Formalization And Correctness Of The CQC Method

Carles Farré, Ernest Teniente and Toni Urpí

Universitat Politècnica de Catalunya

Technical Report LSI-02-68-R

Abstract. In this report we formalize a new method that checks Query Containment for queries with negated IDB and EDB predicates as well as equality, inequality and order comparisons. The generality of our approach allows also our method to deal straightforwardly with query containment under constraints. We prove soundness and completeness of success and of failure of our method and we characterize the databases where these properties hold.

1 Introduction

Query Containment (QC) is the problem concerned with checking whether the answers that a query obtains are a subset of the answers obtained by another query for every database. When considering integrity constraints, the containment relationship between two queries does not need to hold for any state of the database but only for those that satisfy the integrity constraints. This idea is captured by the notion of Query Containment under Constraints (QCuC).

In this report, we formalize a method for checking QC and QCuC in the presence of negation on IDB subgoals. Intuitively, the aim of our Constructive Method for Query Containment Checking (CQC Method) is to construct a *counterexample* that proves that there is no QC (or QCuC).

The calculus we define for our method uses different *Variable Instantiation Patterns (VIPs)*, according to the syntactic properties of the queries and the databases considered in each test. Such a customization only affects the way that the facts to be part of the counterexample are instantiated. The aim is to prune the search of the counterexample by generating only the relevant facts.

Our method deals with the following features jointly:

- Safe negation on IDB and EDB subgoals.
- Equality/inequality comparisons ($=$, $\neq$) and order comparisons ($<$, $\leq$, $\geq$, $>$) interpreted on a dense order domain.

We provide proofs of the soundness a of our method: if the method terminates without building any counterexample, then containment holds (*failure soundness*); and if the method builds a finite counterexample, when recursively-defined IDB relations are not considered, then containment does not hold (*finite success soundness*).

We also claim that our method is semidecidable. That is, if there exists a finite counterexample, we prove that our method finds it and terminates (*finite success completeness*) when negation and recursion is stratified and strict [CL89]. Moreover, if containment holds between two queries we prove that our method fails finitely (*failure completeness*) when recursively-defined IDB relations are not considered.

This report is organized as follows. Section 2 sets the base concepts used through the report. In Section 3, presents the formalization of the CQC Method. Finally, in Section 4, we present the main correctness results of our method.

2 Base Concepts

2.1 Logic Programs

Throughout the paper, a, b, c, a_1, b_1,… are constants. The symbols X, Y, Z, X_1, Y_1, … denote variables. ā, b $\bar{a}_1$, denote sets of constants. $\bar{X}$, $\bar{Y}$, $\bar{Z}$, $\bar{X}_1$, … denote sets of variables. p, q, v, p_1, q_1, … are predicate symbols. A *term* is either a variable or a constant. If p is a n-ary predicate and T_1, …, T_n are terms, then $p(T_1, \ldots, T_n)$ is an *atom*, which can also be written as $p(\bar{T})$ when n is known from the context. The atom is *ground* if every T_i is a constant. A *ordinary literal* is defined as either an atom or a negated atom, i.e. $\neg p(\bar{T})$. A *built-in literal* has the form of $A_1\ \theta\ A_2$, where A_1 and A_2 are terms. Operator θ is $<$, $\leq$, $>$, $\geq$, $=$ or $\neq$.

Variables, constants and predicate symbols together with connectives ($\neg$, $\wedge$, $\vee$, $\leftarrow$ and $\leftrightarrow$), quantifiers ($\forall$, $\exists$) and punctuation symbols ("(", ")") conform an *alphabet*. The *first order language* given by an alphabet consists of the set of all formulas constructed from the symbols of the alphabet.

A *pre-interpretation* of a first order language L consists of a non-empty set *Dom*, called the *domain* of the pre-interpretation; and, for each constant in L, the assignment of an element in *Dom*.

An interpretation I of a first order language L consists of pre-interpretation J with domain *Dom* of L together with an assignment to each n-ary predicate symbol in L of a subset R of Dom^n (R is a *n-ary relation* on *Dom*).

Let I be an interpretation for a first order language L and let F be a closed formula of L. Then I is a model for F if F is true with respect to I. I is a model of a set of closed formulas S if I is a model for each formula in S.

Throughout the paper, we will consider *normal clauses* of the form

$$A \leftarrow L_1 \wedge \ldots \wedge L_m \qquad \text{with } m \geq 0,$$

where A is an atom and each L_i is a literal, either ordinary or built-in. All the variables appearing in A, as well as in each L_i, are assumed to be universally quantified over the whole formula. A is often called the *head* and $L_1 \wedge \ldots \wedge L_m$ is the *body* of the clause.

A set of normal clauses is called a *normal program*.

The *definition* of a predicate symbol r in a normal program P is the set of all clauses in P that have r in their head.

Terms, literals and the syntactic structures made of them, such as rule bodies, whole rules or facts, are *expressions*.

A *substitution* θ is a finite set of the form $\{X_1/T_1, \ldots, X_n/T_n\}$, where each variable X_i is unique and each term T_i is different from X_i. The term T_i is called a *binding* for X_i. θ is called a *ground substitution* if each T_i is a ground term, that is, a constant.

Let $\theta = \{X_1/T_1, \ldots, X_n/T_n\}$ be a substitution and E be an expression. Then Eθ, the *instance* of E by θ, is the expression obtained from E by simultaneously replacing each occurrence of the variable X_i in E by the term T_i.

Let J be a pre-interpretation of a normal program P and I an interpretation based on J. Then the *fixpoint operator* T_P' is defined as follows:

> $T_P'(I) = \{ A \mid$ there exists a normal clause $B \leftarrow L_1 \wedge \ldots \wedge L_n$ in P such that $A = B\theta$ and $I \models (L_1 \wedge \ldots \wedge L_n)\theta$ for some ground substitution θ $\}$

A normal program P is *stratified* if there is a partition $P = P_1 \cup \ldots \cup P_n$ such that the following two conditions hold for i = 1, 2, …, n:

1. If an atom $r(\bar{T})$ occurs positively in the body of a clause in P_i then the definition of r is contained within P_j with $j \leq i$.
2. If an atom $r(\bar{T})$ occurs negatively (as $\neg r(\bar{T})$) in the body of a clause in P_i then the definition of r is contained within P_j with $j < i$.

Let P be a normal program stratified by $P = P_1 \cup \ldots \cup P_n$. Define the following sequence:

$$M_1 = T^J_{P_1} \uparrow \omega\ (\varnothing)$$
$$M_2 = T^J_{P_2} \uparrow \omega\ (M_1)$$
$$\vdots$$
$$M_n = T^J_{P_n} \uparrow \omega\ (M_{n-1}),$$

where each $M_i = T^J_{P_i} \uparrow \omega\ (M_{i-1}) = T^J_{P_i}(T^J_{P_i}(\ldots(T^J_{P_i}(T^J_{P_i}(M_{i-1})))\ldots))$ is a *fixpoint* for $T^J_{P_i}$ verifying that $T^J_{P_i}(M_i) = M_i$. Then M_n is a minimal model of P and is considered to be the *fixpoint semantics* for P.

2.2 Deductive Databases, Queries and Query Containment

A *deductive rule* is a normal clause of the form:

$$p(\bar{X}) \leftarrow L_1 \wedge \ldots \wedge L_m \quad \text{with } m \geq 1$$

Literals in the body may be ordinary and built-in. Predicate symbols in a deductive rule range over the *extensional database* (EDB) predicates, which are the relations stored in the database, and the *intensional database* (IDB) predicates (like p above), which are the relations defined by the deductive rules. EDB predicates must not appear in the head of a deductive rule.

A *condition* is a formula of the (*denial*) form:

$$\leftarrow L_1 \wedge \ldots \wedge L_m \quad \text{with } m \geq 1$$

An interpretation I *satisfies* (is a model for) a condition $\leftarrow L_1 \wedge \ldots \wedge L_m$ if there exists no ground substitution θ such that $I \vDash (L_1 \wedge \ldots \wedge L_n)\theta$.

We require deductive rules and conditions to be *safe*, that is, every variable appearing in the head or in negative-ordinary or built-in literals of the body must also occur in a positive ordinary literal of the body.

A *query* Q is a finite set of deductive rules that defines a dedicated n-ary *query predicate* q. Without loss of generality, other predicates than q appearing in Q are EDB or IDB predicates.

A query Q_1 is *contained* in a query Q_2, denoted by $Q_1 \sqsubseteq Q_2$, if the set of answers of $Q_1(D)$ is a subset of those of $Q_2(D)$ for any database D. Moreover, Q_1 *is contained in* Q_2 *wrt IC*, denoted by $Q_1 \sqsubseteq_{IC} Q_2$, if the set of answers of $Q_1(D)$ is a subset of those of $Q_2(D)$ for any database D satisfying a finite set IC of conditions (*integrity constraints*).

3 Formalization of The CQC Method

Let Q_1 and Q_2 be two queries, DR be the set of deductive rules defining the database IDB relations and IC be a set finite set of conditions expressing the database integrity constraints. If the CQC method performs a successful CQC-derivation from $(\leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)\ \varnothing\ \varnothing\ \varnothing\ K)$ to $([]\ \varnothing\ T\ C\ K')$ then $Q_1 \not\sqsubseteq Q_2$, where K is the set of constants appearing in $DR \cup Q_1 \cup Q_2$. Moreover, if the CQC method performs a successful CQC-derivation from $(\leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)\ IC\ \varnothing\ \varnothing\ K'')$ to $([]\ \varnothing\ T\ C\ K''')$ then $Q_1 \not\sqsubseteq_{IC} Q_2$, where K'' is the set of constants appearing in $DR \cup Q_1 \cup Q_2 \cup IC$.

CQC-derivations start from a 5-tuple $(G_0\ F_0\ T_0\ C_0\ K_0)$ consisting of the goal $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$, the set of conditions to enforce $F_0 = \varnothing$ or IC, the initially-empty EDB $T_0 = \varnothing$, the empty set of conditions to maintain $C_0 = \varnothing$ and the set K_0 of constant values appearing in $DR \cup Q_1 \cup Q_2 [\cup IC]$.

A successful CQC-derivation reaches a 5-tuple $(G_n\ F_n\ T_n\ C_n\ K_n) = ([]\ \varnothing\ T\ C\ K')$, where the empty goal $G_n = []$ means that we have reached the goal G_0 we were looking for. The empty set $F_n = \varnothing$ means that no condition is waiting to be satisfied. $T_n = T$ is an EDB that satisfies G_0 as well as F_0. $C_n = C$ is a set of

conditions recorded along the derivation and that T also satisfies. $K_n = K'$ is the set of constant values appearing in $DR \cup Q_1 \cup Q_2[\cup IC] \cup T$.

On the contrary, if every "fair" CQC-derivation starting from ($\leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ $\varnothing$ $[\cup IC]$ $\varnothing$ $\varnothing$ K) is finite but does not reach ([] $\varnothing$ T C K'), it will mean that no EDB satisfies the goal $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ together with the set of conditions $F_0 = \varnothing$ $[\cup IC]$, concluding that $Q_1 \sqsubseteq Q_2$ ($Q_1 \sqsubseteq_{IC} Q_2$). Section 5 below provides the complete results and proofs regarding the soundness and completeness of the CQC method.

3.1 CQC-Nodes, CQC-Trees and CQC-Derivations

Let Q_1 and Q_2 be two queries, DR be the set of deductive rules defining the database IDB relations and IC be a set finite set of conditions expressing the database integrity constraints. A *CQC-node* is a 5-tuple of the form (G_i F_i T_i C_i K_i), where G_i is a goal to attain; F_i is a set of conditions to enforce; T_i is a set of ground EDB atoms, an EDB under construction; C_i is a set of conditions that are currently satisfied in T_i and that must be maintained; and K_i is the set of constants appearing in $R = DR \cup Q_1 \cup Q_2[\cup IC]$ and T_i.

A *CQC-tree* is inductively defined as follows:

1. The tree consisting of the single CQC-node (G_0 F_0 $\varnothing$ $\varnothing$ K) is a CQC-tree.
2. Let E be a CQC-tree, and (G_n F_n T_n C_n K_n) a leaf CQC-node of E such that $G_n \neq []$ or $F_n \neq \varnothing$. Then the tree obtained from E by appending one or more descendant CQC-nodes according to a CQC-expansion rule applicable to (G_n F_n T_n C_n K_n) is again a CQC-tree.

It may happen that the application of a CQC-expansion rule on a leaf CQC-node (G_n F_n T_n C_n K_n) does not obtain any new descendant CQC-node to be appended to the CQC-tree because some necessary constraint defined on the CQC-expansion rule is not satisfied. In such a case, we say that (G_n F_n T_n C_n K_n) is a *failed* CQC-node.

Each branch in a CQC-tree is a *CQC-derivation* consisting of a (finite or infinite) sequence (G_0 F_0 T_0 C_0 K_0), (G_1 F_1 T_1 C_1 K_1), … of CQC-nodes.

A CQC-derivation is *finite* if it consists of a finite sequence of CQC-nodes; otherwise it is *infinite*. A CQC-derivation is *successful* if it is finite and its last (leaf) CQC-node has the form ([] $\varnothing$ T_n C_n K_n). That is, both the goal to attain and the set of conditions to satisfy are empty. A CQC-derivation is *failed* if it is finite and its last (leaf) CQC-node is failed.

A CQC-tree is *successful* when at least one of its branches is a successful CQC-derivation. A CQC-tree is *finitely failed* when each one of its branches is a failed CQC-derivation.

3.2 Resolution Principle and Unification

Before describing the CQC-expansion rules we must introduce some base concepts regarding the resolution principle applied in some of such rules.

A *substitution* θ is a finite set of the form $\{X_1/Y_1, \ldots, X_n/Y_n\}$, where each variable X is unique and each term Y_i is different from X_i. The term Y_i is called a *binding* for X_i. θ is called a *ground substitution* if each Y_i is a ground term, that is, a constant. Let $\theta = \{X_1/Y_1, \ldots, X_n/Y_n\}$ be a substitution and E be an expression. Then $E\theta$, the *instance* of E by θ, is the expression obtained from E by simultaneously replacing each occurrence of the variable X_i in E by the term Y_i.

A *unifier* of two expressions E and E' is a substitution θ such that $E\theta$ is syntactically identical to $E'\theta$. If two expressions do not have a unifier then they are not *unifiable*. A unifier θ is called a *most general unifier* (*mgu*) for the two expressions if for each unifier α of E and E' there exists a substitution β such that $\alpha = \theta\beta$.

