 5.9.6 provides an illustration of the construction of the function $\mathcal{F}^{(Q)}_{(Q)}$ for the query Q of Example 5.5 in Section 5.9.4.

PROPOSITION 5.43. *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 $1 \le j_1 < j_2 < j_3 \le n^*$, 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 $1 \le k_1 < k_2 < \ldots < k_n = n^*$ be such that for each $j \in \{1, 2, \ldots, n\}$, all monomial classes $\mathcal{C}^{(Q'')}_{k_{j-1}+1}$, $\mathcal{C}^{(Q'')}_{k_{j-1}+2}$, ..., $\mathcal{C}^{(Q'')}_{k_j}$ have the same noncopy signature.*[15]

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 the $(k_j + l)$'th element of the set $\mathcal{C}[Q''] = \{\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}\}$, under a fixed ordering of the elements of the set as specified in the beginning of the statement of this result.) □

(As usual, we denote by $|S|$ the cardinality of the set S. The result of Proposition 5.43 is immediate from Propositions 5.25 and 5.35.)

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

$$\Big| \bigcup_{l=1}^{(k_{j+1})-(k_j)} \Gamma^{(i)}[\mathcal{C}^{(Q'')}_{(k_j)+l}] \Big|, \tag{1}$$

as introduced in Proposition 5.43. 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 5.44, this special case is captured correctly – as expected – by Proposition 5.30.)

Thus, our next result, Proposition 5.44, is the final missing link in the construction of the function $\mathcal{F}^{(Q'')}_{(Q)}$, as Proposition 5.44 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 5.44 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 5.44, 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 5.43) for exactly these k elements of the set $\mathcal{C}[Q'']$. To make easier the notation in the formal results to

[15] 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 $\mathcal{C}^{(Q'')}_1, \ldots, \mathcal{C}^{(Q'')}_{n^*}$.

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 $\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_k^{(Q'')}$ of the set $\mathcal{C}[Q'']$.)

Now let us refer to the vector Ξ as $\Phi_n^{\mathcal{C}_1^{(Q'')}}$. (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 $\mathcal{C}_1^{(Q'')}$ is exactly Ξ.) The reason that we want to refer to the vector Ξ as $\Phi_n^{\mathcal{C}_1^{(Q'')}}$ is that we want, in the formal results to follow, to use the notation $\Pi_{\Phi_n^{\mathcal{C}_1^{(Q'')}}}$ introduced in Section 5.6.3.

We also use the following notation of Section 5.9.1: 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 5.6 in Section 5.9.6 provides an illustration of the construction of the function $\mathcal{F}_{(Q)}^{(Q)}$ for the query Q and for the database of Example 5.5 in Section 5.9.4.

PROPOSITION 5.44. *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 5.11. Therefore, the result of Proposition 5.44 is immediate from Proposition 5.36.

For the case where $r \geq 1$, the result of Proposition 5.44 is immediate from Lemma 5.1. □

To formulate Lemma 5.1, 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, which is the empty tuple when $m = 0$.

LEMMA 5.1. *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 5.1 is immediate from Proposition 5.27, 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 5.28, 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 5.1. For the fixed $i \in \mathbb{N}_+$

and for each $u \in \{1, \ldots, r\}$, denote by Z_u the value $min(V^{(i)}_{ju[1]}, V^{(i)}_{ju[2]}, \ldots, V^{(i)}_{ju[k]})$. 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}, ..., 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}, ..., 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 5.9.1. 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 5.43):

PROPOSITION 5.45. *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 $\mathcal{C}_{k_{j-1}+1}^{(Q'')}$, $\mathcal{C}_{k_{j-1}+2}^{(Q'')}$, ..., $\mathcal{C}_{k_j}^{(Q'')}$ have the same noncopy signature.*[16]

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 $\mathbb{C}^{nondom}(Q'') = \{\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_p^{(Q'')}\}$.) □

[16]Similarly to the condition of Proposition 5.43, 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 $\mathcal{C}_1^{(Q'')}, \ldots, \mathcal{C}_p^{(Q'')}$.

Observe that Proposition 5.42, which constructs the function $\mathcal{F}^{(Q'')}_{(Q)}$ for a special "easy" case as considered in Section 5.9.2, is an immediate corollary of Proposition 5.45 and of the definition of the set $\mathbb{C}^{nondom}(Q'')$.

For an illustration, consider again the function $\mathcal{F}^{(Q)}_{(Q)}$ of Example 5.4 in Section 5.9.3. 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.

5.9.6 *Illustration of the general construction*

In this subsection, we provide an illustration of the results of Section 5.9.5, by following the construction of the function $\mathcal{F}^{(Q)}_{(Q)}$ for the query Q and for the database of Example 5.5 in Section 5.9.4. As discussed in the beginning of Section 5.9.5, the construction cannot be carried out correctly when using just the results of the "easy-case" Section 5.9.2.

EXAMPLE 5.6. *We refer to the query Q and database $D_{\bar{N}^{(i)}}(Q)$ of Example 5.5 in Section 5.9.4. 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 5.5 in the combined-semantics answer to the query Q on the specific database $D_{\bar{N}^{(i)}}(Q)$ of Example 5.5.*

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 5.5 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 5.43, 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

$$\Big| \bigcup_{l=1}^{4} \Gamma^{(i)}[\mathcal{C}_l^{(Q)}] \Big| .$$

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 5.44 to obtain the cardinality of each of the set intersections in the right-hand side of this formula. First, observe that the multipliers $\Pi_{\Phi_n^{l}}^{\mathcal{C}^{(Q)}}$*, 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 5.44 for all the above set intersections, for all k *between 2 (for* $|A \cap B|, |A \cap C|, \ldots, |C \cap D|$*) and 4 (for* $|A \cap B \cap C \cap D|$*). (For convenience in the statement of Proposition 5.44, 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 5.43, 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 5.43 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 5.44, 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 5.43 and 5.44,

$$\mathcal{F}_{(Q)}^{(Q)} = N_1N_2 \times [(N_3)^2 + 2N_3N_4 + (N_4)^2 - 2ZN_3 - 2ZN_4 + Z^2].$$

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*

$$-2N_1 \times N_2 \times min(N_3, N_4) \times N_3,$$

$$-2N_1 \times N_2 \times min(N_3, N_4) \times N_4, \text{ and}$$

$$+N_1 \times N_2 \times (min(N_3, N_4))^2,$$

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 5.5, 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 5.5 in the combined-semantics answer to the query* Q *on the specific database* $D_{\bar{N}^{(i)}}(Q)$ *of Example 5.5. (Please refer to Example 5.5 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.)* □

5.9.7 Function for the query Q' of Example 4.1

[[[**Insert manually here the ID of Example 4.1**]]]

In this section we illustrate by example that, when the condition (i) of Theorem 4.1 (i.e., the condition of Q being an explicit-wave query) is not satisfied, then even in case where $Q \equiv_C Q'$ does hold, there does not have to exist a monomial class for the query Q' such that the multiplicity monomial of that class is the wave of the query Q.

EXAMPLE 5.7. *Let CCQ query Q' be as follows.*

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

(This is the query Q' of Example 4.1, rendered here using somewhat different notation.)

Toward Constructing the Function $\mathcal{F}_{(Q)}^{(Q')}$ for the Databases $\{ D_{\bar{N}^{(i)}}(Q) \}$ of Example 5.5.

We briefly outline here how we construct the function $\mathcal{F}_{(Q)}^{(Q')}$ for the databases $\{ D_{\bar{N}^{(i)}}(Q) \}$, which (databases) we constructed in Example 5.5 of Section 5.9.4. The two monomial classes for the function $\mathcal{F}_{(Q)}^{(Q')}$, for the query Q' and for the databases $D_{\bar{N}^{(i)}}(Q)$, are as follows:

- *Monomial class $\mathcal{C}_1^{(Q')}$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_3\ N_3]$; intuitively, this monomial class results from mapping the query Q' in all possible ways into those atoms of databases $D_{\bar{N}^{(i)}}(Q)$ that (atoms) are associated with the copy variable of the first subgoal of the query Q (in the specific database $D_{\bar{N}^{(i)}}(Q)$ of Example 5.5, these atoms would be ground atoms d_1 an d_2 – recall that the copy number in each of those atoms is $3 = N_3^{(i)}$);*

  and

- *Monomial class $\mathcal{C}_2^{(Q')}$ has noncopy-signature $[Y_1\ Y_2]$ and copy-signature $[N_4\ N_4]$; intuitively, this monomial class results from mapping the query Q' in all possible ways into those atoms of databases $D_{\bar{N}^{(i)}}(Q)$ that (atoms) are associated with the copy variable of the* second *subgoal of the query Q (in the specific database $D_{\bar{N}^{(i)}}(Q)$ of Example 5.5, these atoms would be ground atoms d_3 an d_4).*

There are no other monomial classes for the query Q' and for the databases $D_{\bar{N}^{(i)}}(Q)$. The reason is, both subgoals of the query Q' have the same relational template. (Specifically, unlike the subgoals of the query Q, the two subgoals of the query Q' use the same set variable X_2.) Thus, the two subgoals of the query Q' can only be mapped into one ground atom at a time. At the same time, each database $D_{\bar{N}^{(i)}}(Q)$ always has at least two "relational templates" for its ground atoms. (To gain the intuition, recall that by construction of the databases $D_{\bar{N}^{(i)}}(Q)$, the two set variables X_2 and X_3 used in the two subgoals of the query Q are mapped in the databases into two distinct fixed values.)

