A Generic Scheme for Qualified Constraint Functional Logic Programming*
Technical Report SIC-1-09

Rafael Caballero, Mario Rodríguez-Artalejo and Carlos A. Romero-Díaz

Departamento de Sistemas Informáticos y Computación, Universidad Complutense,
Facultad de Informática, 28040 Madrid, Spain
{rafa,mario}@sip.ucm.es and cromdia@fdi.ucm.es

Abstract. Qualification has been recently introduced as a generalization of uncertainty in the field of Logic Programming. In this report we investigate a more expressive language for First-Order Functional Logic Programming with Constraints and Qualification. We present a Rewriting Logic which characterizes the intended semantics of programs, and a prototype implementation based on a semantically correct program transformation. Potential applications of the resulting language include flexible information retrieval. As a concrete illustration, we show how to write program rules to compute qualified answers for user queries concerning the books available in a given library.

Keywords: Constraints, Functional Logic Programming, Program Transformation, Qualification, Rewriting Logic.

1 Introduction

Various extensions of Logic Programming with uncertain reasoning capabilities have been widely investigated during the last 25 years. The recent recollection [21] reviews the evolution of the subject from the viewpoint of a committed researcher. All the proposals in the field replace classical two-valued logic by some kind of many-valued logic with more than two truth values, which are attached to computed answers and interpreted as truth degrees.

In a recent work [19,18] we have presented a *Qualified Logic Programming* scheme QLP($\mathcal{D}$) parameterized by a *qualification domain* $\mathcal{D}$, a lattice of so-called *qualification values* that are attached to computed answers and interpreted as a measure of the satisfaction of certain user expectations. QLP($\mathcal{D}$)-programs are sets of clauses of the form $A \xleftarrow{\alpha} \overline{B}$, where the head A is an atom, the body $\overline{B}$ is a conjunction of atoms, and $\alpha \in \mathcal{D}$ is called *attenuation* factor. Intuitively, α measures the maximum confidence placed on an inference performed by the clause. More precisely, any successful application of the clause attaches to the

* Research partially supported by projects MERIT–FORMS (TIN2005-09027-C03-03), PROMESAS–CAM(S-0505/TIC/0407) and STAMP (TIN2008-06622-C03-01).

head a qualification value which cannot exceed the infimum of $\alpha \circ \beta_i \in \mathcal{D}$, where β_i are the qualification values computed for the body atoms and $\circ$ is a so-called *attenuation operator*, provided by $\mathcal{D}$.

Uncertain Logic Programming can be expressed by particular instances of QLP($\mathcal{D}$), where the user expectation is understood as a lower bound for the *truth degree* of the computed answer and $\mathcal{D}$ is chosen to formalize a lattice of non-classical truth values. Other choices of $\mathcal{D}$ can be designed to model other kinds of user expectations, as e.g. an upper bound for the *size* of the logical proof underlying the computed answer. As shown in [4], the QLP($\mathcal{D}$) scheme is also well suited to deal with Uncertain Logic Programming based on similarity relations in the line of [20]. Therefore, Qualified Logic Programming has a potential for flexible information retrieval applications, where the answers computed for user queries may match the user expectations only to some degree. As shown in [19], several useful instances of QLP($\mathcal{D}$) can be conveniently implemented by using constraint solving techniques.

In this report we investigate an extension of QLP($\mathcal{D}$) to a more expressive scheme, supporting computation with first-order lazy functions and constraints. More precisely, we consider the first-order fragment of CFLP($\mathcal{C}$), a generic scheme for functional logic programming with constraints over a parametrically given domain $\mathcal{C}$ presented in [13]. We propose an extended scheme QCFLP($\mathcal{D}$,$\mathcal{C}$) where the additional parameter $\mathcal{D}$ stands for a qualification domain. QCFLP($\mathcal{D}$,$\mathcal{C}$)-programs are sets of conditional rewrite rules of the form $f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \Delta$, where the condition Δ is a conjunction of $\mathcal{C}$-constraints that may involve user defined functions, and $\alpha \in \mathcal{D}$ is an attenuation factor. As in the logic programming case, α measures the maximum confidence placed on an inference performed by the rule: any successful application of the rule attaches to the computed result a qualification value which cannot exceed the infimum of $\alpha \circ \beta_i \in \mathcal{D}$, where β_i are the qualification values computed for r and Δ, and $\circ$ is $\mathcal{D}$'s attenuation operator. QLP($\mathcal{D}$) program clauses can be easily formulated as a particular case of QCFLP($\mathcal{D}$,$\mathcal{C}$) program rules.