Let G be a goal or a condition of the form $\leftarrow L_1 \wedge \ldots \wedge p(\bar{X}_1) \wedge \ldots \wedge L_p$, $p \geq 1$, and C be an *input clause* of the form $P(\bar{X}_2) \leftarrow M_1 \wedge \ldots \wedge M_q$, $q \geq 1$. Then S is a *resolvent* for G and C using mgu θ if the following conditions hold:

– $p(\bar{X}_1)$ is an atom, called the *selected* atom, in G.

– θ is a mgu of $p(\bar{X}_1)$ and $p(\bar{X}_2)$.

– S is the clause $\leftarrow (L_1 \wedge \ldots \wedge M_1 \wedge \ldots \wedge M_q \wedge \ldots \wedge L_p)\theta$.

A1) $P(G_i) = d(\bar{X})$ is a positive IDB atom:

$$\frac{(G_i\ F_i\ T_i\ C_i\ K_i)}{(G_{i+1,1}\ F_i\ T_i\ C_i\ K_i) \mid \ldots \mid (G_{i+1,m}\ F_i\ T_i\ C_i\ K_i)}$$

only if $m \geq 1$ and each $G_{i+1,j}$ is the resolvent for G_i and some deductive rule $d(\bar{Y}) \leftarrow M_1 \wedge \ldots \wedge M_q$ in R.

A2) $P(G_i) = b(\bar{X})$ is a positive EDB atom:

$$\frac{(G_i\ F_i\ T_i\ C_i\ K_i)}{((G_i \backslash b(\bar{X}))\sigma_1\ F_{i+1,1}\ T_{i+1,1}\ C_{i+1,1}\ K_{i+1,1}) \mid \ldots \mid ((G_i \backslash b(\bar{X}))\sigma_m\ F_{i+1,m}\ T_{i+1,j}\ C_{i+1,m}\ K_{i+1,m})}$$

such that $F_{i+1,j} = F_i \cup C_i$, $T_{i+1,j} = T_i \cup \{b(\bar{X})\sigma_j\}$ and $C_{i+1,j} = \emptyset$ if $b(\bar{X})\sigma_j \notin T_i$; otherwise $F_{i+1,j} = F_i$, $T_{i+1,j} = T_i$ and $C_{i+1,j} = C_i$. Each σ_j is one out of m possible distinct ground substitutions, obtained via a variable instantiation procedure from $(vars(\bar{X}), \emptyset, K_i)$ to $(\emptyset, \sigma_j, K_{i+1,j})$ according to the appropriate variable instantiation pattern, that assigns a constant from $K_{i+1,j}$ to each variable in $vars(\bar{X})$. See more details in section 3.4.

A3) $P(G_i) = \neg p(\bar{X})$ is a ground negated atom:

$$\frac{(G_i\ F_i\ T_i\ C_i\ K_i)}{(G_i \backslash \neg p(\bar{X})\ F_i \cup \{\leftarrow p(\bar{X})\}\ T_i\ C_i\ K_i)}$$

A4) $P(G_i) = L$ is a ground built-in literal:

$$\frac{(G_i\ F_i\ T_i\ C_i\ K_i)}{(G_i \backslash L\ F_i\ T_i\ C_i\ K_i)}$$

only if L is evaluated true.

Table 3.1

3.3 The CQC-Expansion Rules

The nine *CQC-expansion rules* are listed in tables 3.1 and 3.2. For the sake of notation, if $G_i = \leftarrow L_1 \wedge \ldots \wedge L_{j-1} \wedge L_j \wedge L_{j+1} \wedge \ldots \wedge L_m$ then $G_i \backslash L_j = \leftarrow L_1 \wedge \ldots \wedge L_{j-1} \wedge L_{j+1} \wedge \ldots \wedge L_m$. If $G_i = \leftarrow L_1 \wedge \ldots \wedge L_m$ then $G_i \wedge p(\bar{X}) = \leftarrow L_1 \wedge \ldots \wedge L_m \wedge p(\bar{X})$.

The application of a CQC-expansion rule on a given CQC-node $(G_i\ F_i\ T_i\ C_i\ K_i)$ may result in none, one or several alternative (branching) descendant CQC-nodes depending on the selected literal $P(J_i) = L$. Here, J_i is either the goal G_i or any of the conditions $F_{i,j}$ in F_i. L is selected according to a safe computation rule P [Llo87], which selects negative and built-in literals only when they are fully grounded. To guarantee that such literals are selected sooner or later we require deductive rules and goals to be safe.

Once a literal is selected, only one of the CQC-expansion rules can be applied. We distinguish two classes of rules: the A-rules and the B-rules. A-rules are those where the selected literal belongs to the goal G_i. Instead, B-rules correspond to those where the selected literal belongs to any of the conditions $F_{i,j}$ in F_i. Inside each class of rules, they are differentiated in relation to the type of the selected literal.

In each CQC-expansion rule, the part above the horizontal line presents the CQC-node to which the rule is applied. Below the horizontal line is the description of the resulting descendant CQC-nodes. Vertical bars separate alternatives corresponding to different descendants. Some rules like A1, A5, B2 and B4 include also an "only if" condition that constraints the circumstances under which the expansion is possible. If such a condition is evaluated false, the CQC-node to which the rule is applied becomes a failed CQC-node.

B1) $P(F_{i,j}) = d(\bar{X})$ is a positive IDB atom:

$$\frac{(G_i\ \{F_{i,j}\}\cup F_i\ T_i\ C_i\ K_i)}{(G_i\ S\cup F_i\ T_i\ C_i\ K_i)}$$

where S is the set of all resolvents S_u for clauses in R and $F_{i,j}$ on $d(\bar{X})$. S may be empty.

B2) $P(F_{i,j}) = b(\bar{X})$ is a positive EDB atom:

$$\frac{(G_i\ \{F_{i,j}\}\cup F_i\ T_i\ C_i\ K_i)}{(G_i\ S\cup F_i\ T_i\ C_{i+1}\ K_i)}$$

only if $[] \notin S$.
$C_{i+1} = C_i$ if $\bar{X}$ contains no variables and $b(\bar{X}) \in T_i$; otherwise, $C_{i+1} = C_i \cup \{F_{i,j}\}$
S is the set of all resolvents of clauses in T_i with $F_{i,j}$ on $b(\bar{X})$. S may be empty, meaning that $b(\bar{X})$ cannot be unified with any atom in T_i.

B3) $P(F_{i,j}) = \neg p(\bar{X})$ is a ground negative ordinary literal:

$$\frac{(G_i\ \{F_{i,j}\}\cup F_i\ T_i\ C_i\ K_i)}{(G_i\ \{\leftarrow p(\bar{X})\}\cup\{F_{i,j}\backslash\neg p(\bar{X})\}\cup F_i\ T_i\ C_i\ K_i)\ \textit{only if}\ F_{i,j}\backslash\neg p(\bar{X}) \neq [] \quad | \quad (G_i \wedge p(\bar{X})\ F_i\ T_i\ C_i\ K_i)}$$

B4) $P(F_{i,j}) = L$ is a ground built-in literal that is evaluated true:

$$\frac{(G_i\ \{F_{i,j}\}\cup F_i\ T_i\ C_i\ K_i)}{(G_i\ \{F_{i,j}\backslash L\}\cup F_i\ T_i\ C_i\ K_i)}$$

only if $F_i\backslash L \neq []$.

B5) $P(F_{i,j}) = L$ is a ground built-in literal that is evaluated false:

$$\frac{(G_i\ \{F_{i,j}\}\cup F_i\ T_i\ C_i\ K_i)}{(G_i\ F_i\ T_i\ C_i\ K_i)}$$

Table 3.2.

3.4 Variable Instantiation Patterns

We distinguish three different *variable instantiation patterns*, VIPs for shorthand. Each of them defines how the CQC method has to instantiate the EDB facts to be added on the EDB under construction and they depend on the syntactical properties of the databases an queries considered. The three VIPs are: *Simple VIP*, *Negation VIP* and *Dense Order VIP*.

The CQC method uses the *Simple VIP* when checking containment but not QC under constraints. Moreover, the deductive rules defining query predicates as well as IDB predicates must satisfy the following conditions: they must not have any negative or built-in literal[1] in their rule bodies; they must not have constants in their heads; and they must not have any variable appearing twice or more times in their heads. According to the Simple VIP, each distinct variable is bound to a distinct new constant.

The CQC method uses the *Negation VIP* when checking QCuC or when checking containment under the presence of negation and or (in)equality comparisons (=, ≠) or. In both cases, order comparisons (<, ≤, >, ≥) are not allowed.

The other VIP, *Dense Order VIP*, is applied when there are order comparisons (<, ≤, >, ≥) in the deductive rules, with or without negation, interpreted on a dense order domain. In this case, each distinct variable must be bound to a constant according to either a former or a new location in the total linear order of constants.

[1] However, note that if a deductive rule has a literal of the form $z = k$ or $z = x$ in its body such that z does not appear in the head, then that literal can be omitted by replacing each occurrence of z in the rule body by k or x, respectively.

A variable instantiation procedure from ($\{X_1, X_2, \ldots, X_n\}$ θ_0 K_0) to ($\varnothing$ θ_n K_n) is a sequence ($\{X_1, X_2, \ldots, X_n\}$ θ_0 K_0), ($\{X_2, \ldots, X_n\}$ θ_1 K_1), …, ($\varnothing$ θ_n K_n) such that for each $0 \leq i \leq n$, θ_i is a ground substitution and K_i is a set of constants.

A variable instantiation step performs a transition from ($\bar{X}_i$ θ_i K_i) to ($\bar{X}_{i+1}$ θ_{i+1} K_{i+1}) that instantiates the variable X_{i+1} of $\bar{X}_i$ according to one of the *VIP-rules* defined by the selected *variable instantiation pattern* (*VIP*). The application of the appropriate VIP to a given class of queries and databases ensures the completeness of the CQC method with respect to that class.

VIP-rule for the Simple VIP

S. $\bar{X}_{i+1} = \bar{X}_i \backslash X_{i+1}$, $\theta_{i+1} = \theta_1 \cup \{X_{i+1}/k_{new}\}$ and $K_{i+1} = K_i \cup \{k_{new}\}$, where $k_{new} \notin K_i$.

VIP-rules for the Negation VIP

N1. $\bar{X}_{i+1} = \bar{X}_i \backslash X_{i+1}$, $\theta_{i+1} = \theta_1 \cup \{X_{i+1}/k\}$ and $K_{i+1} = K_i$, where $k \in K_i$.

N2. $\bar{X}_{i+1} = \bar{X}_i \backslash X_{i+1}$, $\theta_{i+1} = \theta_1 \cup \{X_{i+1}/k_{new}\}$ and $K_{i+1} = K_i \cup \{k_{new}\}$, where $k_{new} \notin K_i$.

VIP-rules for the Dense Order VIP

Den1. $\bar{X}_{i+1} = \bar{X}_i \backslash X_{i+1}$, $\theta_{i+1} = \theta_1 \cup \{X_{i+1}/k\}$ and $K_{i+1} = K_i$, where $k \in K_i$.

Den2. $\bar{X}_{i+1} = \bar{X}_i \backslash X_{i+1}$, $\theta_{i+1} = \theta_1 \cup \{X_{i+1}/k_{new}\}$ and $K_{i+1} = K_i \cup \{k_{new}\}$, where $k_{new} < min(K_i)$.

Den3. $\bar{X}_{i+1} = \bar{X}_i \backslash X_{i+1}$, $\theta_{i+1} = \theta_1 \cup \{X_{i+1}/k_{new}\}$ and $K_{i+1} = K_i \cup \{k_{new}\}$, where $k_j < k_{new} < k_{j+1}$, $\{k_j, k_{j+1}\} \subseteq K_i$ and there is no $k_h \in K_i$ such that $k_j < k_h < k_{j+1}$.

Den4. $\bar{X}_{i+1} = \bar{X}_i \backslash X_{i+1}$, $\theta_{i+1} = \theta_1 \cup \{X_{i+1}/k_{new}\}$ and $K_{i+1} = K_i \cup \{k_{new}\}$, where $max(K_i) < k_{new}$.

Table 3.3.

The formalization of the VIP-rules are given in table 3.3. We denote constants as k, k_{new} and k_i. *max* and *min* are two functions that range over sets of constants and they return the constants having the greatest value and the least value, respectively, of those sets.

4 Correctness Results For The CQC Method

Let Q_1 and Q_2 be two queries, *DR* be the set of deductive rules defining the database IDB relations, *IC* be a set finite set of conditions expressing the database integrity constraints and *K* be the set of constants appearing in $DR \cup Q_1 \cup Q_2 [\cup IC]$. Let $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ be the initial goal and $F_0 = \varnothing$ $[\cup IC]$ be the initial set of conditions to enforce. The first property we prove, which is called *failure soundness*, tells us that if there exists a finitely failed CQC-tree rooted at (G_0 F_0 $\varnothing$ $\varnothing$ K) then $Q_1 \sqsubseteq Q_2$ ($Q_1 \sqsubseteq_{IC} Q_2$) (section 4.2).

The second correctness property we present, called *failure completeness*, tells us that if $Q_1 \sqsubseteq Q_2$ ($Q_1 \sqsubseteq_{IC} Q_2$) then *fair* CQC-trees are finitely failed when their root is (G_0 F_0 $\varnothing$ $\varnothing$ K) (section 4.3).

The third property that we prove is *finite-success soundness*. In this way, if there exists a finite successful CQC-derivation rooted at (G_0 F_0 $\varnothing$ $\varnothing$ K) then $Q_1 \not\sqsubseteq Q_2$ ($Q_1 \not\sqsubseteq_{IC} Q_2$) (section 4.4).

Finally, the fourth property is called *finite-success completeness* and it states that if there exists a finite EDB satisfying the noncontaiment goal *NC*, then there exists a successful CQC-derivation rooted at (G_0 F_0 $\varnothing$ $\varnothing$ K) that constructs this EDB, or a subset of it (section 4.5).

Before proving these correctness results, we need to make explicit the model-theoretic semantics to with respect those results are established. This is the subject of section 4.1.