As a result, we obtain in a process that is similar to that of Example 5.6 (in Section 5.9.6) that the closed-form expression function $\mathcal{F}_{(Q)}^{(Q')}$ is

$$\mathcal{F}_{(Q)}^{(Q')} = N_1 \times N_2 \times (N_3^2 + N_4^2 - (min(N_3, N_4))^2) .$$

This is the expression for the multiplicity of the tuple $t_Q^* = (a)$ in the answer to the query Q' *on the databases $D_{\bar{N}^{(i)}}(Q)$* constructed for the query Q*, hence the expression is in terms of the elements of the vector $\bar{N}$* for the query Q*. We can obtain a more compact expression for $\mathcal{F}_{(Q)}^{(Q')}$, which is*

$$\mathcal{F}_{(Q)}^{(Q')} = N_1 \times N_2 \times max(N_3, N_4))^2 .$$

Not surprisingly, the latter expression is identical to what we can obtain for $\mathcal{F}_{(Q)}^{(Q)}$ (for the query Q), see the end of Example 5.6 in Section 5.9.6. (Recall that we have proved that the two queries Q and Q' are combined-semantics equivalent.)

Observe that while $Q \equiv_C Q'$ has been proved to hold, neither monomial class for the query Q' on the databases for the query Q is associated with the wave monomial $\Pi_{j=1}^4 N_j$ of the query Q. Indeed, based on the intuition that we discussed for each monomial class of Q' earlier in this example, we cannot map the two subgoals of the query Q' onto *both subgoals of the query Q. (Such a mapping would be the only possibility for a SCVM from Q' to Q, as a SCVM must map each copy variable of Q' into a distinct copy variable of Q.) The reason that we cannot find such a mapping is simple: The query Q' has the same relational template for both subgoals (note the same set variable in both subgoals of Q'), and hence these subgoals are only mappable into one subgoal of Q at a time.*

Finally, the intuition discussed in the preceding paragraph can also be used to understand why there exists a SCVM from Q to Q' (as opposed to from Q' to Q). Indeed, as Q has different set variables in its two subgoals, it is easy to see that the condition of Q can be mapped onto *the condition of Q' – as a result, we can construct a SCVM from Q to Q'.* □

5.10 For the Q and Q' such that $Q \equiv_C Q'$, When Does Q' Have the Wave of Q?

In Section 5.9 we learned how to construct, for CCQ queries Q and Q'' as specified in Section 5.2.2, 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 5.47, 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 5.10). We show the result of Proposition 5.47 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 5.47 and from Propositions 5.33 and 5.34 of Section 5.7.

Example 5.7 of Section 5.9.7 illustrates that, when 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 even in case $Q \equiv_C Q'$.

We begin the technical exposition by stating a useful auxiliary result in Section 5.10.1.

5.10.1 Equivalence of Multivariate Polynomials

PROPOSITION 5.46. *For a positive integer n, let X_1, X_2, ..., 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.[17] *Let each of $\mathcal{P}_1$ and $\mathcal{P}_2$ be a finite-degree multivariate polynomial in terms of the variables X_1, ..., 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*[18] *$i \in \{1, \ldots, n\}$, the term has the same integer coefficient in $\mathcal{P}_1$ and $\mathcal{P}_2$.* □

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. □

5.10.2 When the Query Q' Has the Wave of Q

We now state and prove the main result of Section 5.10.

PROPOSITION 5.47. *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. □

The proof of Proposition 5.47, to be given in Section 5.10.9, hinges on several results, which we now proceed to introduce. 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.

[17] 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 $\mathbb{Z}$.

[18] 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.

5.10.3 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 5.6 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 5.9.2.) Hence we proceed to prove the above claim for the case $r \geq 2$. Recall from Proposition 5.4(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} ... 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 5.44 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 5.9.5, 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 5.4(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 $\bar{N}_w = [\ 1 \quad N_{m+1} \quad N_{m+2} \quad \ldots \quad N_{m+w}\ ]$. 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 $\bar{K}_w = [\ 1 \quad K_1 \quad K_2 \quad \ldots \quad K_w\ ]$ 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 ... K_w is a permutation of the sequence N_{m+1} N_{m+2} ... 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 $j \in \{2, \ldots, w+1\}$, suppose that the jth element of the vector $\bar{K}_w$ (that is, K_{j-1} in our notation for the vector $\bar{K}_w$) is the 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 5.8. *For* $m = 1$ *and for* $w = 3$*, the vector* $\bar{N}_w$ *is* $\bar{N}_w = [\ 1 \quad N_2 \quad N_3 \quad N_4\ ]$*. Let* $\bar{K}_w := [\ 1 \quad N_3 \quad N_4 \quad N_2\ ]$*. Let* $\bar{N}^{(i)}$*, for some fixed natural number* i*, be* $[\ 5 \quad 3 \quad 7 \quad 6\ ]$*. 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 5.9. *For the vectors* $\bar{K}_w$ *and* $\bar{N}^{(i)}$ *of Example 5.8, 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 $\bar{K}'_w := [\ 1 \quad N_2 \quad N_4 \quad N_3\ ]$*. 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 5.3.1), 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

$$\mathcal{N}^{(\bar{K}_w)} = \{\bar{N}^{(i)} \in \mathcal{N} \mid \bar{N}^{(i)} \text{ agrees with the total order } \mathcal{O}^{(\bar{K}_w)}\}.$$

(Note that in the case $w = 1$, there is only one possible vector $\bar{K}_w = [\ 1 \quad N_{m+1}\ ]$. 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 5.48. *Given a CCQ query* Q *for which* $r \geq 2$*, with vector* $\bar{N}$ *for the query* Q *constructed as defined in Section 5.3.1. 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 5.48 that the set $\mathcal{N}$ is a union of the infinite-cardinality sets $\mathcal{N}^{(\bar{K}_w)}$ ranging over all possible vectors $\bar{K}_w$ for the vector $\bar{N}$ (via the vector $\bar{N}_w$).

PROPOSITION 5.49. *Given a CCQ query* Q *for which* $r \geq 2$*, with vector* $\bar{N}$ *for the query* Q *constructed as defined in Section 5.3.1; and given a CCQ query* Q'' *satisfying the restrictions of Section 5.2.2 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 5.49 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 5.44 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 $\mathcal{F}_{(Q)}^{(Q'')}$. The end of Example 5.6 (in Section 5.9.6) 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 $\mathcal{F}_{(Q)}^{(Q'')}[\bar{K}_w]$; we will use similar notation for the respective functions for the queries Q and Q'. (See Proposition 5.49 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 5.9.2 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 $\bar{N}_w = [\ 1\ \ N_{m+1}\ ]$. 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 $r \leq 1$.)

5.10.4 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 5.9.2 and 5.10.3) 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 5.50, is immediate from the results of Section 5.9.5 and from Propositions 5.46 and 5.49.

PROPOSITION 5.50. *Given a CCQ query Q, with vector $\bar{N}$ for the query Q constructed as defined in Section 5.3.1; 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 $\mathcal{N}^{(\bar{K}_w)}$.* □

5.10.5 Solid terms and phantom terms of the polynomials for the multiplicity functions

To proceed with the proof of Proposition 5.47 (see Section 5.10.2), 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 5.10. 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 5.6 (in Section 5.9.6), 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 $\mathcal{C}_4^{(Q)}$ whose multiplicity monomial is exactly the term $\mathcal{T}$. (See Example 5.5 in Section 5.9.4 for the details on the monomial class $\mathcal{C}_4^{(Q)}$.)