As far as we know, no related work covers the expressivity of our approach. Guadarrama et al. [8] have proposed to use real arithmetic constraints as an implementation tool for a Fuzzy Prolog, but their language does not support constraint programming as such. Starting from the field of natural language processing, Riezler [15,16] has developed quantitative and probabilistic extensions of the classical CLP($\mathcal{C}$) scheme with the aim of computing good parse trees for constraint logic grammars, but his work bears no relation to functional programming. Moreno and Pascual [14] have investigated similarity-based unification in the context of *needed narrowing* [1], a narrowing strategy using so-called *definitional trees* that underlies the operational semantics of functional logic languages such as Curry [9] and $\mathcal{TOY}$ [3], but they use neither constraints nor attenuation factors and they provide no declarative semantics. The approach of the present report is quite different. We work with a class of programs more general and expressive than the *inductively sequential* term rewrite systems used in [14], and our results focus on a rewriting logic used to characterize declarative semantics and to prove the correctness of an implementation technique based on a program transformation. Similarity relations could be easily incorporated to our scheme by using the techniques presented in [4] for the Logic Programming case. Moreover, the good properties of needed narrowing as a computation model are not spoiled by our implementation technique, because our program transformation preserves the structure of the definitional trees derived from the user-given program rules.

```
%% Data types:
type pages, id = int
type title, author, language, genre = [char]
data vocabularyLevel = easy | medium | difficult
data readerLevel = basic | intermediate | upper | proficiency
data book = book(id, title, author, language, genre, vocabularyLevel, pages)

%% Simple library, represented as list of books:
library :: [book]
library --> [ book(1, "Tintin", "Herge", "French", "Comic", easy, 65),
              book(2, "Dune", "F. P. Herbert", "English", "SciFi", medium, 345),
              book(3, "Kritik der reinen Vernunft", "Immanuel Kant", "German",
                   "Philosophy", difficult, 1011),
              book(4, "Beim Hauten der Zwiebel", "Gunter Grass", "German",
                   "Biography", medium, 432) ]

%% Auxiliary function for computing list membership:
member(B,[]) --> false
member(B,H:_T) --> true <== B == H
member(B,H:T) --> member(B,T) <== B /= H

%% Functions for getting the explicit attributes of a given book:
getId(book(Id,_Title,_Author,_Lang,_Genre,_VocLvl,_Pages)) --> Id
getTitle(book(_Id,Title,_Author,_Lang,_Genre,_VocLvl,_Pages)) --> Title
getAuthor(book(_Id,_Title,Author,_Lang,_Genre,_VocLvl,_Pages)) --> Author
getLanguage(book(_Id,_Title,_Author,Lang,_Genre,_VocLvl,_Pages)) --> Lang
getGenre(book(_Id,_Title,_Author,_Lang,Genre,_VocLvl,_Pages)) --> Genre
getVocabularyLevel(book(_Id,_Title,_Author,_Lang,_Genre,VocLvl,_Pages)) --> VocLvl
getPages(book(_Id,_Title,_Author,_Lang,_Genre,_VocLvl,Pages)) --> Pages

%% Function for guessing the genre of a given book:
guessGenre(B) --> getGenre(B)
guessGenre(B) -0.9-> "Fantasy" <== guessGenre(B) == "SciFi"
guessGenre(B) -0.8-> "Essay" <== guessGenre(B) == "Philosophy"
guessGenre(B) -0.7-> "Essay" <== guessGenre(B) == "Biography"
guessGenre(B) -0.7-> "Adventure" <== guessGenre(B) == "Fantasy"

%% Function for guessing the reader level of a given book:
guessReaderLevel(B) --> basic <== getVocabularyLevel(B) == easy, getPages(B) < 50
guessReaderLevel(B) -0.8-> intermediate <== getVocabularyLevel(B) == easy, getPages(B) >= 50
guessReaderLevel(B) -0.9-> basic <== guessGenre(B) == "Children"
guessReaderLevel(B) -0.9-> proficiency <== getVocabularyLevel(B) == difficult,
                                           getPages(B) >= 200
guessReaderLevel(B) -0.8-> upper <== getVocabularyLevel(B) == difficult, getPages(B) < 200
guessReaderLevel(B) -0.8-> intermediate <== getVocabularyLevel(B) == medium
guessReaderLevel(B) -0.7-> upper <== getVocabularyLevel(B) == medium

%% Function for answering a particular kind of user queries:
search(Language,Genre,Level) --> getId(B) <== member(B,library),
                                              getLanguage(B) == Language,
                                              guessReaderLevel(B) == Level,
                                              guessGenre(B) == Genre
```

Fig. 1. Library with books in different languages

Figure 1 shows a small set of QCFLP($\mathcal{U}, \mathcal{R}$) program rules, called the *library program* in the rest of the report. The concrete syntax is inspired by the functional logic language $\mathcal{TOY}$, but the ideas and results of this report could be also applied to Curry and other similar languages. In this example, $\mathcal{U}$ stands for a particular qualification domain which supports uncertain truth values in the real interval $[0, 1]$, while $\mathcal{R}$ stands for a particular constraint domain which supports arithmetic constraints over the real numbers; see Section 2 for more details.