4.1 Partial Completion of Deductive Rules

The *completed definition* of a n-ary IDB predicate *d* is formally defined as follows. Let $d(T_1, \ldots, T_n) \leftarrow L_1 \wedge \ldots \wedge L_m$ a deductive rule defining *d*. First, we transform the given rule into

$$d(X_1, \ldots, X_n) \leftarrow (X_1 = T_1) \wedge \ldots \wedge (X_n = T_n) \wedge L_1 \wedge \ldots \wedge L_m$$

where $X_1, \ldots, X_n$ are variables not appearing in the deductive rule. Then, if $Y_1, \ldots, Y_d$ are the variables of the original rule, we transform this into

$$d(X_1, \ldots, X_n) \leftarrow \exists Y_1 \ldots \exists Y_d ((X_1 = T_1) \wedge \ldots \wedge (X_n = T_n) \wedge L_1 \wedge \ldots \wedge L_m)$$

Now suppose that this transformation is made for each deductive rule in the definition of d. Then we obtain $k \geq 1$transformed formulas of the form

$$\begin{array}{c} d(X_1, \ldots, X_n) \leftarrow E_1 \\ \vdots \\ d(X_1, \ldots, X_n) \leftarrow E_k \end{array}$$

where each E_i has the general form

$$\exists Y_1 \ldots \exists Y_d ((X_1 = T_1) \wedge \ldots \wedge (X_n = T_n) \wedge L_1 \wedge \ldots \wedge L_m)$$

Finally, the completed definition of D is then the formula

$$\forall X_1 \ldots \forall X_n (d(X_1, \ldots, X_n) \leftrightarrow E_1 \vee \ldots \vee E_k).$$

Let R be a set of deductive rules. We define the *partial completion* of R, denoted by $pComp(R)$, as the collection of completed definitions of IDB predicates in R together with an equality theory. This later one includes a set of axioms stating explicitly the meaning of the built-in predicate = introduced in the completed definitions.

The following theorem relates *pComp* semantics and fixpoint semantics, which will be useful in section 4.3.

Theorem 4.1
Let R be a set of deductive rules, J a pre-interpretation of R and I an interpretation based on J. Then I is a model of $pComp(R)$ if and only if $T_R^J(I) = I$ (I is a fixpoint of R).

Proof: We adapt the proof given in [Llo87] for proving that I is a model of $Comp(R)$ if and only if $T_R^J(I) = I$.

Suppose first that $T_R^J(I) = I$. It suffices to show that I is a model for each of he completed definitions of $pComp(R)$ of the form $\forall x_1 \ldots \forall x_n (d(X_1, \ldots, X_n) \leftrightarrow E_1 \vee \ldots \vee E_k)$. Since $T_R^J(I) \subseteq I$, it follows that I is a model for $\forall x_1 \ldots \forall x_n (d(X_1, \ldots, X_n) \leftarrow E_1 \vee \ldots \vee E_k)$. Also, since $T_R^J(I) \supseteq I$, it follows that I is a model for $\forall x_1 \ldots \forall x_n (d(X_1, \ldots, X_n) \rightarrow E_1 \vee \ldots \vee E_k)$.

Conversely, suppose that I is a model for $pComp(R)$. Then using the fact that I is a model for formulas of the form $\forall x_n (d(X_1, \ldots, X_n) \leftarrow E_1 \vee \ldots \vee E_k)$, it follows that $T_R^J(I) \subseteq I$. Similarly, using the fact that I is a model for formulas of the form $\forall x_1 \ldots \forall x_n (d(X_1, \ldots, X_n) \rightarrow E_1 \vee \ldots \vee E_k)$, it follows that $T_R^J(I) \supseteq I$. □

4.2 Failure Soundness

Let Q_1 and Q_2 be two queries, DR be the set of deductive rules defining the database IDB relations and $R = DR \cup Q_1 \cup Q_2$. A CQC-node $(G_i\ F_i\ T_i\ C_i\ K_i)$ is *satisfiable* if there exists an interpretation I which is a model for $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_n)\} \cup \forall F_i \cup T_i \cup \forall C_i$, where $G_i = \leftarrow L_1 \wedge \ldots \wedge L_n$.

A CQC-derivation is *satisfiable* if each one of its CQC-nodes is satisfiable. A CQC-tree is *satisfiable* if it has at least one satisfiable CQC-derivation.

We first prove in Lemma 4.2 that whichever a variable substitution is, there exists a variable procedure instantiation that obtains it according to the appropriate VIP, Negation or Dense Order.

Lemma 4.2 (*Completeness of the Negation and Dense Order VIPs*).
Let $\bar{X} = \{X_1, \ldots, X_n\}$ be a set of distinct uninstantiated variables with $n > 0$, then

- $\forall$ substitution $\sigma = \{X_1/k_1, \ldots, X_n/k_n\}$ such that each variable X_i in $\bar{X}$ s instantiated by a constant $k_i \in K \supseteq K_0$, where K_0 is the initial set of constants appearing in $R \cup Q_1 \cup Q_2 \cup \{G_0\} \cup F_0$.
- $\forall\ K' \subseteq K$ such that $K_0 \subseteq K'$.

there exists a variable instantiation procedure from $(\bar{X}\ \varnothing\ K')$ to $(\varnothing\ \sigma\ K'')$ according to the Negation (Dense Order) VIP such that $K_0 \subseteq K' \subseteq K'' \subseteq K$.

Proof: The proof is by induction over the size n of $\bar{X}$.
Case n = 1: In this case, $\bar{X} = \{X_1\}$, $k_1 \in K$ and $\sigma = \{X_1/k_1\}$. Therefore, the variable instantiation procedure has just one step from $(\{X_1\}\ \varnothing\ K')$ to $(\varnothing\ \sigma\ K'')$. Then

Negation VIP)

- either $k_1 \in K'$ and we apply N1, so $K'' = K'$.
- or $k_1 \notin K'$ and we apply N2, so $K'' = K' \cup \{k_1\}$. Since $k_1 \in K$, $K' \cup \{k_1\} \subseteq K$, so $K'' \subseteq K$.

Dense Order VIP)

- either $k_1 \in K'$ and we apply Den1, so $K'' = K'$.
- or $k_1 < \min(K')$ and we apply Den 2, so $K'' = K' \cup \{k_1\}$.
- or $k_j < k_1 < k_{j+1}$ where $k_j, k_{j+1} \in K'$ and there is no $k_h \in K'$ such that $k_j < k_h < k_{j+1}$; then we apply Den3, so $K'' = K' \cup \{k_1\}$.
- or $k_1 > \max(K')$ and we apply Den4, so $K'' = K' \cup \{k_1\}$.

In any case, $K'' \subseteq K$ since $K' \cup \{k_1\} \subseteq K$.

General case: $\bar{X} = \{X_1, \ldots, X_n\}$ and $\sigma = \{X_1/k_1, \ldots, X_n/k_n\}$ with n > 1. Let X_i a variable, $k_i \in K$ and $X_i/k_i \in \sigma$. There exists a variable instantiation step from $(\{X_i, X_{i+1}, \ldots, X_n\}\ \sigma_{i-1}\ K_{i-1})$ to $(\{X_{i+1}, \ldots, X_n\}\ \sigma_i\ K_i)$, where $\sigma_i = \sigma_{i-1} \cup \{X_i/k_i\}$. Then

Negation VIP)

- either $k_i \in K_{i-1}$ and we apply N1, so $K_i = K_{i-1}$.
- or $k_i \notin K_{i-1}$ and we apply N2, so $K_i = K_{i-1} \cup \{k_i\}$. Since $k_1 \in K$ and $K_{i-1} \subseteq K$ (by induction hypothesis), $K_{i-1} \cup \{k_1\} \subseteq K$.

Dense Order VIP)

- either $k_i \in K_{i-1}$ and we apply Den1, so $K_i = K_{i-1}$.
- or $k_i < \min(K_{i-1})$ and we apply Den 2, so $K_i = K_{i-1} \cup \{k_i\}$.
- or $k_j < k_i < k_{j+1}$ where $k_j, k_{j+1} \in K_{i-1}$ and there is no $k_h \in K_{i-1}$ such that $k_j < k_h < k_{j+1}$; then we apply Den3, so $K_i = K_{i-1} \cup \{k_i\}$.
- or $k_i > \max(K_{i-1})$ and we apply Den, so $K_i = K_{i-1} \cup \{k_i\}$.

In any case $K_{i-1} \cup \{k_i\} \subseteq K$ since $k_i \in K$. □

Lemma 4.4
The application of a CQC-Expansion Rule to a satisfiable CQC-Tree results in a satisfiable CQC-Tree, provided that the deductive rules and conditions are safe.

Proof: Suppose E is a satisfiable CQC-Tree, and a CQC Expansion Rule is applied to some CQC-node N_i on a branch (CQC-derivation) θ of E, producing a CQC-Tree E^*. We must show E^* is also a satisfiable CQC-Tree. The proof has several cases and subcases. Since E is satisfiable, E has at least one satisfiable CQC-derivation. Choose one, say it τ.

1. $\tau \neq \theta$. Then since a rules was applied only to θ, τ is still a CQC-derivation of E^*, hence E^* is satisfiable.
2. $\tau = \theta$. Then θ is itself satisfiable, that is, all the CQC-nodes occurring on it are satisfiable. Now we have subcases depending on which Rule was applied to $N_i = (G_i\ F_i\ T_i\ C_i\ K_i)$.

   A1) Suppose that G_i is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and its selected positive IDB literal $P(G_i) = M_j$ is $d(S_1, \ldots, S_n)$. Suppose also that the completed definition of d in $pComp(R)$ is
   $$\forall(d(X_1, \ldots, X_n) \leftrightarrow E_1 \vee \ldots \vee E_k)$$
   where E_i is
   $$\exists Y_{i,1} \ldots \exists Y_{i,d_i} ((X_1 = T_{i,1}) \wedge \ldots \wedge (X_n = T_{i,n}) \wedge L_{i,1} \wedge \ldots \wedge L_{i,m_i}).$$

Since $pComp(R) \cup \{\exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge d(S_1, \ldots, S_n) \wedge M_{j+1} \wedge \ldots \wedge M_q)\} \cup \forall F_i \cup T_i \cup \forall C_i$ are true with respect to an interpretation I, it follows that

$$\bigvee_{i=1}^{k} \exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge (S_1 = T_{i,1}) \wedge \ldots \wedge (S_n = T_{i,n}) \wedge L_{i,1} \wedge \ldots \wedge L_{i,m_i} \wedge M_{j+1} \wedge \ldots \wedge M_q)$$

is also true wrt. I.

On the other hand, if σ_i is an mgu for a resolvent for $d(S_1, \ldots, S_n)$ and a deductive rule $d(T_{i,1}, \ldots, T_{i,n}) \leftarrow L_{i,1} \wedge \ldots \wedge L_{i,m_i}$ in R, by [Lloyd87, lemma 15.2] we have

$$pComp(R) \vdash \forall((S_1 = T_{i,1}) \wedge \ldots \wedge (S_n = T_{i,n}) \leftrightarrow \sigma_i)$$

Hence by applying equality axioms,

$$\bigvee_{i=1}^{k} \exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge (S_1 = T_{i,1}) \wedge \ldots \wedge (S_n = T_{i,n}) \wedge L_{i,1} \wedge \ldots \wedge L_{i,m_i} \wedge M_{j+1} \wedge \ldots \wedge M_q) \leftrightarrow \bigvee_{i=1}^{k} \exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge L_{i,1} \wedge \ldots \wedge L_{i,m_i} \wedge M_{j+1} \wedge \ldots \wedge M_q)\sigma_i$$

Therefore, there exist at least one child CQC-node, $N_{i+1,g} = (G_{i+1,g}\ F_i\ T_i\ C_i\ K_i)$, with $G_{i+1,g} = \leftarrow (M_1 \wedge \ldots \wedge M_{j-1} \wedge L_{g,1} \wedge \ldots \wedge L_{g,m_g} \wedge M_{j+1} \wedge \ldots \wedge M_q)\sigma_g$, which is also true wrt. I.

A2) Suppose that G_i is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and $P(G_i) = M_j = b(\bar{X})$ is a EDB atom. Since $pComp(R) \cup \{\exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge B(t) \wedge M_{j+1} \wedge \ldots \wedge M_q)\} \cup \forall F_i \cup T_i \cup \forall C_i$ is true with respect to an interpretation I. Therefore, there must exist a variable substitution that, say ρ, which substitutes the variables in $\bar{X}$ by constants of the domain of I. Then it is straightforward to prove that. $pComp(R) \cup \{\exists((M_1 \wedge \ldots \wedge M_{j-1} \wedge b(\bar{X}) \wedge M_{j+1} \wedge \ldots \wedge M_q)\rho)\} \cup \forall F_i \cup T_i \cup \forall C_i$ is true wrt. I and, consequently, $pComp(R) \cup \{\exists((M_1 \wedge \ldots \wedge M_{j-1} \wedge M_{j+1} \wedge \ldots \wedge M_q)\rho)\} \cup \forall(F_i \cup C_i) \cup (T_i \cup \{b(\bar{X})\rho\})$ is also true wrt. I. On the other hand, according to Lemma 4.2, there exist a variable instantiation procedure from $(\bar{X}\ \varnothing\ K_i)$ to $(\varnothing\ \sigma_j\ K_{i+1,j})$ such that $\sigma_j = \rho$, using the appropriate variable instantiation pattern.

A3) Suppose that G_i is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and $P(G_i) = M_j = \neg p(\bar{X})$ is a ground negated ordinary atom. Since $pComp(R) \cup \{\exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge \neg\, p(\bar{X}) \wedge M_{j+1} \wedge \ldots \wedge M_q)\} \cup \forall F_i \cup T_i \cup \forall C_i$ is true with respect to an interpretation I, it follows that $pComp(R) \cup \{\exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge M_{j+1} \wedge \ldots \wedge M_q) \cup \forall(\{\leftarrow p(\bar{X}) \cup F_i) \cup T_i \cup \forall C_i$ is also true wrt. I.

A4) Suppose that G_i is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and $P(G_i) = M_j$ is a ground built-in atom which is evaluated true. Since $pComp(R) \cup \{\exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge M_j \wedge M_{j+1} \wedge \ldots \wedge M_q)\} \cup \forall F_i \cup T_i \cup \forall C_i$ is true with respect to an interpretation I, it follows that $pComp(R) \cup \{\exists(M_1 \wedge \ldots \wedge M_{j-1} \wedge M_{j+1} \wedge \ldots \wedge M_q) \cup \forall F_i \cup T_i \cup \forall C_i$ is also true wrt. I.