Now consider an abstract example of a phantom term. (We do not know whether there are CCQ queries whose associated functions $\mathcal{F}$ have phantom terms, hence this example is abstract. However, to prove Theorem 4.1, we have to prove that phantom terms do not arise in certain parts of $\mathcal{F}_{(Q)}^{(Q')}$ for those CCQ queries Q' that are combined-semantics equivalent to explicit-wave CCQ query Q.)

EXAMPLE 5.10. *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 $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$. By $m = 0$, we have that the noncopy signature of each of $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ is the empty vector. Suppose that the copy signature of $\mathcal{C}_1^{(Q'')}$ is $[\ N_1\ N_2\ ]$, and that the copy signature of $\mathcal{C}_2^{(Q'')}$ is $[\ N_2\ N_1\ ]$.*

The noncopy signatures of the two monomial classes are identical. Thus, by our results of Section 5.9.5, 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 $\mathcal{C}_1^{(Q'')}$ (as well as $\mathcal{C}_2^{(Q'')}$) 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. □

5.10.6 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 5.3.1, 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 5.51 and 5.52, 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 5.51. *Given an explicit-wave CCQ query Q, with vector $\bar{N}$ constructed as defined in Section 5.3.1. 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).*

□

We note that *implicit-wave* CCQ queries are not covered by the above result. In fact, the implicit-wave query Q of Example 4.1 does not satisfy Proposition 5.51. (Please see Example 5.6 of Section 5.9.6 for the details on this query.)

The observations of Proposition 5.51 are immediate from the relevant definitions, from the construction of the function $\mathcal{F}^{(Q)}_{(Q)}$, and from the definition of explicit-wave query. (Intuitively, we use the definition of explicit-wave CCQ queries, Definition 4.1, to argue that all the monomial classes in question that have the same noncopy signature, must also have the same copy signature. In fact, Definition 4.1 has been formulated so that Proposition 5.51, and consequently Theorem 4.1, could go through.) Specifically, the second bullet of Proposition 5.51 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 and by the definition of its wave function, Definition 5.10. We then use the results of Section 5.9.2 (specifically of Proposition 5.42) 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 5.51 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 5.52, are about the properties of the function $\mathcal{G}^{(Q)}_{(Q)}$ for explicit-wave queries Q. The results of Proposition 5.52 hold by the relevant definitions and by Proposition 5.33. The intuition for Proposition 5.52 is the same as the intuition for Proposition 5.51.

PROPOSITION 5.52. *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 $\{D_{\bar{N}^{(i)}}(Q)\}$.*

□

5.10.7 Intuition for the proof of Proposition 5.47

In this subsection we provide the intuition for the remainder of this proof of Theorem 4.1. Specifically, we explain how we plan, in the remainder of this proof, to prove Proposition 5.47 of Section 5.10.2.

We have just established (see Proposition 5.52) that for the query Q given in Theorem 4.1, the wave $\mathcal{P}^{(Q)}_{*}$ of the query Q is present as a solid term in the m-covering part of the function $\mathcal{F}^{(Q)}_{(Q)}$. We now make an easy observation that follows trivially from Propositions 5.50 and 5.52:

PROPOSITION 5.53. *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 $\{D_{\bar{N}^{(i)}}(Q)\}$.* □

What remains to be shown, to complete the proof of Proposition 5.47 of Section 5.10.2, is the following claim:

CONJECTURE 5.1. *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 $\{D_{\bar{N}^{(i)}}(Q)\}$.* □

The only difference between the formulations of Proposition 5.53 and of Conjecture 5.1 is that Conjecture 5.1 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 $\bar{K}_w$.

Observe that if Conjecture 5.1 is true, then it implies the result of Proposition 5.47 of Section 5.10.2. That is, proving Conjecture 5.1 would complete immediately the proof of Theorem 4.1, please see Section 5.1.2.

Conjecture 5.1 does turn out to be true. (We will introduce its recasting, in identical wording, as Proposition 5.57 of Section 5.10.9.) In the remainder of this subsection, we provide the intuition for the planned proof.

Our plan of attack for the proof of the result of Conjecture 5.1 is as follows. We obtain that result by contradiction, by assuming that $\mathcal{P}_*^{(Q)}$ is a *phantom* term in the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$, for at least one vector $\bar{K}_w$. Our key point in the proof is that if $\mathcal{P}_*^{(Q)}$ is a phantom term for some fixed vector $\bar{K}_w$, then the expression for $\mathcal{P}_*^{(Q)}$ using a product of *min*-expressions (i.e., in a way that does not factor in any specific orderings $\bar{K}_w$) must include a *min*-expression involving at least two distinct variables among the variables $N_{m+1}, \ldots, N_{m+w}$ of the query Q. (Hence our assumption that $w \geq 2$ would be essential in that part of the remainder of the proof of Theorem 4.1.) It is rather straightforward to obtain from this fact that for some vector $\bar{K}'_w$ different from the fixed vector $\bar{K}_w$, we could get a *different* monomial for the term that "looked like" $\mathcal{P}_*^{(Q)}$ under the ordering $\bar{K}_w$, because we would obtain $\bar{K}'_w$ essentially by swapping the relative order of some two distinct variables N_A and N_B in the vector $\bar{K}_w$. Then, *unless* the latter monomial (i.e., the monomial that results from $\mathcal{P}_*^{(Q)}$ when we replace $\bar{K}_w$ with $\bar{K}'_w$) gets canceled out by other terms in the polynomial resulting from $\mathcal{F}_{(Q)}^{(Q')}$ under $\bar{K}'_w$, we can achieve the desired contradiction by observing that the overall m-covering parts of the tuple-multiplicity function for Q' are different under the vectors $\bar{K}_w$ and $\bar{K}'_w$. (The contradiction follows immediately from the results of Section 5.10.6 and from Proposition 5.50, which together say that for each pair ($\bar{K}_w$, $\bar{K}'_w$), the m-covering part of the tuple-multiplicity function for the query Q' under $\bar{K}_w$ and the m-covering part of the tuple-multiplicity function for Q' under $\bar{K}'_w$ must be identical polynomials.)

However, it is not clear at all how to prove the claim that the requisite terms in the functions for Q' do not get canceled out in a way that is "convenient for the function Q'." Here is the source of the difficulty: Observe that if $\mathcal{P}_*^{(Q)}$ is indeed a phantom term in the tuple-multiplicity polynomial for some fixed $\bar{K}_w$, then it must be that $\mathcal{P}_*^{(Q)}$ under that $\bar{K}_w$ is a term in a polynomial, $\mathcal{R}$, such that $\mathcal{R}$ arises from the inclusion-exclusion principle for counting the cardinality of a union of sets of tuples for some monomial classes for the query Q' and for databases $D_{N^{(i)}}(Q)$. Further, $\mathcal{P}_*^{(Q)}$ under that $\bar{K}_w$ being a *phantom* term of that polynomial $\mathcal{R}$ means that:

(*) The polynomial for the union that we are speaking about must be for a union of the sets of tuples for *at least two* such monomial classes.

(This fact is immediate from our assumption that $\mathcal{P}_*^{(Q)}$ is a phantom term, that is, $\mathcal{P}_*^{(Q)}$ cannot be the multiplicity monomial for any monomial class for Q' and for databases $D_{N^{(i)}}(Q)$).

As a result, we have that:

(**) The polynomial $\mathcal{R}$ for the above union must have terms with positive coefficients, as well as terms with negative coefficients.

The fact (**) furnishes a significant challenge in the proof of Conjecture 5.1. Indeed, if $\mathcal{P}_*^{(Q)}$ is a phantom term in such a polynomial for some fixed $\bar{K}_w$, then (by definition of phantom terms) $\mathcal{P}_*^{(Q)}$ must be a phantom term in the relevant polynomials for *all possible* vectors $\bar{K}_w$, and thus the polynomials as in (*) *for all possible vectors* $\bar{K}_w$ must have terms with positive coefficients, as well as terms with negative coefficients. As a result, there is a theoretical possibility, for the query Q', that under each different vector $\bar{K}_w$, $\mathcal{P}_*^{(Q)}$ is a *different* term in some such union-of-multiplicities polynomial. Thus, it is theoretically possible that in the overall polynomial that is the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$, the remaining terms (i.e., those terms that do not "look like" $\mathcal{P}_*^{(Q)}$ under the given vector $\bar{K}_w$) always cancel each other out in a convenient way. This way, we can make no conclusion about the wave $\mathcal{P}_*^{(Q)}$ of the query Q being backed up by assignments from the query Q' to the databases $D_{\bar{N}^{(i)}}(Q)$, and thus can make no conclusion about the result of Theorem 4.1 for (explicit-wave CCQ queries Q and) arbitrary CCQ queries Q'.