B1) Suppose that G_i is $\leftarrow L_1 \wedge \ldots \wedge L_m$, $F_{i,j}$ is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and $P(F_{i,j}) = M_g = d(\bar{X})$ is a positive IDB atom. According to [Lloyd87, lemma 1.5.3], $pComp(R) \vdash F_{i,j} \leftrightarrow \bigwedge_{u=1}^{k} S_u$, where each S_u is a distinct resolvent for clauses in R and $d(\bar{X})$. Therefore, if $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\{F_{i,j}\} \cup F_i) \cup T_i \cup \forall C_i$ is true with respect to an interpretation I, then $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\bigcup_{u=1}^{k} S_u \cup F_i) \cup T_i \cup \forall C_i$ is also true wrt. I.

B2) Suppose that G_i is $\leftarrow L_1 \wedge \ldots \wedge L_m$, $F_{i,j}$ is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and $P(F_{i,j}) = M_g = b(\bar{X})$ is a positive EDB atom. If $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\{F_{i,j}\} \cup F_i) \cup T_i \cup \forall C_i$ is true with respect to an interpretation I then $\forall(\neg M_1 \vee \ldots \vee \neg M_{g-1} \vee \neg\, b(\bar{X}) \vee \neg M_{g+1} \vee \ldots \vee \neg M_q)$ is true wrt. I and $T_i \subseteq I$. Then,

a. If $\forall b(\bar{X})$ is false wrt. T_i, in other words, no fact in T_i unifies with $b(\bar{X})$, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall F_i \cup T_i \cup \forall(C_i \cup \{F_{i,j}\})$ also is true wrt. I, even if $\exists b(\bar{X})$ is true wrt. I.
b. If $\exists b(\bar{X})$ is true wrt. T_i, then for each variable substitution σ_u such that $b(\bar{X})\sigma_u$ is true wrt T_i, $\forall(\neg M_1 \vee \ldots \vee \neg M_{g-1} \vee \neg M_{g+1} \vee \ldots \vee \neg M_q)\sigma_u$ must be true wrt. I. Therefore, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\bigcup_{u=1}^{k} S_u \cup F_i) \cup T_i \cup \forall(C_i \cup \{F_{i,j}\})$, where S_u is a resolvent for $F_{i,j}$ and some fact from T_i.

B3) Suppose that G_i is $\leftarrow L_1 \wedge \ldots \wedge L_m$, $F_{i,j}$ is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and $P(F_{i,j}) = M_g = \neg p(\bar{X})$ is a negated ordinary atom. If $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\{F_{i,j}\} \cup F_i) \cup T_i \cup \forall C_i$ is true with respect to an interpretation I then $\forall(\neg M_1 \vee \ldots \vee \neg M_{g-1} \vee \neg M_{g+1} \vee \ldots \vee \neg M_q) \vee p(\bar{X})$ is true wrt. I. Therefore,

a. If $p(\bar{X})$ is true wrt. I, then $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m \wedge p(\bar{X}))\} \cup \forall F_i \cup T_i \cup \forall C_i$ is true wrt. I.

b. If $p(\bar{X})$ is false wrt. *I*, then $\forall(\neg M_1 \vee \ldots \vee \neg M_{g-1} \vee \neg M_{g+1} \vee \ldots \vee \neg M_q)$ must be true wrt. *I*. Therefore, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\{\leftarrow p(\bar{X})\} \cup \{F_{i,j} \backslash \neg p(\bar{X})\} \cup F_i) \cup T_i \cup \forall C_i$ is true wrt. *I*.

B+) Suppose that G_i is $\leftarrow L_1 \wedge \ldots \wedge L_m$, $F_{i,j}$ is $\leftarrow M_1 \wedge \ldots \wedge M_q$ and $P(F_{i,j}) = M_g$ is an built-in atom. If $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\{F_{i,j}\} \cup F_i) \cup T_i \cup \forall C_i$ is true with respect to an interpretation *I* then $\forall(\neg M_1 \vee \ldots \vee \neg M_{g-1} \vee \neg M_{g+1} \vee \ldots \vee \neg M_q) \vee \neg M_g$ is true wrt. *I*. Therefore,

B4) M_g is evaluated true, so $\forall(\neg M_1 \vee \ldots \vee \neg M_{g-1} \vee \neg M_{g+1} \vee \ldots \vee \neg M_q) \vee \neg M_g$ must be true wrt. *I*. Therefore, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall(\{F_{i,j} \backslash M_g\} \cup F_i) \cup T_i \cup \forall C_i$ is true wrt. *I*.

B5) M_g is evaluated false, so $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall F_i \cup T_i \cup \forall C_i$ is true wrt. *I*. □

Lemma 4.5
A finitely failed CQC-tree is not satisfiable.

Proof: According to the definition given above, all the branches (CQC-derivations) in a finitely failed CQC-tree are finite and their last (leaf) CQC-node is failed. Analyzing all the possible cases where a CQC-node is failed, we conclude that such CQC-nodes are not satisfiable. Let us consider a failed CQC-node $(G_n\ F_n\ T_n\ C_n\ K_n)$ such that $G_n = \leftarrow L_1 \wedge \ldots \wedge L_m$:

1. The selected literal $P(G_n) = d(\bar{X})$ was a IDB atom, so the applied CQC-rule was A1. However, in this case, there was no resolvent for every derivation rule in *R* and G_i Therefore, $pComp(R) \models G_i$ and, hence, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall F_n \cup T_n \cup \forall C_n$ cannot be satisfiable.
2. The selected literal $P(G_n) = L$ was a ground built-in literal, so the applied CQC-rule was A5, but L was evaluated false. Clearly, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall F_n \cup T_n \cup \forall C_n$ cannot be satisfiable.
3. The selected literal $P(F_{n,j}) = b(\bar{X})$ was a positive EDB literal, where $F_{n,j} = \leftarrow b(\bar{X}) \in F_n$, so the applied CQC-rule was B2. In this case $b(\bar{X})$ unified with some fact in T_n, so [] was a resolvent for $b(\bar{X})$ and a fact in T_n. Therefore, there is a variable substitution σ such that $b(\bar{X})\sigma$ is true wrt. T_n and, thus, $\forall(\leftarrow b(\bar{X})) \cup T_n$ cannot be satisfiable. Hence, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall F_n \cup T_n \cup \forall C_n$ is not satisfiable.
4. The selected literal $P(F_{n,j}) = L$ was a ground built-in literal that was evaluated true, where $F_{n,j} = \leftarrow L \in F_n$, so the applied CQC-rule was B4, but $F_{n,j} \backslash L = []$. Clearly, since $\forall(\leftarrow L)$ is not satisfiable, $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_m)\} \cup \forall F_n \cup T_n \cup \forall C_n$ cannot be satisfiable.

Since no failed CQC-node is satisfiable, a finitely failed CQC-tree cannot be satisfiable according to the definitions given above. □

Theorem 4.6 (*Failure Soundness*)
Let Q_1 and Q_2 be two queries, *DR* be the set of deductive rules defining the database IDB relations and $R = DR \cup Q_1 \cup Q_2$. Let also $G_0 = \leftarrow q_1(X_1, \ldots, X_n) \wedge \neg q_2(X_1, \ldots, X_n)$ be the initial goal, $F_0 = \varnothing$ [$\cup IC$] be the initial set of conditions to enforce and *K* be the set of constants appearing in $DR \cup Q_1 \cup Q_2 \cup F_0$. If there exists a finitely failed CQC-Tree rooted at $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ then $Q_1 \sqsubseteq Q_2$ ($Q_1 \sqsubseteq_{IC} Q_2$) provided that the deductive rules and conditions in $DR \cup Q_1 \cup Q_2$[$\cup IC$] are safe.

Proof: We have a finitely failed CQC-tree rooted at $(G_0\ F_0\ \varnothing\ \varnothing\ K)$. Let us suppose that $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ is satisfiable and we derive a contradiction. Since the initial CQC-tree $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ is satisfiable, every subsequent CQC-Tree we construct is satisfiable by Lemma 4.4. However, a satisfiable CQC-tree cannot be finitely failed by Lemma 4.5.

Since $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ is not satisfiable, we have that $pComp(R) \cup \{\exists X_1,\ldots,X_n\ (q_1(X_1,\ldots,X_n) \wedge \neg q_2(X_1,\ldots,x_n)\} \cup \forall F_0$ is not satisfiable by definition. Therefore, $pComp(R) \cup \forall F_0 \models G_0$ or, what is the same, $pComp(R) \cup \forall F_0 \models \forall x_1,\ldots,x_n\ (q_1(X_1,\ldots,X_n) \rightarrow q_2(X_1,\ldots,X_n))$. In other words, $Q_1 \sqsubseteq Q_2$ ($Q_1 \sqsubseteq_{IC} Q_2$). □

4.3 Failure Completeness

A *level mapping* of a set of clauses (i.e. deductive rules, conditions and/or goals) is a mapping from its set of IDB predicate to the non-negative integers. We refer to the value of an IDB predicate under this mapping as the *level* of that IDB predicate. The level of any EDB predicate is 0.

A set of clauses is *hierarchical* if has a level mapping such that, in every deductive rule $d(X_1, \ldots, X_n) \leftarrow E_i$ defining d, the level of every ordinary predicate in E_i is less than the level of d.

A CQC-derivation is *open* when it is not failed. That is, when the derivation is either infinite or finite with its last (leaf) CQC-node having the form of $([] \varnothing T_n C_n K_n)$.

A CQC-derivation θ is *saturated for the CQC-expansion Rules* if for every CQC-node $(G_i F_i T_i C_i K_i)$ in θ the following properties hold:

1. For each literal $L_{i,j} \in G_i$ there exists a node $(G_n F_n T_n C_n K_n)$, $n \geq i$, such that $P(G_n) = L_{i,j}\sigma_{i+1}\ldots\sigma_n$ is the selected literal on that node to apply a CQC A-rule, where $\sigma_{i+1}\ldots\sigma_n$ is the composition of the substitutions used in the intermediate nodes.
2. For each condition $F_{i,j_i} \in F_i$ there exists a node $(G_n F_n T_n C_n K_n)$, $n \geq i$, such that $F_{n,j_n} \in F_n$ is the selected condition on that node to apply a CQC B-rule and $F_{n,j_n} = F_{i,j_i}$.

A CQC-derivation is said to be *fair* when it is either failed or open and saturated for the CQC-expansion Rules. A CQC-tree is *fair* if each one of its CQC-derivations (branches) is fair. Note that a finitely failed CQC-tree is always fair, but the inverse is not necessarily true.

Lemma 4.7
Let R be a set of deductive rules, $G = \leftarrow L_1 \wedge \ldots \wedge L_k$ be a goal, F be a set of conditions and K be the set of constants in $\{G_0\} \cup F_0 \cup R$. If there exists a saturated open CQC-derivation starting from $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ then $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_k)\} \cup \forall F_0$ has a model provided that $\{G_0\} \cup F_0 \cup R$ is safe and hierarchical.

Proof: We prove the lemma by induction on the maximum level of the ordinary predicates in $\{G_0\} \cup F_0 \cup R$.

Predicate level = 0
The maximum level of the ordinary predicates in $\{G_0\} \cup F_0 \cup R$ is 0, so $R = \varnothing$ and all the literals in $\{G_0\} \cup F_0$ are either EDB or built-in. The proof is just a special case of the proof for predicate levels > 0. Hence it suffices to prove this latter.

Predicate level > 0
Now the maximum level of the ordinary predicates in $\{G_0\} \cup F_0 \cup R$ is $n > 0$.

Suppose that the saturated open CQC-derivation consist of a sequence of CQC nodes $(G_0\ F_0\ \varnothing\ \varnothing\ K_0)$, $(G_1\ F_1\ T_1\ C_1\ K_1)$, … and $\sigma_1, \sigma_2, \ldots$ are the variable substitutions used by the CQC expansion rules to each CQC-step. We define the following sets:

$POS = \{ A \mid A$ is a positive ordinary literal occurring in some goal $G_i\}$

$NEG = \{ B \mid \neg B$ is a negative ordinary literal occurring in some goal $G_i\}$

$BI = \{ C \mid C$ is a built-in literal occurring in some goal $G_i\}$

We also define the pre-interpretation J [Llo87] on which the model we construct will be based. Suppose that L is the underlying first order language for $\{G_0\} \cup F_0 \cup R$. We assume that no variable appears in more than one input clause from R in any CQC expansion in the derivation and that no variable appearing in G_i also occurs in any input clause[2]. This property is called *variable standardization*. We define a relation * on the set of all terms in L as follows. Let S and T be terms in L, variables or constants. Then $S*T$ if there exists $j \geq 1$ such that $S\sigma_1\ldots\sigma_j = T\sigma_1\ldots\sigma_j$, that is, $\sigma_1\ldots\sigma_j$ unifies s and T. Clearly, * is an equivalence relation. The domain D of the pre-interpretation J is then defined to be the set of all *-

[2] That is, each time that a deductive rule defined in **R** is used as input clause in some CQC expansion, the variables appearing in this rule are substituted by new variables that have not appeared before.

equivalence classes of terms in L. If S is a term, we denote its equivalence class by $[S]$. If c is a constant, we may say that [c] = c. We adopt the convention that if $p(\bar{S}) = p(S_1,\ldots, S_m)$ then $p([S_1], \ldots, [S_m])$ may also be denoted by $p([\bar{S}])$..

Due to the fact that R is safe and hierarchical and that the open CQC-derivation is saturated, it is straightforward to prove that for any variable V_k appearing in a literal in G_i exists $\sigma_{i+1}\ldots\sigma_j$, $i \leq j$, such that $V_k\sigma_{i+1}\ldots\sigma_j$ occurs in at least one positive EDB literal in G_j. Moreover, there exists $\sigma_{i+1}\ldots\sigma_j\sigma_{j+1}\ldots\sigma_m$, $i \leq j \leq m$, such that $V_k\sigma_{i+1}\ldots\sigma_j\sigma_{j+1}\ldots\sigma_m = c$ is a constant. For this reason, $[V_k]$ denotes the equivalence class to which V_k belongs as well as the constant c itself as a privileged representative of such an equivalence class.

We will construct a fixpoint M_k for $T^J_{R_k}$[3] where $0 \leq k \leq n$, which must satisfy the following properties:

(i) $p(\bar{T}) \in POS$ and level(p) $\leq k \Rightarrow P([\bar{T}]) \in M_k$.

(ii) If $F_{i,j_i} = \leftarrow a_1(\bar{A}_1) \wedge \ldots \wedge a_p(\bar{A}_p) \wedge \neg b_1(\bar{B}_1) \wedge \ldots \wedge \neg b_q(\bar{B}_q) \wedge C_1 \wedge\ldots \wedge C_t \in F_i$, p+q+t ≥ 1, and level(a_f) $\leq k$ and level(b_g) $\leq k$, then there exist no ground variable substitution ρ such that $a_f(\bar{A}_f\rho) \in M_k$, $a_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true.

(iii) $q(\bar{S}) \in NEG$ and level(q) $\leq k \Rightarrow q([\bar{S}]) \notin M_k$.

(iv) $C = A_1\ \theta\ A_2 \in BI$ such that A_1, A_2 are constants, and/or variables occuring in positive ordinary literals about predicates whose level $\leq k \Rightarrow [A_1]\ \theta\ [A_2]$ is true.

We first define:

$I_0 = \{p([\bar{T}]) \mid p(\bar{T}) \in POS$ and level(p) = 0 $\}$

Since $R_0 = \varnothing$, that is, the subset of deductive rules defining predicates of level = 0 in R is empty, clearly the fixpoint $M_0 = T^J_{R_0}\uparrow\omega(I_0) = T^J_{R_0}(I_0) = I_0$. Indeed, M_0 coincides with the set of ground EDB facts added to the EDB T that is incrementally constructed along the derivation. Moreover, M_0 would be a model for $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_k)\} \cup \forall F_0$ if the maximum level of the ordinary predicates in $\{G_0\}\cup F_0\cup R$ was 0. We show that the properties (i), (ii), (iii) and (iv) hold for this fixpoint:

(i) Obvious.

(ii) The proof is by contradiction. Suppose that $F_{i,j_i} = \leftarrow a_1(\bar{A}_1) \wedge \ldots \wedge a_p(\bar{A}_p) \wedge \neg b_1(\bar{B}_1) \wedge \ldots \wedge \neg b_q(\bar{B}_q) \wedge C_1 \wedge\ldots \wedge C_t \in F_i$ in the derivation θ, p+q+t ≥ 1, level(a_f) = 0, level(b_g) = 0 and there exists a ground variable substitution ρ such that $a_f(\bar{A}_f\rho) \in M_k$, $b_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true. If this hypothesis is true then we will show that the derivation must be failed leading, thus, to a contradiction. More precisely, we will see that sooner or later we get a (successor) condition (of F_{i,j_i}) containing just one literal with no CQC B-rule being applicable to.

First, suppose that F_{i,j_i} contains p+q+1 > 1 literals. Since the derivation is saturated, the condition $F_{m,j_m} = F_{i,j_i}$ will be the selected condition for some F_m, $m \geq i$, to apply some CQC B-rule. Let $P(F_{m,j_m})$ be the selected literal in F_{m,j_m}.

- Suppose that $P(F_{m,j_m}) = b(\bar{T})$ is a positive EDB atom. Since $b(\bar{T}\rho) \in M_0$, it means that $B(t\rho)$ appears in T_r, for each $r \geq h$, $h \geq 1$. The CQC-Expansion rule B2 is applied:

  If $m \geq h$, then there clearly exists a mgu μ for $b(\bar{T})$ and $b(\bar{T}\rho)$. Therefore $F_{m+1,j_{m+1}} = (F_{m,j_m}\backslash B(\bar{T}))\mu$ is a successor for F_{m,j_m} and there exists a ground variable substitution, say φ, such that $a_f(\bar{A}_f\mu\varphi) \in M_k$, $b_g(\bar{B}_g\mu\varphi) \notin M_k$ and $C_h\mu\varphi$ is true for each $a_f(\bar{A}_f\mu)$, $\neg b_g(\bar{B}_g\mu)$ and $C_h\mu$ in $F_{m+1,j_{m+1}}$.

  Otherwise, if $m < h$, since F_{m,j_m} is included in the condition set C_{m+1}, F_{m,j_m} will be re-checked later on after $b(\bar{T}\rho)$ being added to T_h.

- Suppose that $P(F_{m,j_m}) = \neg b(\bar{T})$ is a ground negative EDB literal. Since $b(\bar{T})$ must be ground, $b(\bar{T}) \notin M_0$ under the above hypothesis. The CQC B3-rule, the only one applicable here, offers to possibilities:

[3] $\mathbf{R}_k$ denotes the subset of $\mathbf{R}$ defining only the IDB predicates of level $\leq k$.

– Drop the whole condition F_{m,j_m}, i.e. no successor is added to F_{m+1}, and include the subgoal $b(\bar{T})$ in G_{m+1}. This option would contradict directly the above hypothesis because then $b(\bar{T}) \in POS$ and, thus, $b(\bar{T}) \in M_0$.

– F_{m,j_m} is substituted by two successor conditions in F_{m+1}: $F_{m+1,j_{m+1}} = \leftarrow b(\bar{T})$ and $F_{m+1,h_{m+1}} = F_{m,j_m} \backslash \neg b(\bar{T})$. Under the assumed hypothesis, $F_{m+1,j_{m+1}}$ will have no further successor whereas $a_f(\bar{A}_f\rho) \in M_k$, $a_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true for each $a_f(\bar{A}_f)$, $\neg b_g(\bar{B}_g)$ and C_h in $F_{m+1,h_{m+1}}$.

- Suppose that $P(F_{m,j_m}) = C$ is a ground built-in literal. C is true and the CQC B4 rule is applied. $F_{m+1,j_{m+1}} = (F_{m,j_m} \backslash C)\mu$ is the successor for F_{m,j_m} and there exists a ground variable substitution, say φ, such that $a_f(\bar{A}_f\rho) \in M_k$, $b_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true for each $a_f(\bar{A}_f)$, $\neg b_g(\bar{B}_g)$ and C_h in $F_{m+1,h_{m+1}}$.

If follows from this case analysis that if there exists a ground variable substitution ρ such that for each $a_f(\bar{A}_f)$, $\neg b_g(\bar{B}_g)$ and C_h in F_{i,j_i} $a_f(\bar{A}_f\rho) \in M_k$, $b_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true, then there exist al least one successor condition of F_{i,j_i} satisfying the same property, provided that F_{i,j_i} contains two ore more literals. It is straightforward to prove by induction that this property will hold along a chain of successor conditions ending with a successor condition containing just one literal. For this "final" condition, it is clear that the application of either the CQC B2-rule, the B3-rule or the B4-rule would obtain the empty condition [] if the CQC method were not ban explicitly such a possibility. Therefore, any of the CQC B2, B3 and B4 rules can be applied successfully on that just-one-literal condition and, thus, the derivation will be closed at the point that this later condition will be selected.

(iii) Let $q(\bar{S}) \in NEG$ and level(q) = 0. Since the derivation is saturated, the fully instantiated version of $q(\bar{S})$, say $q([\bar{S}])$, will be the selected literal in some goal G_h in the derivation to apply the CQC A4-rule. This rule will add the condition $\leftarrow q([\bar{S}])$ to F_{h+1}. By property (ii), $q([\bar{S}]) \notin M_0$.

(iv) Since the derivation is saturated, the fully instantiated version of $A_1\ \theta\ A_2$, say $[A_1]\ \theta\ [A_2]$, will be the selected literal in some goal G_h in the derivation to apply the CQC A4-rule. $[A_1]\ \theta\ [A_2]$ must be necessarily true.

As it is defined, M_0 is a model for $pComp(R) \cup \{\exists(L_1 \wedge \ldots \wedge L_k)\} \cup \forall F_0$ when the maximum level of the ordinary predicates in $\{G_0\} \cup F_0 \cup R$ is 0.

Now assume that we have a fixpoint M_{k-1} to level k-1 satisfying properties (i), (ii), (iii) and (iv), where $0 \leq k \leq n$. We define

$I_k = M_{k-1} \cup \{\ p([\bar{T}]) \mid p(\bar{T}) \in POS \text{ and level}(p) = k\ \}$.

We next show that $I_k \subseteq T^J_{R_k}(I_k)$. Suppose that $p([\bar{T}]) \in I_k$, where $p(\bar{T})$ appears in some goal G_i in the derivation. Because the derivation is saturated, there exists $m \geq 0$ such that $p(\bar{S}) = p(\bar{T}\sigma_{i+1}\ldots\sigma_{i+m})$ appears in goal G_{i+m} and $p(\bar{S})$ is the selected atom in G_{i+m} to apply the CQC-rule A1. Suppose that the input clause used in this CQC-step is a deductive rule $p(\bar{X}) \leftarrow r_1(\bar{U}_1) \wedge \ldots \wedge r_s(\bar{U}_s) \wedge \neg s_1(\bar{V}_1) \wedge \ldots \wedge \neg s_t(\bar{V}_t) \wedge C_1 \wedge \ldots \wedge C_z$. By the definition of $T^J_{R_k}$, it follows that $p([\bar{X}\sigma_{i+m+1}]) \in T^J_{R_k}(I_k)$, since each $r_j([\bar{U}_j\sigma_{i+m+1}]) \in M_{k-1} \subseteq I_k$; $s_h(\bar{V}_h\sigma_{i+m+1}) \in NEG$ and level(s_h) < k imply $s_h([\bar{V}_h\sigma_{i+m+1}]) \notin I_k$; and for each $C_i = A_1\ \theta\ A_2$, $[A_1\sigma_{i+m+1}]\ \theta\ [A_2\sigma_{i+m+1}]$ is true. Then using the fact that, for each j, $\sigma_1\ldots\sigma_j$ can be assumed to be idempotent, we have that $P([\bar{T}]) = P([\bar{T}\sigma_{i+1}\ldots\sigma_{i+m}]) = P([\bar{S}]) = P([\bar{S}\sigma_{i+m+1}]) = P([\bar{X}\sigma_{i+m+1}])$, so that $P([\bar{T}]) \in T^J_{R_k}(I_k)$. Thus $I_k \subseteq T^J_{R_k}(I_k)$.

Therefore, by [CL89, proposition 1], $M_k = T^J_{R_k}\uparrow\omega(I_k)$ is a fixpoint of $T^J_{R_k}$. We show that the properties (i), (ii), (iii) and (iv) also hold for this fixpoint:

(i) Obvious.

(ii) Again the proof is by contradiction. Suppose that $F_{i,j_i} = \leftarrow a_1(\bar{A}_1) \wedge \ldots \wedge a_p(\bar{A}_p) \wedge \neg b_1(\bar{B}_1) \wedge \ldots \wedge \neg b_q(\bar{B}_q) \wedge C_1 \wedge \ldots \wedge C_t \in F_i$ in the derivation θ, p+q+t ≥ 1, level(a_f) = 0, level(b_g) = 0 and there exists a ground variable substitution ρ such that $a_f(\bar{A}_f\rho) \in M_k$, $b_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true. If this property is true, we first prove that any successor condition obtained from F_{i,j_i} by applying some "consistency" CQC B-rule preserves such a property. Since θ is saturated, the condition

$F_{m,j_m} = F_{i,j_i}$ will be the selected condition for some F_m, m ≥ i, to apply some CQC B-rule. Let $R(F_{m,j_m})$ be the selected literal in F_{m,j_m}.

- Suppose that $P(F_{m,j_m}) = b(\bar{T})$ is a positive EDB atom. Since $b(\bar{T}\rho) \in M_k$, it means that $b(\bar{T}\rho)$ appears in T_r, for each r ≥ h, h ≥ 1. The CQC-Expansion rule B2 is applied:

  If m ≥ h, then there clearly exists a mgu μ for $b(\bar{T})$ and $b(\bar{T}\rho)$. Therefore $F_{m+1,j_{m+1}} = (F_{m,j_m} \setminus b(\bar{T}))\mu$ is a successor for F_{m,j_m} and there exists a ground variable substitution, say φ, such that $a_f(\bar{A}_f\mu\varphi) \in M_k$, $b_g(\bar{B}_g\mu\varphi) \notin M_k$ and $C_h\mu\varphi$ is true for each $a_f(\bar{A}_{ff}\mu)$, $\neg b_g(\bar{B}_g\ \mu)$ and $C_h\mu$ in $F_{m+1,j_{m+1}}$.

  Otherwise, if m < h, since F_{m,j_m} is included in the condition set C_{m+1}, F_{m,j_m} will be re-checked later on after $b(\bar{T}\rho)$ being added to T_h.

- Suppose that $P(F_{m,j_m}) = d(\bar{T})$ is a positive IDB atom. Since $d(\bar{T}\rho) \in M_k$, it means that there exists $d(\bar{R})$ appearing in some goal G_h in the derivation such that $d([\bar{R}]) = d(\bar{T}\rho)$. Because the derivation is saturated, there exists n ≥ 0 such that $d(\bar{S}) = d(\bar{R}\sigma_{h+1}\ldots\sigma_{h+n})$ appears in goal G_{h+n} and $d(\bar{S})$ is the selected atom in G_{h+n} to apply the CQC A1-rule using. the deductive rule $d(\bar{X}) \leftarrow r_1(\bar{U}_1) \wedge \ldots \wedge r_s(\bar{U}_s) \wedge \neg s_1(\bar{V}_1) \wedge \ldots \wedge \neg s_t(\bar{V}_t) \wedge L_1 \wedge\ldots \wedge L_z$ as input clause with mgu σ_{h+n+1}. As we have seen above, each $r_h([\bar{U}_h\sigma_{h+n+1}]) \in M_k$, each $s_j([\bar{V}_j\sigma_{h+n+1}]) \notin M_k$ and for each $L_i = A_1\ \theta\ A_2$, $[A_1\sigma_{i+m+1}]\ \theta\ [A_2\sigma_{i+m+1}]$ is true. Then note that $d([\bar{R}) = d([\bar{R}\sigma_{h+1}\ldots\sigma_{h+n}\sigma_{h+n+1}]) = d([\bar{X}\sigma_{h+n+1}]) = d(\bar{T}\rho)$. Hence, there exists a mgu, say μ, for $d(\bar{X})$ and $d(\bar{T})$ so that the successor condition $F_{m+1,j_{m+1}} = (F_{m,j_m}[d(\bar{T})\setminus r_1(\bar{U}_1) \wedge \ldots \wedge r_s(\bar{U}_s) \wedge \neg s_1(\bar{V}_1) \wedge \ldots \wedge \neg s_t(\bar{V}_t) \wedge L_1 \wedge\ldots \wedge L_z])\mu$ is obtained by applying the CQC B1-rule. Moreover, there exists a ground variable substitution, say φ, such that $d(\bar{T}\mu\varphi) = d(\bar{X}\mu\varphi) = d(\bar{T}\rho)$ and $a_f(\bar{A}_f\mu\varphi) \in M_k$, $b_g(\bar{B}_g\mu\varphi) \notin M_k$ and $C_h\mu\varphi$ is true for each $a_f(\bar{A}_{ff}\mu)$, $\neg b_g(\bar{B}_g\ \mu)$ and $C_h\mu$ in $F_{m+1,j_{m+1}}$.