Our plan of attack for overcoming this major challenge is in developing a class of signatures, for multivariate polynomials with integer coefficients in the context of the proof of Theorem 4.1, in such a way that the signature for each such polynomial is always a set that features (some combinations of) variables used in those polynomials, always with positive coefficients. (As a result, no part of such a signature can cancel out another part of the signature.) Further, whenever the signatures for two such polynomials are different, then the polynomials must also be different. We explain how to construct the signatures in Section 5.10.8.

Then, in Section 5.10.9, we prove Conjecture 5.1, by essentially following the idea-of-the-proof argument of this subsection, but by looking at the signatures instead of at the polynomials that generate those signatures. This way, we overcome the challenge that arises from the presence of both positive- and negative-coefficient terms under our assumption-toward-contradiction in the proof of Conjecture 5.1, and thus complete the proof of Theorem 4.1.

5.10.8 Signatures for the polynomials $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$

In this subsection, for the multivariate polynomials with integer coefficients in the context of the proof of Theorem 4.1, we define a class of signatures. One property of these signatures is that the signature for each such polynomial is always a set that features (some combinations of) variables used in those polynomials, always with positive coefficients. (As a result, no part of such a signature can cancel out another part of the signature.) Another property of these signatures is that, whenever the signatures for two such polynomials are different, then the polynomials must also be different. We use these signatures in Section 5.10.9, to complete the proof of Theorem 4.1.

The general idea of our proposed signatures is to produce, for a given multivariate polynomial, in terms of the variables N_{m+1} through N_{m+w} and with integer coefficients, the "overall count" of the number of occurrences of each variable in the polynomial, separately for each of the variables. (We count as positive occurrences all such occurrences of variables in terms with positive coefficients, and count as negative occurrences all such occurrences of variables in terms with negative coefficients.) Intuitively, we do the counting on a version of the polynomial where (i) each power of the form X^p, with X a variable of interest and with p an integer value greater than two, is expanded as $X \times X \times \ldots \times X$, with X occurring p times in the product, and where (ii) each term of the form $C \times \Pi$, with Π being a product of variables with

the unity coefficient and with the absolute value $|C|$ of C being strictly greater than unity, is expanded as $sign(C) \times \Pi \times \Pi \times \ldots \times \Pi$, with Π occurring $|C|$ times in the product.

To construct the signatures, we recall the "original" formulations of the m-covering part (only) of the functions $\mathcal{F}_{(Q)}^{(Q'')}$, as given via Propositions 5.43 and 5.44 of Section 5.9.5. For the functions $\mathcal{F}_{(Q)}^{(Q'')}$, these are the formulations that do not take into account any fixed copy-variable-ordering vectors $\bar{K}_w$, and hence the formulations may use *min*-expressions. To construct the signatures in this section, in each such formulation of a function $\mathcal{F}_{(Q)}^{(Q'')}$, we order the elements of each term according to the provenance of each element from the respective copy signature. That is, whenever a *min*-expression, $\mathcal{M}$, in any such term $\mathcal{T}$, is a *min*-expression for some fixed value of u between 1 and r, as given in the notation of Proposition 5.44 of Section 5.9.5, then the *min*-expression $\mathcal{M}$ is exactly the uth element, from left to right, in the term $\mathcal{T}$. (In any special case where the *min*-expression $\mathcal{M}$ evaluates either to the constant 1 or to a single variable name N_j for some jth element, $j \in \{ m+1, \ldots, m+w \}$, of the vector $\bar{N}$, – that is, in all cases where the *min*-expression contains only one element, – the respective uth element in the term $\mathcal{T}$ is exactly the value 1 or that variable name, rather than the (unevaluated) *min*-expression.)

In the expression that results from the procedure described in the previous paragraph, we further group together, by using parentheses, each separate union-of-multiplicity-terms expression as in the format of Proposition 5.43 of Section 5.9.5.

The purpose of the above ordering is to make it drastically easier, as we will see in Proposition 5.56, to compute our proposed signatures. To make it even easier to compute the signatures, in case where $m \geq 1$ we divide each such ordered expression as above by the product $\Pi_{i=1}^{m} N_i$. Thus, the expression that we use to construct a signature is always an (ordered) expression in terms of *only* the variables N_{m+1} through N_{m+w}.

For a fixed CCQ query Q'', the first input to our signature constructor is always the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q'')}$, which we then preprocess by (a) dividing the entire expression by $\Pi_{i=1}^{m} N_i$ in case where $m \geq 1$, and by then (b) ordering, as explained in the third and fourth paragraphs of this subsection, the elements of all the terms in the output of the step (a). (All the "expansions" outlined in the second paragraph of this subsection result naturally from the preprocessing step (b).) The second input to our signature constructor is always some copy-variable-ordering vector $\bar{K}_w$ for the query Q.

In the remainder of this subsection, we denote by $f^{(\mathcal{S})}(Q'')$ the output of our preprocessing steps (a) and (b) when given as input the function $\mathcal{F}_{(Q)}^{(Q'')}$ for a CCQ query Q'' and for a family of databases $D_{\bar{N}^{(i)}}(Q)$.

Consider an illustration.

EXAMPLE 5.11. *Consider the function*

$$2 \times N_1 \times N_2 - (min(N_1, N_2))^2 \qquad (2)$$

that was constructed in Example 5.10 for a hypothetical CCQ query Q''. As $m = 0$ in the context of that example, Eq. (2) defines exactly the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q'')}$. Observe that Eq. (2) contains terms with nonunity coefficients (see the first term in the Equation) and, as we will see in this example, does not observe the ordering that we proposed in the third paragraph of this subsection.

Our preprocessing step (a) is to divide the entire expression by $\Pi_{i=1}^{m} N_i$, in case where $m \geq 1$. As $m = 0$ for this query Q'', the output of this step for the function of Eq. (2) is exactly the input of this step (a).

We now begin the preprocessing step (b), by ordering, as explained in the third paragraph of this subsection, the elements of all the terms in the output of the step (a). Specifically, the first term $2 \times N_1 \times N_2$ in Eq. (2) stands for two separate identical expressions $N_1 \times N_2$, with different meanings, as follows. The first expression $N_1 \times N_2$ is the expression for the cardinality of the set $\bigcap_{s=1}^{1} \Gamma^{(i)}[\mathcal{C}_s^{(Q'')}]$ (see Proposition 5.44 of Section 5.9.5), that is for the cardinality of the intersection with itself of the tuple-multiplicity set for the monomial class $\mathcal{C}_1^{(Q'')}$ of Example 5.10. The copy signature of that monomial class is $[N_1\ N_2]$, hence this first expression is reordered (vacuously) in this step (b) as the product of N_1 and N_2, in this order. (Trivially, when we compute the cardinality of the intersection of a cardinality set with itself, in each product-of-min-expressions term as given by Proposition 5.44 of Section 5.9.5 we have that each min-expression has only one argument, and hence is evaluated immediately to the respective element of the copy signature.)

The meaning of the second copy of the expression $N_1 \times N_2$ in Eq. (2) is different from the meaning of the first copy of that expression. Specifically, the second copy of the expression $N_1 \times N_2$ is the expression for the cardinality of the set $\bigcap_{s=2}^{2} \Gamma^{(i)}[\mathcal{C}_s^{(Q'')}]$ (see Proposition 5.44 of Section 5.9.5), that is for the cardinality of the intersection with itself of the tuple-multiplicity set for the monomial class $\mathcal{C}_2^{(Q'')}$ of Example 5.10. The copy signature of that monomial class is $[N_2\ N_1]$, hence this expression (unlike the first expression $N_1 \times N_2$) is reordered (nontrivially) in this step (b) as the product of N_2 and N_1, in this order.

Now the term "$-(min(N_1, N_2))^2$" of Eq. (2) is reformatted in this preprocessing step (b) as the product of two copies of the expression "$min(N_1, N_2)$," with the negative sign then passed over to the output from the input expression. That is, the step (b) reformats "$-(min(N_1, N_2))^2$" into "$-min(N_1, N_2) \times min(N_1, N_2)$." In this product expression, the leftmost element of the product stands for the minimum between the first element (N_1) of the copy signature of the monomial class $\mathcal{C}_1^{(Q'')}$ and the first element (N_2) of the copy signature of the monomial class $\mathcal{C}_2^{(Q'')}$. Similarly, the rightmost element of the product stands for the minimum between the second element (N_2) of the copy signature of the monomial class $\mathcal{C}_1^{(Q'')}$ and the second element (N_1) of the copy signature of the monomial class $\mathcal{C}_2^{(Q'')}$.