- Suppose that $P(F_{m,j_m}) = \neg p(\bar{T})$ is a ground negative ordinary literal. Since $p(\bar{T})$ must be ground, $p(\bar{T}) \notin M_k$ under the above hypothesis. The CQC B3-rule, the only one applicable here, offers to possibilities:

  – Drop the whole condition F_{m,j_m}, i.e. no successor is added to F_{m+1}, and include the subgoal $p(\bar{T})$ in G_{m+1}. This option would contradict directly the above hypothesis because then $p(\bar{T}) \in POS$ and, thus, $p(\bar{T}) \in M_k$.

  – F_{m,j_m} is substituted by two successor conditions in F_{m+1}: $F_{m+1,j_{m+1}} = \leftarrow p(\bar{T})$ and $F_{m+1,h_{m+1}} = F_{m,j_m}\setminus\neg p(\bar{T})$. Under the assumed hypothesis, $F_{m+1,j_{m+1}}$ will have no further successor whereas $a_f(\bar{A}_f\rho) \in M_k$, $a_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true for each $a_f(\bar{A}_f)$, $\neg b_g(\bar{B}_g)$ and C_h in $F_{m+1,h_{m+1}}$.

If follows from this case analysis that if there exists a ground variable substitution ρ such that for each $a_f(\bar{A}_f)$, $\neg b_g(\bar{B}_g)$ and C_h in F_{i,j_i} $a_f(\bar{A}_f\rho) \in M_k$, $b_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true, then there exist al least one successor condition of F_{i,j_i} satisfying the same property. It is straightforward to prove by induction that this property will hold along a chain of successor conditions.

On the other hand, since the derivation is saturated and the predicate definitions are hierarchical, in some point of this chain there must be a successor condition $F_{i+n,j_{i+n}} = \leftarrow a_1(\bar{X}_1) \wedge \ldots \wedge a_s(\bar{X}_s) \wedge \neg b_1(\bar{Y}_1) \wedge \ldots \wedge \neg b_t(\bar{Y}_t) \wedge L_1 \wedge\ldots \wedge L_z$ where level(a_h) ≤ k-1 and level(b_g) ≤ k-1 for each 1 ≤ h ≤ s and 1 ≤ g ≤ t. We know that there exist no ground variable substitution ρ such that $a_h(\bar{X}_h\rho) \in M_{k-1}$, $b_g(\bar{Y}_g\rho) \notin M_{k-1}$ and $L_i\rho$is true and, hence, it is impossible that there is a ground variable substitution φ such that $a_h(\bar{X}_h\ \varphi) \in M_k$, $b_g(\bar{Y}_g\ \varphi) \notin M_k$ and $L_i\varphi$ for each 1 ≤ h ≤ s, 1 ≤ l ≤ t and 1 ≤ h ≤ z. Therefore the above hypothesis have led us to a contradiction an we must conclude that if $F_{i,j_i} = \leftarrow a_1(\bar{A}_1) \wedge \ldots \wedge a_p(\bar{A}_p) \wedge \neg b_1(\bar{B}_1) \wedge \ldots \wedge \neg b_q(\bar{B}_q) \wedge C_1 \wedge\ldots \wedge C_t \in F_i$, p+q+t ≥ 1, and level($a_f$) ≤ k and level($b_g$) ≤ k, then there exist no ground variable substitution ρ such that $a_f(\bar{A}_f\rho) \in M_k$, $a_g(\bar{B}_g\rho) \notin M_k$ and $C_h\rho$ is true.

(iii) Let $q(\bar{S}) \in NEG$ and level(q) = 0. Since the derivation is saturated, the fully instantiated version of $q(\bar{S})$, say $q([\bar{S}])$, will be the selected literal in some goal G_h in the derivation to apply the CQC A4-rule. This rule will add the condition $\leftarrow q([\bar{S}])$ to F_{h+1}. By property (ii), $q([\bar{S}]) \notin M_k$.

(iv) Since the derivation is saturated, the fully instantiated version of $A_1\ \theta\ A_2$, say $[A_1]\ \theta\ [A_2]$, will be the selected literal in some goal G_h in the derivation to apply the CQC A4-rule. $[A_1]\ \theta\ [A_2]$ must be necessarily true.

Thus, properties (i), (ii), (iii) and (iv) hold for all levels k, where $0 \le k \le n$.

If the initial goal G_0 has the form $\leftarrow A_1 \wedge \ldots \wedge A_m \wedge \neg B_1 \wedge \ldots \wedge \neg B_p \wedge C_1 \wedge \ldots \wedge C_q$, by the definition of *POS*, *NEG* and *BI* then each $A_i \in POS$, each $B_j \in NEG$ and each $C_h \in BI$. Thus, by theorem 4.1 and properties (i), (ii), (iii) and (iv), M_n is a model for $pComp(R) \cup \{\exists(A_1 \wedge \ldots \wedge A_m \wedge \neg B_1 \wedge \ldots \wedge \neg B_p \wedge C_1 \wedge \ldots \wedge C_q)\} \cup \forall F_0$. □

Theorem 4.8 (*Failure completeness*)
Let Q_1 and Q_2 be two queries, *DR* be the set of deductive rules defining the database IDB relations and $R = DR \cup Q_1 \cup Q_2$. Let also $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ be the initial goal, $F_0 = \varnothing\ [\cup IC]$ be the initial set of conditions to enforce and *K* be the set of constants appearing in $DR \cup Q_1 \cup Q_2 \cup F_0$. If $Q_1 \sqsubseteq Q_2$ $(Q_1 \sqsubseteq_{IC} Q_2)$ then every fair CQC-Tree rooted at $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ is finitely failed provided that $\{G_0\} \cup F_0 \cup DR \cup Q_1 \cup Q_2$ is safe and hierarchical.

Proof: Since $Q_1 \sqsubseteq Q_2$ $(Q_1 \sqsubseteq_{IC} Q_2)$, we have that $pComp(R) \cup\ \forall F_0 \models \forall X_1,...,X_n\ q_1(X_1, ..., X_n) \rightarrow \neg q_2(X_1, ..., X_n)$, and consequently $pComp(R) \cup\ \forall F_0 \models \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$, where $R = IDB \cup Q_1 \cup Q_2$. Suppose that there exists a fair CQC-Tree rooted at $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ that is not finitely failed. In other words, there exists a fair CQC-Tree rooted at $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ with a saturated open derivation. However, it would mean that $pComp(R) \cup \{\exists X_1,...,X_n\ q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)\} \cup \forall F_0$ has a model according to Lemma 4.7. Therefore, no fair CQC-Tree rooted at $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ can have a saturated open derivation. □

4.4 Finite Success Soundness

The following theorem is a corollary of Lemma 4.7.

Theorem 4.9 (*Finite success soundness*)
Let Q_1 and Q_2 be two queries, *DR* be the set of deductive rules defining the database IDB relations and $R = DR \cup Q_1 \cup Q_2$. Let also $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ be the initial goal, $F_0 = \varnothing\ [\cup IC]$ be the initial set of conditions to enforce and *K* be the set of constants appearing in $DR \cup Q_1 \cup Q_2 \cup F_0$. If there exists a finite successful CQC-derivation starting from $(G_0\ F_0\ \varnothing\ \varnothing\ K)$ then $Q_1 \not\sqsubseteq Q_2$ $(Q_1 \not\sqsubseteq_{IC} Q_2)$ provided that $\{G_0\} \cup F_0 \cup DR \cup Q_1 \cup Q_2$ is safe and hierarchical.

Proof: Since every finite successful CQC-derivation is fair by definition, according to Lemma 4.7, there exists a model for $pComp(R) \cup \{\exists X_1,...,X_n\ q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)\} \cup \forall F_0$, where $R = DR \cup Q_1 \cup Q_2$. Therefore, $pComp(R) \cup \forall F_0 \not\models \forall X_1,...,X_n\ q_1(X_1, ..., X_n) \rightarrow q_2(X_1, ..., X_n)$ and thus $Q_1 \not\sqsubseteq Q_2$ $(Q_1 \not\sqsubseteq_{IC} Q_2)$. □

4.5 Finite Success Completeness

In this Section, we relate the completeness of the CQC Method to that of the SLDNF resolution. This Section is divided in two subsections. Section 4.5.1 proves the completeness of the CQC method for these cases where the Negation or Dense Order VIPs are required. Section 4.5.2 proves that the CQC method is also complete for these cases where the Simple VIP is applicable.

4.5.1 Finite Succees Completeness with the Negation and Dense Order VIPs

We prove first lemmas 4.10, 4.11 and 4.12, which are the basis for Theorem 4.13. We use the concept of rank of an SLDNF refutation and rank of a finitely failed SLDNF tree as it is defined in [Llo87]. We also introduce a function *constants*: logic expression → set of constants, such that it returns the set of constants that appear in a given logic expression.

Lemma 4.10

Let G a goal; F, F_1, F_2, C and C' condition sets; T and T' EDBs; and K and K' sets of constant values. The following results hold:

1. If there exists a CQC-derivation from ($G \varnothing T\ C\ K$) to ([] $\varnothing\ T'\ C'\ K'$), then there exists a CQC-derivation from. ($G\ F\ T\ C\ K$) to ([] $F\ T'\ C'\ K'$).
2. If there exists a CQC-derivation from ([] $F\ T\ C\ K$) to ([] $\varnothing\ T'\ C'\ K'$), then there exists a CQC-derivation from. ($G\ F\ T\ C\ K$) to ($G \varnothing\ T'\ C'\ K'$).
3. If there exists a CQC-derivation from ([] $F_1\ T\ C\ K$) to ([] $\varnothing\ T'\ C'\ K'$), then there exists a CQC-derivation from. ([] $F_1 \cup F_2\ T\ C\ K$) to ([] $F_2\ T'\ C'\ K'$).

Proof: Let us prove 1. We would use the same reasoning for 2 and 3. If there exists a CQC-derivation from ($G \varnothing\ T\ C\ K$) to ([] $\varnothing\ T'\ C'\ K'$), at least there exists the CQC-derivation from. ($G\ F\ T\ C\ K$) to ([] $F\ T'\ C'\ K'$) that we would obtain by applying one to one the same CQC-steps with the same literal-selection rule.

Lemma 4.11
Let R be a set of deductive rules, G a goal, T, T' and T" EDBs, F, C, C' and C" condition sets and K, K' and K" sets of constants. Then, the two following results hold:

- Considering an SLDNF refutation of rank n of $R \cup T \cup \{G\}$ then
  - $\forall\ T'$ such that $T' \subseteq T$, $constants(R \cup T') = K' \subseteq K = constants(R \cup T)$, and
  - $\forall\ C'$ such that $\forall\ C \in C'$ the SLDNF tree for $R \cup T \cup \{C\}$ fails finitely and has rank n-1,

  there exists a CQC-derivation from ($G \varnothing\ T'\ C'\ K'$) to ([] $\varnothing\ T''\ C''\ K''$) such that:
  - $T'' \subseteq T$, $constants(R \cup T'') = K'' \subseteq K$, and
  - $\forall\ C \in C''$, the SLDNF tree for $R \cup T'' \cup \{C\}$ fails finitely and has rank n-1.
- Considering a finitely failed SLDNF tree of rank n for $R \cup T \cup F$ then
  - $\forall\ T'$ such that $T' \subseteq T$, $constants(R \cup T') = K' \subseteq K = constants(R \cup T)$, and
  - $\forall\ C'$ such that $\forall\ C \in C'$ the SLDNF tree for $R \cup T \cup \{C\}$ fails finitely and has rank n-1

  there exists a CQC-derivation from ([] $F\ T'\ C'\ K'$) to ([] $\varnothing\ T''\ C''\ K''$) such that:
  - $T'' \subseteq T$, $constants(R \cup T'') = K'' \subseteq K$, and
  - $\forall\ C \in C''$- C', the SLDNF tree for $R \cup T \cup \{C\}$ fails finitely and has rank n.

Proof: the proof is by induction over the rank n of the refutation or the finitely failed tree.
Case n = 0:

1) In this case, the goal G contains only positive literals, and we have to show that there exists a CQC-derivation from ($G \varnothing\ T' \varnothing\ K'$) to ([] $\varnothing\ T'' \varnothing\ K''$), for any $T' \subseteq T$. The idea is to associate to each step in the SLDNF refutation just one CQC-step from ($G_i \varnothing\ T_i \varnothing\ K_i$) to ($G_i \varnothing\ T_{i+1} \varnothing\ K_{i+1}$), and prove that $T_{i+1} \subseteq T$. Initially, $G_0 = G$, $T_0 = T'$ and $K_0 = K' = constants(R \cup T')$.

   Let L_j be the selected literal in the refutation. The step applied in the CQC-derivation depends on the type of L_j, as follows:

   - L_j is a IDB atom. We apply A1) and obtain ($S \varnothing\ T_i \varnothing\ K_i$).
   - L_j is a an EDB atom instantiated by SLDNF with substitution σ. From Lemma 4.2, there exists a variable instantiation procedure from ($\bar{X} \varnothing\ K_i$) to ($\varnothing\ \sigma\ K_{i+1}$) where $\bar{X}$ is the set of nonground variables in L_j. Therefore, we apply A2) and get (($G_i \backslash L_j)\sigma \varnothing\ T_i \cup \{L_j\sigma\} \varnothing\ K_{i+1}$). Note again that $L_j\sigma \in T$ and then $T_i \cup \{L_j\sigma\} \subseteq T$. Note also that $constants(R \cup T_i \cup \{L_j\sigma\}) = K_{i+1} \subseteq K = constants(R \cup T)$.
   - L_j is a ground evaluable literal that is evaluated true. We apply A4) and get ($G_i \backslash L_j \varnothing\ T_i \varnothing\ K_i$)

   The derivation ends with the empty clause [].