Finally, all three expressions above, that is, "$N_1 \times N_2$," "$N_2 \times N_1$," and "$-min(N_1, N_2) \times min(N_1, N_2)$," all belong to the same cardinality expression for a union of monomial classes. Hence, as explained in the fourth paragraph of this subsection, the output $f^{(\mathcal{S})}(Q'')$ of the step (b) and of the overall preprocessing of this subsection is the single parenthesized expression

$$f^{(\mathcal{S})}(Q'') = (N_1 \times N_2 + N_2 \times N_1 - min(N_1, N_2) \times min(N_1, N_2)) .$$ □

We make the following observation, which holds trivially by construction of the expression $f^{(\mathcal{S})}(Q'')$:

PROPOSITION 5.54. *Given the m-covering part, call it $\mathcal{F}_m^{(Q'')}$, of the function $\mathcal{F}_{(Q)}^{(Q'')}$ for CCQ queries Q and Q''. Denote by $\mathcal{G}_m^{(Q'')}$ the result of dividing the expression $\mathcal{F}_m^{(Q'')}$ by the product $\Pi_{j=1}^{m} N_j$ in case $m \geq 1$; in case where $m = 0$ we denote by $\mathcal{G}_m^{(Q'')}$ the original expression $\mathcal{F}_m^{(Q'')}$. Then for each $i \geq 1$, the expressions $f^{(\mathcal{S})}(Q'')$ and $\mathcal{G}_m^{(Q'')}$ evaluate to the same value when the elements of the vector $\bar{N}^{(i)}$ are substituted into each expression as values for the elements of $\bar{N}$.* □

We now define the signatures, which will be the key element of the proof in Section 5.10.9. Given the function $f^{(\mathcal{S})}(Q'')$ for a CCQ query Q'' and given a copy-variable-ordering vector $\bar{K}_w$ for the vector $\bar{N}$ of the fixed explicit-wave query query Q, we say that *the signature* $\mathcal{S}(Q'', \bar{K}_w)$ *of* $f^{(\mathcal{S})}(Q'')$ *with respect to* $\bar{K}_w$ is the set resulting from (i) evaluating the value of each *min*-expression in $f^{(\mathcal{S})}(Q'')$ using the total ($\leq$) order $\bar{K}_w$ (that is, given a fixed $\bar{K}_w$, each *min*-expression as in Proposition 5.44 of Section 5.9.5 evaluates to a single element of the set { 1, N_{m+1}, …, N_{m+w} }), and then from (ii) counting the number of occurrences of each variable name separately in the resulting expression, taking into account the sign of each term (that is, an occurrence of a variable name in a term with the negative sign is a negative occurrence, whereas an occurrence of a variable name in a term with the positive sign is a positive occurrence). The resulting signature $\mathcal{S}(Q'', \bar{K}_w)$ is a set that has a separate element for each variable name in the set { N_{m+1}, …, N_{m+w} }, giving the total number of occurrences of that variable name in $f^{(\mathcal{S})}(Q'')$ under the vector $\bar{K}_w$; we omit from $\mathcal{S}(Q'', \bar{K}_w)$ the mention of all the variable names whose total occurrence in $f^{(\mathcal{S})}(Q'')$ under $\bar{K}_w$ is zero. For instance, in the context of Example 5.11, the signature $\mathcal{S}(Q'', \bar{K}_w)$ of $f^{(\mathcal{S})}(Q'')$ with respect to $\bar{K}_w$ = [1 N_1 N_2] is $\mathcal{S}(Q'', \bar{K}_w)$ = { $2 \times N_2$ }, as is easy to ascertain using the expression for $f^{(\mathcal{S})}(Q'')$ that we obtained in Example 5.11.

In evaluating $\mathcal{S}(Q'', \bar{K}_w)$, we must take care not to "bundle together" into exponents repeated occurrences of the same variable name; that is, for instance, the product $N_{m+1} \times N_{m+1}$, which is the same as N_{m+1}^2, counts as two occurrences of N_{m+1} rather than one. Further, we do not count any occurrences of the constant 1 in evaluating $\mathcal{S}(Q'', \bar{K}_w)$. The reason is, all we need to complete the proof of Theorem 4.1 is to show that a hypothetical function of the form $f^{(\mathcal{S})}(Q'')$, in the context of two distinct vectors $\bar{K}_w$ and $\bar{K}'_w$, results in two nonidentical multivariate polynomials. We will do this in our proof in Section 5.10.9 by arguing that the signatures of those polynomials are not the same. Clearly, by construction of the signatures we have that:

PROPOSITION 5.55. *Given expressions $f^{(\mathcal{S})}(Q')$ and $f^{(\mathcal{S})}(Q'')$, where Q' and Q'' may or may not be the same CCQ query, and given two distinct vectors $\bar{K}_w$ and $\bar{K}'_w$. Then, whenever the signatures $\mathcal{S}(Q', \bar{K}_w)$ and $\mathcal{S}(Q'', \bar{K}'_w)$ are nonidentical sets, then the polynomial that results from evaluating the value of each min-expression in $f^{(\mathcal{S})}(Q')$ under the total ($\leq$) order $\bar{K}_w$ is not identical to the polynomial that results from evaluating the value of each min-expression in $f^{(\mathcal{S})}(Q'')$ under the total ($\leq$) order $\bar{K}'_w$.* □

In the remainder of this subsection, we make a key observation, Proposition 5.56. The intuition for that result is that it is very easy to compute the signature of each expression $f^{(\mathcal{S})}(Q'')$, w.r.t. each fixed vector $\bar{K}_w$, by first forming a single positive-sign *max*-expression for each position (from 1st to rth) of the copy signatures associated with each parenthesized expression of $f^{(\mathcal{S})}(Q'')$, by then evaluating the resulting *max*-expressions under the total ($\leq$) order $\bar{K}_w$, and by finally summing up all the resulting *positive-sign-only* occurrences of the variable names.

EXAMPLE 5.12. *We continue Example 5.11. In the expression for $f^{(\mathcal{S})}(Q'')$ that we obtained in that Example,*

$$f^{(\mathcal{S})}(Q'') = (N_1 \times N_2 + N_2 \times N_1 - min(N_1, N_2) \times min(N_1, N_2)) .$$

*the "*first *position" in the only parenthesized expression in $f^{(\mathcal{S})}(Q'')$ is the expression*

$$f_1^{(\mathcal{S})}(Q'') = (N_1 + N_2 - min(N_1, N_2)\ ) .$$

Intuitively, that first position in the only parenthesized expression in $f^{(\mathcal{S})}(Q'')$ was obtained by keeping only the first *element of each term in the original parenthesized expression. These correspond to a "union expression" for exactly the first element of each of the two copy signatures ([N_1 N_2] and [N_2 N_1]) associated with this parenthesized expression.*

*Similarly, the "*second *position" in the only parenthesized expression in $f^{(\mathcal{S})}(Q'')$ is the expression*

$$f_2^{(\mathcal{S})}(Q'') = (N_2 + N_1 - min(N_1, N_2)\ ) .$$

Intuitively, that second position in the only parenthesized expression in $f^{(\mathcal{S})}(Q'')$ was obtained by keeping only the second *element of each term in the original parenthesized expression. These correspond to a "union expression" for exactly the second element of each of the two copy signatures ([N_1 N_2] and [N_2 N_1]) associated with this parenthesized expression.* □

Interestingly, a more concise expression for $f_1^{(\mathcal{S})}(Q'')$ in Example 5.12 is $f_1^{(\mathcal{S})}(Q'') = max(N_1, N_2)$. Indeed, the expression given as $f_1^{(\mathcal{S})}(Q'')$ in Example 5.12 is the equivalent expansion of the expression $max(N_1, N_2)$ using the inclusion-exclusion principle for unions of sets. This is exactly the intuition for our promised easy evaluation for signatures: The first position in the expression for $f^{(\mathcal{S})}(Q'')$ in Example 5.11 corresponds to a formula, using the inclusion-exclusion principle, for the cardinality of the union of the sets { 1, …, N_1 } and { 1, …, N_2 }. The former set is exactly the range of the values that are provided by the assignments in the monomial class $\mathcal{C}_1^{(Q'')}$ to the column for the first copy variable of the query Q'' (see Example 5.11 for the first element of the copy signature of that monomial class), when Q'' is evaluated on any of the databases $D_{\bar{N}^{(i)}}(Q)$. Similarly, the latter set, i.e., { 1, …, N_2 }, is exactly the range of the values that are provided by the assignments in the monomial class $\mathcal{C}_2^{(Q'')}$ to the column for the first copy variable of the query Q'' (see Example 5.11 for the first element of the copy signature of that monomial class), when Q'' is evaluated on any of the databases $D_{\bar{N}^{(i)}}(Q)$. That is, the expression for $f_1^{(\mathcal{S})}(Q'')$ in Example 5.12 is an expression for evaluating the cardinality of the union of these two sets; the latter union is exactly the set of values of the first copy

variable of the query Q'', when Q'' is evaluated on any of the databases $D_{\bar{N}^{(i)}}(Q)$ under all the assignments provided by the monomial classes $\mathcal{C}_1^{(Q'')}$ and $\mathcal{C}_2^{(Q'')}$ together. We can obtain a similar intuition for the expression for $f_2^{(\mathcal{S})}(Q'')$ in Example 5.12, that is again as a $max(N_1, N_2)$.

EXAMPLE 5.13. *We continue with Example 5.12. Fix a vector* $\bar{K}_w = \{\ N_1\ N_2\ \}$. *Then the expression for* $f_1^{(\mathcal{S})}(Q'')$ *in Example 5.12 evaluates to* N_2, *and so does the expression for* $f_2^{(\mathcal{S})}(Q'')$ *in Example 5.12. As a result, the set* $\mathcal{S}(Q'', \bar{K}_w)$ *for this vector* $\bar{K}_w$ *is the result of adding up these two positive-sign occurrences of* N_2, *from* $f_1^{(\mathcal{S})}(Q'')[\bar{K}_w] = max(N_1, N_2)[\bar{K}_w] = N_2$ *and similarly from* $f_2^{(\mathcal{S})}(Q'')[\bar{K}_w] = N_2$. *That is, the set* $\mathcal{S}(Q'', \bar{K}_w)$ *for this vector* $\bar{K}_w$ *is* $\mathcal{S}(Q'', \bar{K}_w) = \{\ 2N_2\ \}$. □

We now formalize the above intuition in Proposition 5.56. To formulate Proposition 5.56, we use the following notation: Let the function $f^{(\mathcal{S})}(Q'')$, for a CCQ query Q'', have exactly $q \geq 1$ parenthesized subexpressions as defined by the fourth paragraph of this subsection; we enumerate these parenthesized subexpressions from left to right as the 1st through qth parenthesized subexpression of the function $f^{(\mathcal{S})}(Q'')$. In the kth such parenthesized subexpression, for any k between 1 and q inclusively, let the $l_k \geq 1$ monomial classes involved in the union in this subexpression be $\mathcal{C}_{k1}^{(Q'')}$ through $\mathcal{C}_{kl_k}^{(Q'')}$. Further, for each $p \in \{\ 1, \ldots, l_k\ \}$, let the copy signature of the class $\mathcal{C}_{kp}^{(Q'')}$ be denoted by $[\ V_{kp}^{(1)},\ V_{kp}^{(2)},\ \ldots,\ V_{kp}^{(r)}\ ]$, where each element of the vector is (by definition of copy signature) an element of the set $\{\ 1,\ N_{m+1},\ \ldots,\ N_{m+w}\ \}$. For a fixed $j \in \{\ 1, \ldots, r\ \}$, consider the result of keeping in the kth parenthesized subexpression of $f^{(\mathcal{S})}(Q'')$ only the jth element of each term, with the (positive or negative) sign of the original term kept around. We refer to the resulting expression as *the jth projection of the kth parenthesized subexpression of* $f^{(\mathcal{S})}(Q'')$. (For instance, for the function $f^{(\mathcal{S})}(Q'')$ that we obtained in Example 5.11, the 1st projection of the first (and only) parenthesized subexpression of $f^{(\mathcal{S})}(Q'')$ is the expression $f_1^{(\mathcal{S})}(Q'') = (N_1 + N_2 - min(N_1, N_2))$; see Example 5.12 for the details.)

We are now ready to formulate Proposition 5.56.

PROPOSITION 5.56. *Let the function* $f^{(\mathcal{S})}(Q'')$, *for a CCQ query* Q'', *have* $q \geq 1$ *parenthesized subexpressions. Fix a value* $k \in \{\ 1, \ldots, q\ \}$ *and a value* $j \in \{\ 1, \ldots, r\ \}$. *Denote by* $\mathcal{C}_{k1}^{(Q'')}$ *through* $\mathcal{C}_{kl_k}^{(Q'')}$, *for some* $l_k \geq 1$, *all the monomial classes contributing to the kth parenthesized subexpression of* $f^{(\mathcal{S})}(Q'')$. *Then the jth projection of the kth parenthesized subexpression of* $f^{(\mathcal{S})}(Q'')$ *is equivalent to the expression* $f_{kj}^{(\mathcal{S})}(Q'') = max(V_{k1}^{(j)}, \ldots, V_{kl_k}^{(j)})$, *where each of* $V_{k1}^{(j)}$, $\ldots$, $V_{kl_k}^{(j)}$ *is the jth element of the copy signature of the respective monomial class.* □

The result of Proposition 5.56 is by construction of the union expressions for Proposition 5.43 of Section 5.9.5, as given by the inclusion-exclusion principle for unions of sets and by the *min*-expressions of Proposition 5.44 of Section 5.9.5 for intersections of sets.

Proposition 5.56 gives us immediately a linear-time procedure for constructing the signature $\mathcal{S}(Q'', \bar{K}_w)$ for each function $f^{(\mathcal{S})}(Q'')$ and copy-variable-ordering vector $\bar{K}_w$:

COROLLARY 5.2. *Let the function* $f^{(\mathcal{S})}(Q'')$, *for a CCQ query* Q'', *have* $q \geq 1$ *parenthesized subexpressions. Then for each copy-variable-ordering vector* $\bar{K}_w$, *the signature* $\mathcal{S}(Q'', \bar{K}_w)$ *of* $f^{(\mathcal{S})}(Q'')$ *with respect to* $\bar{K}_w$ *can be constructed by (1) obtaining all the* $q \times r$ *positive-sign max-expressions given by Proposition 5.56, by then (2) evaluating each max-expression under the total* $(\leq)$ *order* $\bar{K}_w$, *and by finally (3) adding up all the positive-sign occurrences of all the variables* N_{m+1}, $\ldots$, N_{m+w}, *separately for each variable, in these results of evaluating the max-expressions under* $\bar{K}_w$. □

The result of Corollary 5.2 is straightforward from Proposition 5.56 and from the following Lemma 5.2.

LEMMA 5.2. *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 5.2 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.)

We end this subsection by observing that all elements of the set $\mathcal{S}(Q', \bar{K}_w)$ have positive coefficients, which add up to exactly $q \times r$. This result is immediate from Proposition 5.56.

COROLLARY 5.3. *Given a CCQ query* Q' *and a copy-variable-ordering vector* $\bar{K}_w$ *as specified in the proof of Theorem 4.1. Denote by* $q \geq 1$ *the total number of parenthesized expressions in the function* $f^{(\mathcal{S})}(Q')$. *Then we have that:*

1. *Each element of the signature* $\mathcal{S}(Q', \bar{K}_w)$ *of* $f^{(\mathcal{S})}(Q')$ *w.r.t.* $\bar{K}_w$ *has a positive coefficient; and*
2. *All the positive coefficients of all the elements of the set* $\mathcal{S}(Q', \bar{K}_w)$ *add up to* $q \times r$.

□

5.10.9 Proof of Proposition 5.57

We are now ready to prove Conjecture 5.1 (of Section 5.10.7), that is the following Proposition 5.57. This proof completes the proof of Proposition 5.47 of Section 5.10.2 (see Section 5.10.7), and hence the proof of Theorem 4.1, as outlined in Section 5.1.2.

PROPOSITION 5.57. *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 5.57, we will use the following observation.

PROPOSITION 5.58. *Consider two CCQ queries* Q *and* Q' *as in the statement of Theorem 4.1. Let* $\bar{K}_w$ *be a copy-variable-ordering vector, for the vector* $\bar{N}$ *of the query* Q, *and let* $\mathcal{P}[\bar{K}_w]$ *be an arbitrary phantom term in the m-covering part of* $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$. *Then for the product-of-min-expressions,* $\mathcal{P}$, *in* $\mathcal{F}_{(Q)}^{(Q')}$, *such that* $\mathcal{P}$ *gives rise to* $\mathcal{P}[\bar{K}_w]$ *under the total* $(\leq)$ *order* $\bar{K}_w$, *we have that for at least one* $j \in \{\ 1, \ldots, r\ \}$, *the min-expression that is the jth projection of* $\mathcal{P}$ *has (the min-expression) at least two distinct arguments from the set* $\{\ 1,\ N_{m+1},\ \ldots,\ N_{m+w}\ \}$. □

(Please see Section 5.10.8 for both the jth-projection terminology and for the idea of product-of-*min*-expressions generating terms of m-covering parts of $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ under the total ($\leq$) orders $\bar{K}_w$.)