2) As before, the conditions of F contain only positive literals, and we have to show that there exists a CQC-derivation from ([] F T' $\varnothing$ K') to ([] $\varnothing$ T'' C'' K''), for any $T' \subseteq T$. Again, the idea is to associate to each step in the SLDNF tree just one CQC-step from ([] F_i T_i C_i K_i) to ([] F_{i+1} T_{i+1} C_{i+1} K_{i+1}), and prove that $T_{i+1} \subseteq T$ and $\forall\ C \in C_{i+1}$ the SLDNF tree for $R \cup T \cup \{C\}$ fails finitely and has rank 0. Initially, $F_0 = F$, $T_0 = T'$, $C_0 = \varnothing$ and $K_0 = K' = constants(R \cup T')$. Note that, in general, F_i is a set of goals that corresponds to a subset of the nodes of the SLDNF tree.

Let $H_i = \leftarrow L_1 \wedge \ldots \wedge L_k$ be a node in the SLDNF tree and L_j the selected literal. Let $F_i = \{H_i\} \cup F'_i$. The step applied in the CQC-derivation depends on the type of L_j, as follows:

- L_j is a IDB atom and S' is the set of all resolvents of clauses in R. We apply B1) and obtain ([] $S' \cup F'_i$ T_i C_i K_i).
- L_j is a EDB atom and S' is the set, maybe $\varnothing$, of all resolvents of clauses in T_i, we apply B2) and get ([] $S' \cup F'_i$ T_i C_{i+1} K_i), where $C_{i+1} = C_i$ if L_j is ground and $L_j \in T_i$, otherwise $C_{i+1} = C_i \cup \{H_i\}$. In this later case, the SLDNF tree for $R \cup T \cup \{H_i\}$ has rank 0 and fails finitely since H_i is the current node.
  Note that if [] $\in S'$, the tree would not be failed. Note also that if $T_i \subset T$, the SLDNF tree would contain additional nodes, but all of them would end with a failure.
- L_j is a ground evaluable literal that is evaluated true. We apply B4) and get ([] $\{H_i \backslash L_j\} \cup F'_i$ T_i C_i K_i). Note that if k=1, the tree would not be failed.
- L_j is a ground evaluable literal that is evaluated false. We apply B5) and get ([] F'_i T_i C_i K_i).

The derivation ends with $\varnothing$.

General case: Assume that the result holds for SLDNF refutations and finitely failed SLDNF trees of rank n-1. We are going to prove that it also holds for refutations and finitely failed trees of rank n.

3) In this case, the goal G may contain positive and negative literals, and we have to show that there exists a CQC-derivation from (G $\varnothing$ T' C' K') to ([] $\varnothing$ T'' C'' K''), for any $T' \subseteq T$. The idea now is to associate to each step in the SLDNF refutation a finite CQC-(sub)derivation from (G_i $\varnothing$ T_i C_i K_i) to (G_{i+1} $\varnothing$ T_{i+m} C_{i+m} K_{i+m}), m $\geq$ 1, and prove that $T_{i+m} \subseteq T$ and $\forall\ C \in C_{i+m}$, the SLDNF tree for $R \cup T_{i+m} \cup \{C\}$ fails finitely and has rank n-1.

Let L_j be the selected literal in the refutation. The steps applied in the CQC-derivation depend on the type of L_j, as follows:

<u>L_j is positive:</u>

- L_j is a IDB atom. We apply A1) and obtain (S $\varnothing$ T_i C_i K_i).
- L_j is an EDB atom instantiated by SLDNF with substitution σ. From Lemma 4.2, there exists a variable instantiation procedure from ($\bar{X}$ $\varnothing$ K_i) to ($\varnothing$ σ K_{i+1}) where $\bar{X}$ is the set of nonground variables in L_j. Therefore, we apply A2) and:
  - if $L_j\sigma \in T_i$, we get (($G_i \backslash L_j$)σ $\varnothing$ T_i C_i K_{i+1}).
  - if $L_j\sigma \notin T_i$, we add $L_j\sigma$ to T_i and get (($G_i \backslash L_j$)σ C_i $T_i \cup \{L_j\sigma\}$ $\varnothing$ K_{i+1}). Since the SLDNF tree for $R \cup T \cup C_i$ fails finitely and has rank n-1, then, by induction hypothesis, there exists a CQC-derivation from ([] C_i $T_i \cup \{L_j\sigma\}$ $\varnothing$ K_{i+1}) to ([] $\varnothing$ T_{i+m} C_{i+m} K_{i+m}) with $T_{i+m} \subseteq T$ and for each $C \in C_{i+m} - C_i$, the SLDNF tree for $R \cup T_{i+m} \cup \{C\}$ fails finitely and has rank n-1. Therefore, according to Lemma 4.10, there exists a CQC-derivation from (($G_i \backslash L_j$)σ C_i $T_i \cup \{L_j\sigma\}$ $\varnothing$ K_{i+1}) to (($G_i \backslash L_j$)σ $\varnothing$ T_{i+m} C_{i+m} K_{i+m}).
- L_j is a ground evaluable literal that is evaluated true. We apply A4) and get ($G_i \backslash L_j$ $\varnothing$ T_i C_i K_i)

<u>L_j is negative</u>: We apply A3) and get ($G_i \backslash L_j$ $\{\leftarrow \neg L_j\}$ T_i C_i K_i). Since the SLDNF tree for $R \cup T \cup \{\leftarrow \neg L_j\}$ fails finitely and has rank n-1, by the induction hypothesis, there exists a CQC-derivation from ([] $\{\leftarrow \neg L_j\}$ T_i C_i K_i) to ([] $\varnothing$ T_{i+m} C_{i+m} K_{i+m}) where T_{i+m} and C_{i+m} satisfy the conditions of the lemma. Therefore, according to Lemma 4.10, there exists a CQC-derivation from ($G_i \backslash L_j$ $\{\leftarrow \neg L_j\}$ T_i C_i K_i) to ($G_i \backslash L_j$ $\varnothing$ T_{i+m} C_{i+m} K_{i+m}).

The derivation ends with [].

4) Now the conditions of F may contain positive and negative literals, and we have to show that there exists a CQC-derivation from ([] F T' C' K') to ([] ∅ T'' C'' K''), for any $T' \subseteq T$. Again, the idea is to associate to each step in the SLDNF tree a finite CQC-(sub)derivation from ([] F_i T_i C_i K_i) to ([] F_{i+l} T_{i+m} C_{i+m} K_{i+m}), m ≥ 1, and prove that $T_{i+m} \subseteq T$ and $\forall\, C \in C_{i+m} - C_i$, the SLDNF tree for $R \cup T \cup \{C\}$ fails finitely and has rank n.

Let $H_i = \leftarrow L_1 \wedge \ldots \wedge L_k$ be a node in the SLDNF tree and L_j the selected literal. Let $F_i = \{H_i\} \cup F'_i$. The steps applied in the CQC-(sub)derivation depends on the type of L_j, as follows:

L_j is positive:

- L_j is a IDB atom and S' is the set of all resolvents of clauses in R. We apply B1) and obtain ([] $S' \cup F'_i$ T_i C_i K_i).
- L_j is a EDB atom and S' is the set, maybe ∅, of all resolvents of clauses in T_i, we apply B2) and get ([] $S' \cup F'_i$ T_i C_{i+l} K_i), where $C_{i+l} = C_i$ if L_j is ground and $L_j \in T_i$, otherwise $C_{i+l} = C_i \cup \{H_i\}$. The SLDNF tree for $R \cup T \cup \{H_i\}$ has rank n and fails finitely since H_i is the current node. Note that if $T_i \subset T$, the SLDNF tree would contain additional nodes, but all of them would end with a failure.
- L_j is a ground evaluable literal that is evaluated true. We apply B4) and get ([] $\{H_i \backslash L_j\} \cup F'_i$ T_i C_i K_i). Note that if k=1, the tree would not be failed.
- L_j is a ground evaluable literal that is evaluated false. We apply B5) and get ([] F'_i T_i C_i K_i).

L_j is negative:

- Suppose that the SLDNF tree for $R \cup T \cup \{\leftarrow \neg L_j\}$ fails finitely and has rank n-1. We apply B3) and get ([] $\{\leftarrow \neg L_j\} \cup \{H_i \backslash L_j\} \cup F'_i$ T_i C_i K_i). By the induction hypothesis, there exists a CQC-derivation from ([] $\{\leftarrow \neg L_j\}$ T_i C_i K_i) to ([] ∅ T_{i+m} C_{i+m} K_{i+m}) where T_{i+m} and C_{i+m} satisfy the conditions of the lemma. Therefore, according to Lemma 4.10, there exists a CQC-derivation from ([] $\{\leftarrow \neg L_j\} \cup \{H_i \backslash L_j\} \cup F'_i$ T_i C_i K_i) to ([] $\{H_i \backslash L_j\} \cup F'_i$ T_{i+m} C_{i+m} K_{i+m}).
- Suppose that there is SLDNF refutation for $R \cup T \cup \{\leftarrow \neg L_j\}$ with rank n-1. We apply B3) and get ($\leftarrow \neg L_j$ F'_i T_i C_i K_i). Then, applying the induction hypothesis, there exists a CQC-derivation from ($\leftarrow \neg L_j$ ∅ T_i C_i K_i) to ([] ∅ T_{i+m} C_{i+m} K_{i+m}) where T_{i+m} and C_{i+m} satisfy the conditions of the lemma. Therefore, according to Lemma 4.10, there exists a CQC-derivation from ($\leftarrow \neg L_j$ F'_i T_i C_i K_i) to ([] F'_i T_{i+m} C_{i+m} K_{i+m}).

The derivation ends with ∅. □

Lemma 4.12

Let R be a set of deductive rules, F be a set of conditions, G be a goal, T and T' be EDBs and K be the set of constant values appearing in $R \cup F \cup G$. If there exists a SLDNF refutation of $R \cup T \cup \{G\}$ and every SLDNF-tree for $R \cup T \cup F$ fails finitely, then there exists a CQC-derivation from (G F ∅ ∅ K) to ([] ∅ T' C K'), using the Negation or Dense Order VIP, where $T \subseteq T'$.

Proof: It is a direct consequence of applying Lemmas 4.10 and 4.11. □

Theorem 4.13 (*Finite success completeness with the Negation and Dense Order VIPS*)

Let Q_1 and Q_2 be two queries, DR be the set of deductive rules defining the database IDB relations and $R = DR \cup Q_1 \cup Q_2$. Let also $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ be the initial goal, $F_0 = ∅$ [$\cup IC$] be the initial set of conditions to enforce and K be the set of constants appearing in $DR \cup Q_1 \cup Q_2 \cup F_0$. If $Q_1 \not\equiv Q_2$ (or $Q_1 \not\equiv_{IC} Q_2$) then there exists a successful CQC-derivation from (G_0 F_0 ∅ ∅ K) to ([] ∅ T C K') provided that $\{G_0\} \cup F_0 \cup DR \cup Q_1 \cup Q_2$ is safe and either hierarchical or strict-stratified [CL89].

Proof: $Q_1 \not\equiv Q_2$ (or $Q_1 \not\equiv_{IC} Q_2$) if there exists a model M for $pComp(R)$ [$\cup \forall IC$] $\cup$ $\{\exists X_1 \ldots X_n\ q_1(X_1,\ldots,X_n) \wedge \neg q_2(X_1,\ldots,X_n)\}$, where $R = DR \cup Q_1 \cup Q_2$. Let $T_M \subseteq M$ the set of all EDB facts belonging to M. Then M is a model also for $Comp(R \cup T_M) \cup \forall F_0 \cup \{\exists X_1 \ldots X_n\ q_1(X_1,\ldots,X_n) \wedge \neg q_2(X_1,\ldots,X_n)\}$. From the completeness of the SLDNF resolution for databases and goals that are safe and either hierarchical or strict-stratified [CL89], it follows that there exists an SLDNF refutation of $R \cup T_M \cup \{G_0\}$ and that every SLDNF-tree for $R \cup T_M \cup F_0$ fails finitely. Hence, by Lemma 4.12, there is a CQC-derivation from (G_0 F_0 ∅ ∅ K) to ([] ∅ T C K') with $T \subseteq T_M$. □

4.5.2 Completeness with the Simple VIP

In this Section, we suppose that the Simple VIP is applicable because following conditions holds at a time:

- Integrity constraints are not considered.
- Deductive rules do not have any negative or built-in literals in their rule bodies.
- Deductive rules do not have constants in their heads. Note that a rule of the form $d(X, k) \leftarrow B_1 \wedge \ldots \wedge B_m$, for instance, is equivalent to $D(X, Y) \leftarrow B_1 \wedge \ldots \wedge B_m \wedge Y = k$.
- Deductive rules do not have any variable appearing twice or more times in their heads. Note that a rule of the form $d(X, X) \leftarrow B_1 \wedge \ldots \wedge B_m$, for instance, is equivalent to $d(X, Y) \leftarrow B_1 \wedge \ldots \wedge B_m \wedge Y = X$.

Assume that there exists an SLDNF refutation for $R \cup T_x \cup \{G_0\}$, where $R = DR \cup Q_1 \cup Q_2$ and $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ Since there is just a negative literal, $\neg q_2(X_1, ..., X_n)$, such an SLDNF refutation has rank 1. We prove in this Section (Theorem 4.16) that there will be a CQC-derivation from $(G_0 \varnothing \varnothing \varnothing K_0)$ to $([] \varnothing T C K)$ such that $T\mu \subseteq T_x$, where K_0 is the initial set of constants appearing in R and μ a complete mapping, defined below, from K to constants(T_x).

Let $K = \{k_1, \ldots, k_n\}$ and K' be sets of constants. A *complete mapping* μ *from K to K'* is a set of the form $\{k_1/q_1, \ldots, k_n/q_m\}$ where $n \geq m$ and each $q_i \in K'$.

Let E be an expression and $\mu = \{k_1/q_1, \ldots, k_n/q_m\}$ a complete mapping μ from *constants*(E) to some set of constants K'. Then $E\mu$ is the expression obtained from E by simultaneously replacing each occurrence of k_i in E by q_j iff $k_1/q_1 \in \mu$. If μ and ν are two complete mappings then $E\mu\nu = (E\mu)\nu$.

Lemma 4.15
Let R be a set of deductive rules, F and F' condition sets, T and T' EDBs, K = *constants*(T') and μ a complete mapping from K to *constants*(T). If there is a finite failed SLDNF-tree of rank 0 for $R \cup T \cup F$, then $\forall T'$, $\forall C$ and $\forall \mu$ such that $T'\mu \subseteq T$ and $F'\mu = F$, there exist a CQC-derivation from $([] F' T' C K)$ to $([] \varnothing T' C' K)$. Moreover, $\forall C_i \in (C'\text{-}C)$ the SLDNF tree for $R \cup T \cup \{C_i\mu\}$ fails finitely and has rank 0.

Proof: The idea is to associate to each SLDNF step a corresponding CQC-step from $([] F_i T C_i K)$ to $([] F_{i+1} T C_{i+1} K)$, and prove that if a new condition C_i is added to C_{i+1}, $C_{i+1} = C_i \cup \{C_i\}$, then the SLDNF tree for $R \cup T \cup \{C_i\mu\}$ fails finitely and has rank 0. Initially, $F_0 = F'$ and $C_0 = C$. Note that, in general, F_i is a set of conditions that corresponds to a subset of the nodes of the SLDNF tree.

Let $H_i = \leftarrow A_1 \wedge \ldots \wedge A_k$ be a node in the SLDNF tree, A_j its selected atom and $S = \{H_{i+1}, \ldots, H_{i+p}\}$ the set of its child nodes. Each H_{i+u} in S is the resolvent of H_i with a clause $C_u = A_u \leftarrow B_{u,1} \wedge \ldots \wedge B_{u,q}$, belonging either to R ($q \geq 1$) or to T' ($q = 0$), on A_j with mgu $\theta_{i,u}$.

Let $F_i = \{H'_i\} \cup F'_i$. The step applied in the CQC-derivation depends on the type of the selected atom A'_j in H'_i, where $A'_j\mu = A_j$ and $H'_i\mu = H_i$.

- A'_j is a IDB atom. Since all the terms in A_u are distinct variables it will always exist a mgu $\theta'_{i,u}$ of A'_j and A_u. Therefore we apply B1) and obtain $F_{i+1} = S' \cup F'_i$ and $C_{i+1} = C_i$. $S' = \{H'_{i+1}, \ldots, H'_{i+p}\}$ and each H'_{i+u} is the resolvent of H'_i with the clause C_u on A'_j with mgu $\theta'_{i,u}$. Note that if $\theta'_{i,u}$ is the mgu of A'_j and A_u then $\theta'_{i,u}\mu$ is the mgu of $A'_j\mu$ and A_u. Then $\theta'_{i,u}\mu = \theta_{i,u}$ since $A'_j\mu = A_j$. Therefore $S'\mu = S$.
- A'_j is a EDB atom and S' is the set of all resolvents of clauses in T_i, we apply B2) and get $F_{i+1} = S' \cup F'_i$ and $C_{i+1} = C_i$ if L_j is ground and $L_j \in T_i$, otherwise $C_{i+1} = C_i \cup \{H'_i\}$. $S' = \{H'_{i+1}, \ldots, H'_{i+t}\}$ and each H'_{i+w} is the resolvent of H'_i with a clause $\leftarrow A_w$ of T_i on A'_j with mgu $\theta'_{i,w}$. Note that if $\theta'_{i,w}$ is the mgu of A'_j and A_w then $\theta'_{i,w}\mu$ is the mgu of $A'_j\mu$ and $A'_w\mu$. Then for each $\theta'_{i,w}$ exists a $\theta_{i,u}$ such that $\theta'_{i,w}\mu = \theta_{i,u}$ since $\theta_{i,u}$ is the mgu of A_j and some $A_u \in T$, $A_j = A'_j\mu$ and $A_u = A_w\mu \in T_i\mu \subset T$. Therefore $S'\mu \subseteq S$. Moreover, the SLDNF tree for $R \cup T \cup \{H'_i\mu\}$ fails finitely since $H'_i\mu = H_i$ beeing the current node. If $S'\mu \subset S$, the SLDNF tree may contain additional nodes, but all of them will end with a failure. Note that if $[] \in S'$ then $[] \in S$ too and the tree would not be failed.

– A'$_j$ is a EDB atom and there is no clause in T_i that can be unified with A'$_j$, we apply B3) and get F_{i+1} = F'_i and C_{i+1} = $C_i \cup \{H'_i\}$. As before, the SLDNF tree for $R \cup T \cup \{H'_i\mu\}$ fails finitely. Moreover, if $T_i\mu \subset T$, the SLDNF tree may contain additional nodes, but all of them will end with a failure.

The derivation ends with ∅. □

Lemma 4.16

Let R be a set of deductive rules; G a goal; T, T' and T'' EDBs; C, C' and C'' condition sets; K, K' and K'' sets of constants; and μ and ρ complete mappings. Let an SLDNF-refutation of rank 1 for $R \cup T \cup \{G\}$ then

– $\forall T'$ and $\forall\mu$ such that $T'\mu \subseteq T$, where μ is a complete mapping from *constants*(T') = K' to K = *constants*(T), and

– $\forall C'$ such that the SLDNF tree for $R \cup T \cup \{C'\mu\}$ fails finitely and has rank 0

there exists a CQC-derivation from (G' ∅ T' C' K') to ([] ∅ T'' C'' K'') such that:

– $G'\mu = G$,

– $T''\rho \subseteq T$, where ρ is a complete mapping from *constants*(T'') = K'' to K, and

– the SLDNF tree for $R \cup T''\rho \cup \{C''\rho\}$ fails finitely and has rank 0.

Proof: Again, the idea is to associate to each SLDNF-refutation step a corresponding CQC-(subderivation)derivation from (G'_i ∅ T_i C_i K_i) to (G'_{i+1} ∅ T_{i+1} C_{i+m} K_{i+1}), m ≥ 1, and prove that if $T_i\mu_i \subseteq T'$ then $T_{i+1}\mu_{i+1} \subseteq T'$. Moreover, if $C_i \subset C_{i+m}$, we must prove that SLDNF tree for $R \cup T_{i+1}\mu_{i+1} \cup \{C_{i+m}\mu_{i+1}\}$ fails finitely and has rank 0. Initially, $G'_0 = G'$, $T_0 = T'$, $C_0 = C'$, $K_0 = K'$ and $\mu_0 = \mu$.

Let $G_i = \leftarrow L_1 \wedge \ldots \wedge L_j \wedge \ldots \wedge L_m$, m ≥ 1, and L_j be the current goal and its selected atom, respectively, in the SLDNF-refutation step. The step applied in the CQC-derivation depends on the type of the selected literal L'$_j$ in G'_i, where L'$_j\mu_i$ = L_j and $G'_i\mu_i = G_i$.

L$_j$ and L'$_j$ are positive. Then $G_{i+1} = \leftarrow L_1\theta_i \wedge \ldots \wedge L_{j-1}\theta_i \wedge B_1\theta_i \wedge \ldots \wedge B_p\theta_i \wedge L_{j+1}\theta_i \wedge \ldots \wedge L_m\theta_i$, where θ_i is a mgu of L_j and H and H ← $B_1 \wedge \ldots \wedge B_p$, belongs either to R (p ≥1) or to T (p = 0).

– L'$_j$ is a IDB atom. Since all variables the terms in H are distinct variables, where H ← $B_1 \wedge \ldots \wedge B_p$ belongs to R (p ≥1), it will always exist a mgu θ'_i of L'$_j$ and H. Therefore we apply A1) and get $G'_{i+1} = \leftarrow L'_1\theta_i \wedge \ldots \wedge L'_{j-1}\theta_i \wedge B_1\theta_i \wedge \ldots \wedge B_p\theta_i \wedge L'_{j+1}\theta_i \wedge \ldots \wedge L'_m\theta_i$. And $T_{i+1} = T_i$, $C_{i+1} = C_i$, and $K_{i+1} = K_i$. Note that if θ'_i is the mgu of L'$_j$ and H then $\theta'_i\mu_i$ is the mgu of L'$_j\mu_i$ and H. Then $\theta'_i\mu_i = \theta_i$ since L'$_j\mu_i$ = L_j. Therefore if $\mu_{i+1} = \mu_i$ then $G'_{i+1}\mu_{i+1} = G_{i+1}$.

– L'$_j$ is a ungrounded EDB atom and there exists a variable instantiation procedure from (x ∅ K_i) to (∅ σ K_{i+1}) using the Simple VIP, where x is the set of ungrounded variables in A$_j$. Let us define the complete mapping ν = {k_1/q_1, …, k_n/q_m} from K_{i+1} to *constants*(θ_1) such that $k_i/q_j \in \nu$ if $x_g/k_i \in \sigma$ and $x_g/q_j \in \theta_1$. So μ_{i+1} will be $\mu_i\nu$ (= $\mu_i \cup \nu$). Note also that L'$_j\sigma \notin T_i$ since the Simple VIP always makes $K_i \subset K_{i+1}$. Therefore, we apply A2) and

  - if L'$_j\sigma \in T_i$ we get (($G'_i \backslash$L'$_j$)σ ∅ T_i C_i K_{i+1}), where ($G'_i \backslash$L'$_j$)σμ_{i+1} = G_{i+1};
  - if L'$_j\sigma \notin T_i$ we get (($G'_i \backslash$L'$_j$)σ C_i $T_i \cup \{$L'$_j\sigma\}$ ∅ K_{i+1}), where ($G'_i \backslash$L'$_j$)σμ_{i+1} = G_{i+1}. Note that $(T_i \cup \{L'_j\sigma\})\mu_{i+1} \subseteq T$ since $(T_i \cup \{L'_j\sigma\})\mu_{i+1} = T_i\mu_i \cup \{L'_j\sigma\mu_1\nu\}$ and L'$_j\sigma\mu_1\nu$ = L'$_j\mu_1\sigma\nu$ = $L_j\sigma\nu$ = $L_j\theta_1 \in T$. The SLDNF tree for $R \cup T \cup C_i\mu_{i+1}$ fails finitely and has rank 0. Therefore, from lemma 4.14, there exists a CQC-derivation from ([] C_i $T_i \cup \{L_j\sigma\}$ ∅ K_{i+1}) to ([] ∅ $T_i \cup \{L_j\sigma\}$ C_{i+m} K_{i+1}). Moreover, lemma 4.14 also guarantees that for each $C_g \in C_{i+m}$-C_i the SLDNF tree for $R \cup T \cup \{C_g\mu_{i+1}\}$ fails finitely and has rank 0. Hence, the SLDNF tree for $R \cup T_{i+1} \cup C_{i+1}\mu_{i+1}$ fails finitely and has rank 0. Therefore, by lemma 4.10, there exists a CQC-derivation from (($G'_i \backslash$L'$_j$)σ C_i $T_i \cup \{L_j\sigma\}$ ∅ K_{i+1}) to (($G'_i \backslash$L'$_j$)σ ∅ $T_i \cup \{L_j\sigma\}$ C_{i+m} K_{i+1}).

L$_j$ and L'$_j$ are negative: We apply A3) and get ($G'_i \backslash$L'$_j$ {← ¬L'$_j$} T_i C_i K_i) with $\mu_{i+1} = \mu_i$. The SLDNF tree for $R \cup T \cup \{\leftarrow \neg L_j\}$ that fails finitely and has rank 0. Since L_j = L'$_j\mu_i$ and from lemma 4.14, there exists a CQC-derivation from ([] {← ¬L'$_j$} T_i C_i K_i) to ([] ∅ T_i C_{i+m} K_i). Moreover, lemma 4.14 also guarantees that for each $C_g \in C_{i+m}$-C_i the SLDNF tree for $R \cup T \cup \{C_g\mu_i\}$ fails finitely and has rank 0.

Hence, the SLDNF tree for $R \cup T_{i+1} \cup C_{i+1}\mu_{i+1}$ fails finitely and has rank 0. Therefore, by lemma 4.10, there exists a CQC-derivation from ($G'_i \backslash L'_j \{\leftarrow \neg L'_j\}\ T_i\ C_i\ K_i$) to ($G'_i \backslash L'_j \varnothing\ T_i\ C_{i+m}\ K_i$).

The derivation ends with []. □

Lemma 4.16. Let R be a set of deductive rules, K_0 the set of constant values appearing in R, T and T' EDBs and G a goal. If there exists a SLDNF refutation of rank 1 for $R \cup T \cup \{G\}$ then there exists a CQC-derivation from ($G \varnothing \varnothing \varnothing K_0$) to ([] $\varnothing$ T' C K) where $T'\mu \subseteq T$ and μ is a complete mapping from K to constants(T).

Proof: It is a direct consequence of Lemma 4.15. □

Theorem 4.17 (*Finite success completeness with the Simple VIP*)
Let Q_1 and Q_2 be two queries, DR be the set of deductive rules defining the database IDB relations and $R = DR \cup Q_1 \cup Q_2$. Let also $G_0 = \leftarrow q_1(X_1, ..., X_n) \wedge \neg q_2(X_1, ..., X_n)$ be the initial goal and K be the set of constants appearing in $DR \cup Q_1 \cup Q_2$. If $Q_1 \not\equiv Q_2$ then there exists a successful CQC-derivation from ($G_0 \varnothing \varnothing \varnothing K$) to ([] $\varnothing$ T C K') when the Simple VIP is applicable, provided that $\{G_0\} \cup F_0 \cup DR \cup Q_1 \cup Q_2$ is safe and either hierarchical or strict-stratified [CL89].

Proof: $Q_1 \not\equiv Q_2$ if there exists a model M for $pComp(R) \cup \{\exists X_1 \ldots X_n\ q_1(X_1,\ldots,X_n) \wedge \neg q_2(X_1,\ldots,X_n)\}$, where $R = DR \cup Q_1 \cup Q_2$. Let $T_M \subseteq M$ the set of all EDB facts belonging to M. Then M is a model also for $Comp(R \cup T_M) \cup \{\exists X_1 \ldots X_n\ q_1(X_1,\ldots,X_n) \wedge \neg q_2(X_1,\ldots,X_n)\}$. From the completeness of the SLDNF resolution for databases and goals that are safe and either hierarchical or strict-stratified [CL89], it follows that there exists an SLDNF refutation of $R \cup T_M \cup \{G_0\}$. In the particular case of deductive rules and goals that we handle in this Section, such a SLDNF refutation has rank 1.

From Lemma 4.16, if there exists an SLDNF refutation of rank 1 for $R \cup T_M \cup \{G_0\}$, then there is a CQC- derivation from ($G_0 \varnothing \varnothing \varnothing K_0$) to ([] $\varnothing$ T C K) with $T\mu \subseteq T_M$, where K_0 is the set of constants appearing in R and μ is a complete mapping from K to constants(T_M). □

References

[CL89] L. Cavedon, J.W. Lloyd. A Completeness Theorem for SLDNF Resolution, *Journal of Logic Programming*, 7(3):177-191, 1989.

[Llo87] J.W. Lloyd. *Foundations of Logic Programming*, Springer, 1987.