PROOF. The proof is by contradiction: Assume that for no j between 1 and r inclusively does the (*min*-expression that is the) jth projection of $\mathcal{P}$ have more than one argument. Then $\mathcal{P}$ must be a solid term, and so must be $\mathcal{P}_*^{(Q)}$ generated from $\mathcal{P}$ under $\bar{K}_w$, a contradiction. (It is immediate from Proposition 5.44 of Section 5.9.5 that, if the *min*-expression that is the jth projection of a term has only one argument, for all j from 1 to r inclusively, then the term must be a multiplicity monomial for some nonempty monomial class for the CCQ query in question.) □

PROOF. (Proposition 5.57) We consider first the special case where Q' is under the jurisdiction of Proposition 5.42 (of Section 5.9.2; our query Q' would be the Q'' in the statement of Proposition 5.42). 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 5.53 we have the desired result of Proposition 5.57 for this case. Observe that this special case includes all cases where $w = 0$.

Note that in all cases where Q' is *not* under the jurisdiction of Proposition 5.42, it holds that we have $r \geq 2$. (Recall Corollary 5.1 of Section 5.9.2; the Corollary outlines a special case of Proposition 5.42 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 5.42 (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 5.53.) Note that it follows immediately from the definitions of solid and phantom terms (see Section 5.10.5) 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.

For the remainder of the proof, we fix any one vector $\bar{K}_w$ such that 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. In addition, we denote by $\mathcal{P}$ the product-of-*min*-expressions in $\mathcal{F}_{(Q)}^{(Q')}$, such that $\mathcal{P}$ gives rise to $\mathcal{P}[\bar{K}_w]$ under the total ($\leq$) order $\bar{K}_w$. (We use here the same terminology as in the statement of Proposition 5.58.)

In the proof by contradiction, we consider separately the special case where $w = 1$, and then the case where $w \geq 2$. (Recall that we dealt with the case $w = 0$ in the first paragraph of the proof.)

(1) Suppose $w = 1$. For the $\mathcal{P}$ fixed as explained above, by Proposition 5.58 there must be a j between 1 and r inclusively, such that the *min*-expression that is the jth projection of $\mathcal{P}$ has (the *min*-expression) at least two distinct arguments from the set $\{ 1, N_{m+1}, \ldots, N_{m+w} \}$. Fix an arbitrary value of j that satisfies this condition. Observe that in case where $w = 1$, the set $\{ 1, N_{m+1}, \ldots, N_{m+w} \}$ has exactly two elements, 1 and N_{m+1}. Thus, the only case where the jth projection of $\mathcal{P}$ can have at least two arguments is when that *min*-expression (which is the jth projection of $\mathcal{P}$) has exactly two arguments, one of them 1 and the other N_{m+1}.

We now recall that for all copy-variable-ordering vectors $\bar{K}_w$, the value of N_{m+1} is always greater than or equal to 1. Thus, any *min*-expression as above would always evaluate immediately to 1 in $\mathcal{P}$. With this conclusion, we arrive at the desired contradiction with the assumption that under the fixed vector $\bar{K}_w$, the term $\mathcal{P}$ could give rise, in the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$, to the wave $\mathcal{P}_*^{(Q)}$ of the query Q. Indeed, the wave of the query Q is by definition a product of $m+r$ (not necessarily distinct) variable names from the set $\{ 1, N_1, \ldots, N_{m+w} \}$. At the same time, each term generated from the function $\mathcal{F}_{(Q)}^{(Q')}$ under each vector $\bar{K}_w$ cannot be a product of more than $m+r$ elements, please see Proposition 5.44 of Section 5.9.5. That is, whenever any element of the product in any such term is the constant 1, then the total number of the remaining multipliers constituting the term cannot exceed $m+r-1$. This contradiction concludes our proof for the case where $w = 1$.

(2) Suppose now that $w \geq 2$; that is, we have that the set $\{ 1, N_{m+1}, \ldots, N_{m+w} \}$ has at least two distinct variable names corresponding to the copy variables of the query Q. Similarly to our argument for the case where $w = 1$, for the $\mathcal{P}$ fixed as explained above, by Proposition 5.58 there must be a j between 1 and r inclusively, such that the *min*-expression that is the jth projection of $\mathcal{P}$ has (the *min*-expression) at least two distinct arguments from the set $\{ 1, N_{m+1}, \ldots, N_{m+w} \}$.

Fix an arbitrary value of j that satisfies this condition. Again similarly to our argument for the case where $w = 1$, we obtain that for the (at least) two distinct arguments from the set $\{ 1, N_{m+1}, \ldots, N_{m+w} \}$ in the *min*-expression that is the jth projection of $\mathcal{P}$, these distinct arguments must include at least two distinct elements of the set $\{ N_{m+1}, \ldots, N_{m+w} \}$. That is, the jth projection of $\mathcal{P}$ must have at least two distinct variable names, from among $N_{m+1}, \ldots, N_{m+w}$, as arguments of its *min*-expression. Denote the set of all those distinct arguments of that *min*-expression that are elements of the set $\{ N_{m+1}, \ldots, N_{m+w} \}$ as the set S^*. By our argument, the cardinality $|S^*|$ of the set S^* is at least two.

Now consider the function $f^{(\mathcal{S})}(Q')$ constructed for the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}$ as described in Section 5.10.8. Suppose that the term $\mathcal{P}$ in $f^{(\mathcal{S})}(Q')$, where $\mathcal{P}$ is fixed as explained before item (1) of this proof, belongs to the kth parenthesized expression of $f^{(\mathcal{S})}(Q')$, for some $k \geq 1$. In the remainder of the proof, we keep this value k fixed. Denote by (w.l.o.g.) $\mathcal{C}_1^{(Q')}$ through $\mathcal{C}_p^{(Q')}$, for some $p \geq 1$, all the p distinct monomial classes that have contributed to the construction of this kth parenthesized expression of $f^{(\mathcal{S})}(Q')$. From the set S^*, of cardinality at least two, being the set of arguments of the *min*-expression in the jth projection of $\mathcal{P}$, we infer that $p \geq 2$. Further, for the j fixed as above, consider the set, call it S^{**}, of all items that occur as the jth element of the copy signature of $\mathcal{C}_l^{(Q')}$, for all $l \in \{ 1, \ldots, p \}$. From the fact that $\mathcal{P}$ is a term in the kth parenthesized expression of $f^{(\mathcal{S})}(Q')$ to whose (the kth parenthesized expression) construction all of $\mathcal{C}_1^{(Q')}$ through

$\mathcal{C}_p^{(Q')}$ have contributed, we have that S^{**} is a superset of the set S^*. Hence, the set S^{**} contains at least two distinct elements of the set $\{ N_{m+1}, \ldots, N_{m+w} \}$, and the cardinality $|S^{**}|$ of the set S^{**} is at least two.

The contradiction that we are to arrive at is going to be as follows. We will show that, in addition to the vector $\bar{K}_w$ fixed as explained above, there exists a different copy-variable-ordering vector $\bar{K}'_w$ for the query Q, with the following property: The signature $\mathcal{S}(Q', \bar{K}_w)$ of the function $f^{(\mathcal{S})}(Q')$ w.r.t. the vector $\bar{K}_w$ is not identical to the signature $\mathcal{S}(Q', \bar{K}'_w)$ of the function $f^{(\mathcal{S})}(Q')$ w.r.t. the vector $\bar{K}'_w$. By Proposition 5.55, we will then obtain that the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ and the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}'_w]$ (for the query Q' and for the two distinct vectors $\bar{K}_w$ and $\bar{K}'_w$) are nonidentical polynomials. That allows us to immediately obtain a contradiction with Propositions 5.50 and 5.51, which together claim that for all pairs $(\bar{K}_w, \bar{K}'_w)$ of copy-variable-ordering vectors for the query Q, the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}_w]$ and the m-covering part of the function $\mathcal{F}_{(Q)}^{(Q')}[\bar{K}'_w]$ must be identical polynomials, whenever $Q \equiv_C Q'$ and Q is an explicit-wave query.

Thus, to complete this proof it remains to show that there exists a copy-variable-ordering vector $\bar{K}'_w$ for the query Q such that $\bar{K}'_w$ is not identical to the fixed vector $\bar{K}_w$, and such that the signature $\mathcal{S}(Q', \bar{K}_w)$ of the function $f^{(\mathcal{S})}(Q')$ w.r.t. the vector $\bar{K}_w$ is not identical to the signature $\mathcal{S}(Q', \bar{K}'_w)$ of the function $f^{(\mathcal{S})}(Q')$ w.r.t. the vector $\bar{K}'_w$.

We construct one vector $\bar{K}'_w$ with this desired property from the vector $\bar{K}_w$, as follows. Recall the set S^{**}, which contains at least two distinct elements of the set $\{ N_{m+1}, \ldots, N_{m+w} \}$. In the set S^{**}, let N_A be the element that is the minimal element of S^{**} under the total ($\leq$) order $\bar{K}_w$. Similarly, in the set S^{**}, let N_B be the element that is the maximal element of S^{**} under the total ($\leq$) order $\bar{K}_w$. (Actually, picking N_A to be any element of S^{**} that is distinct from variable name N_B would be sufficient for our purpose in this proof. We choose N_A as the minimal element to make it easy to see how we construct $\bar{K}'_w$ from $\bar{K}_w$ – by swapping, in the relative $\leq$ order of the variable names, the minimal element with the maximal element of S^{**} in the vectors $\bar{K}_w$ and $\bar{K}'_w$.)

By construction of the set S^{**}, it is immediate that:

- N_A and N_B are two distinct elements of the set $\{ N_{m+1}, \ldots, N_{m+w} \}$; and
- N_B is the variable name that the jth projection of the kth parenthesized expression of $f^{(\mathcal{S})}(Q')$ contributes to the signature $\mathcal{S}(Q', \bar{K}_w)$ of the function $f^{(\mathcal{S})}(Q')$ w.r.t. the vector $\bar{K}_w$. (This fact is immediate from Proposition 5.56.)

We now take as the desired vector $\bar{K}'_w$ an arbitrary copy-variable-ordering vector for Q such that N_B is the minimal element of $\bar{K}'_w$, with the exception of the element 1 of $\bar{K}'_w$, and such that N_A is the maximal element of $\bar{K}'_w$. That is, we take as $\bar{K}'_w$ an arbitrary copy-variable-ordering vector for Q whose first two elements are 1 and N_B, in this order, and whose last element is N_A. Clearly, by $w \geq 2$ we have that at least one such vector $\bar{K}'_w$ must exist.

Consider the variable name that the jth projection of the kth parenthesized expression of $f^{(\mathcal{S})}(Q')$ contributes to the signature $\mathcal{S}(Q', \bar{K}'_w)$ of the function $f^{(\mathcal{S})}(Q')$ w.r.t. the vector $\bar{K}'_w$. By construction of the set S^{**} and of the vector $\bar{K}'_w$, we have that this variable name is N_A.

Now for the two signatures $\mathcal{S}(Q', \bar{K}_w)$ and $\mathcal{S}(Q', \bar{K}'_w)$ to still be identical sets, it must be that some other, fixed projection of some fixed parenthesized expression of $f^{(\mathcal{S})}(Q')$ would contribute:

- the variable name N_B to the signature $\mathcal{S}(Q', \bar{K}'_w)$, and
- some variable name N_C that is distinct from N_B, to the signature $\mathcal{S}(Q', \bar{K}_w)$. ($N_C$ may or may not be identical to N_A. Intuitively, for the two signatures to be identical sets "on the balance," there is to be a cycle going through the variable names N_A and N_B, and that cycle may or may not involve other elements of the set $\{ N_{m+1}, \ldots, N_{m+w} \}$.)

This observation gives is the desired contradiction. Indeed, if some projection of some parenthesized expression of $f^{(\mathcal{S})}(Q')$ contributes the variable name N_B to the signature $\mathcal{S}(Q', \bar{K}'_w)$, then the *max*-expression for that projection can contain, as its arguments, only 1 in addition to N_B. (Recall that N_B is the minimal-value variable name under the total ($\leq$) order $\bar{K}'_w$.) Thus, it is not possible for the signatures $\mathcal{S}(Q', \bar{K}_w)$ and $\mathcal{S}(Q', \bar{K}'_w)$ to be identical sets. This observation completes the proof for the case $w \geq 2$, and hence completes the proof of the entire Proposition 5.57. □

5.10.10 Theorem 4.1: End of the proof

As summarized in Section 5.1.1, the proof of Theorem 4.1 is immediate from three results, as follows.

- Proposition 5.33 of Section 5.7 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 5.34 of Section 5.7 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 5.47 of Section 5.10 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. $\{D_{\bar{N}^{(i)}}(Q)\}$.

We can finally conclude that the result of Theorem 4.1 holds. Q.E.D.

6. 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 *ACM 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, U. Dayal, and V. R. Narasayya. An overview of business intelligence technology. *Commun. ACM*, 54(8):88–98, 2011.

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

[6] R. Chirkova. Equivalence and minimization of conjunctive queries under combined semantics. Technical Report TR-2010-24, NCSU, 2010. Available from `http://www.csc.ncsu.edu/research/tech/reports.php`.

[7] R. Chirkova. Equivalence and minimization tests for conjunctive queries under combined semantics. *Submitted for publication*, 2012.

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

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

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

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

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

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

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

[15] 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`.

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

[17] 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

For the convenience of the reviewers, this optional appendix provides additional information from the online paper [6] concerning the results presented in this current manuscript.

A. SUFFICIENT CONDITION FOR BAG CONTAINMENT

In this appendix we provide a detailed discussion of the relationship of Theorem 3.3 with the following result of [5].

THEOREM A.1. *[5] Given two CCQ bag queries Q and Q' such that there exists a CVM from Q' to Q. Then we have that $Q \sqsubseteq_B Q'$.* □

It is easy to see that Theorem A.1 is an immediate corollary of Theorem 3.3.

Note that Theorem A.1 is formulated here using the syntax of [9] that we adopt in this current paper. Recall that in [5], 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 [9]) 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 [5]. 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 [9], 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 A.1 reflects correctly the result of [5].

B. THE NONSURJECTIVE CONTAINMENT EXAMPLE

In this appendix we recall an example from [5].

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

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

For bag queries Q and Q', the authors of [5] 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 B.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.

C. CVMS VS MULTISET HOMOMORPHISMS

In this appendix we provide Example C.1 showing that general CVMs and multiset homomorphisms are incomparable when applied to pairs of CCQ queries. We also prove Proposition 3.3.

EXAMPLE C.1. *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: $\mu = \{A \to D, B \to E, C \to E\}$; $\mu' = \{D \to A, E \to B, F \to B\}$; and $\mu'' = \{D \to A, E \to B, F \to C\}$. 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 2.3 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 2.3 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 2.3. 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." □

D. PROOF OF SUFFICIENT CONDITION FOR A CCQ QUERY TO BE AN EXPLICIT-WAVE QUERY

This appendix provides a proof of Proposition 4.1.

PROOF. We prove that for all queries such as Q in the statement of Proposition 4.1, 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. □

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

In this appendix 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 F.)

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

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

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.

F. IN EXAMPLE 4.1, $Q \equiv_C Q'$ HOLDS

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

PROPOSITION F.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. We also rename the copy variables in a consistent way. That is, we use

$Q(X_1) \leftarrow r(X_1, Y_1, Y_2, X_2; Y_3), r(X_1, Y_1, Y_2, X_3; Y_4),$

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

PROOF. We will prove the claim of Proposition F.1 if we show that for an arbitrary database D and for an arbitrary constant $a \in adom(D)$, the sets $\Gamma_{\bar{S}}^{(a)}(Q, D)$ and $\Gamma_{\bar{S}}^{(a)}(Q', D)$ 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 $\{ X_1 \to a, Y_1 \to b, Y_2 \to c, X_2 \to d, X_3 \to e, Y_3 \to 1, Y_4 \to 1 \}$ 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 $\mu_{(n_1,n_2,l)} = \{ X_1 \to a, Y_1 \to b, Y_2 \to c, X_2 \to l, X_3 \to l, Y_3 \to n_1, Y_4 \to n_2 \}$ 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 is immediate 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. □

G. NECESSARY AND SUFFICIENT CONDITIONS OF [9] FOR COMBINED-SEMANTICS QUERY EQUIVALENCE

Cohen in [9] 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 [9] 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 Proposition 4.1 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 Proposition 4.1 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 [9] 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 [9] for combined-semantics equivalence of queries are subsumed by Theorem 4.2 of this current paper.

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 [9] would apply to a pairing of this query with an *arbitrary* query in the query language considered in [9]. (By definition, see Definition 2.1, CCQ queries do belong to the query language considered in [9].) 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 4.2 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.