The program rules are intended to encode expert knowledge for computing qualified answers to user queries concerning the books available in a simplified library, represented as a list of objects of type `book`. The various `get` functions extract the explicit values of book attributes. Functions `guessGenre` and `guessReaderLevel` infer information by performing qualified inferences, relying on analogies between different genres and heuristic rules to estimate reader levels on the basis of other features of a given book, respectively. Some program rules, as e.g. those of the auxiliary function `member`, have attached no explicit attenuation factor. By convention, this is understood as the implicit attachment of the attenuation factor `1.0`, the top value of $\mathcal{U}$. For any instance of the QCFLP($\mathcal{D}, \mathcal{C}$) scheme, a similar convention allows to view CFLP($\mathcal{C}$)-program rules as QCFLP($\mathcal{D}, \mathcal{C}$)-program rules whose attached qualification is optimal.

The last rule for function `search` encodes a method for computing qualified answers to a particular kind of user queries. Therefore, the queries can be formulated as goals to be solved by the program fragment. For instance, answering the query of a user who wants to find a book of genre `"Essay"`, language `"German"` and user level `intermediate` with a certainty degree of at least 0.65 can be formulated as the goal:

```
(search("German","Essay",intermediate) == R) # W | W >= 0.65
```

The techniques presented in Section 4 can be used to translate the QCFLP($\mathcal{U}, \mathcal{R}$) program rules and goal into the CFLP($\mathcal{R}$) language, which is implemented in the $\mathcal{TOY}$ system. Solving the translated goal in $\mathcal{TOY}$ computes the answer $\{R \mapsto 4\}\{0.65 \leq W, W \leq 0.7\}$, ensuring that the library book with `id` 4 satisfies the query's requirements with any certainty degree in the interval [0.65,0.7], in particular 0.7. The computation uses the 4th program rule of `guessGenre` to obtain `"Essay"` as the book's genre with qualification 0.7, and the 6th program rule of `guessReaderLevel` to obtain `intermediate` as the reader level with qualification 0.8.

The rest of the report is organized as follows. In Section 2 we recall known proposals concerning qualification and constraint domains, and we introduce a technical notion needed to relate both kinds of domains for the purposes of this report. In Section 3 we present the generic scheme QCFLP($\mathcal{D}, \mathcal{C}$) announced in this introduction, and we formalize a special Rewriting Logic which characterizes the declarative semantics of QCFLP($\mathcal{D}, \mathcal{C}$)-programs. In Section 4 we present a semantically correct program transformation converting QCFLP($\mathcal{D}, \mathcal{C}$) programs and goals into the qualification-free CFLP($\mathcal{C}$) programming scheme, which is supported by existing systems such as $\mathcal{TOY}$. Section 5 concludes and points to some lines of planned future work.

2 Qualification and Constraint Domains

Qualification Domains were introduced in [19]. Their intended use has been already explained in the Introduction. In this section we recall and slightly improve their axiomatic definition.

Definition 1 (Qualification Domains). *A* Qualification Domain *is any structure $\mathcal{D} = \langle D, \trianglelefteq, \mathbf{b}, \mathbf{t}, \circ \rangle$ verifying the following requirements:*

1. *D, noted as $D_\mathcal{D}$ when convenient, is a set of elements called* qualification values*.*
2. *$\langle D, \trianglelefteq, \mathbf{b}, \mathbf{t} \rangle$ is a lattice with extreme points* $\mathbf{b}$ *and* $\mathbf{t}$ *w.r.t. the partial ordering $\trianglelefteq$. For given elements $d, e \in D$, we write $d \sqcap e$ for the* greatest lower bound *(glb) of d and e, and $d \sqcup e$ for the* least upper bound *(lub) of d and e. We also write $d \triangleleft e$ as abbreviation for $d \trianglelefteq e \wedge d \neq e$.*
3. *$\circ : D \times D \longrightarrow D$, called* attenuation operation*, verifies the following axioms:*
   (a) $\circ$ is associative, commutative and monotonic w.r.t. $\trianglelefteq$.
   (b) $\forall d \in D : d \circ \mathbf{t} = d$.
   (c) $\forall d, e \in D \setminus \{\mathbf{b}, \mathbf{t}\} : d \circ e \triangleleft e$.
   (d) $\forall d, e_1, e_2 \in D : d \circ (e_1 \sqcap e_2) = d \circ e_1 \sqcap d \circ e_2$. □

As an easy consequence of the previous definition one can prove the following proposition. [1]

Proposition 1 (Additional properties of qualification domains). *Any qualification domain $\mathcal{D}$ satisfies the following properties:*

1. *$\forall d, e \in D : d \circ e \trianglelefteq e$.*
2. *$\forall d \in D : d \circ \mathbf{b} = \mathbf{b}$.*

Proof. Since $\mathbf{t}$ is the top element of the lattice, we know $d \trianglelefteq \mathbf{t}$ for any $d \in D$. As $\circ$ is monotonic w.r.t. $\trianglelefteq$, $d \circ e \trianglelefteq \mathbf{t} \circ e$ also holds for any $e \in D$ which, due to commutativity and axiom (b) of $\circ$, yields $d \circ e \trianglelefteq e$. Therefore *1*. holds. Now, taking $e = \mathbf{b}$, one has $d \circ \mathbf{b} \trianglelefteq \mathbf{b}$ which implies $d \circ \mathbf{b} = \mathbf{b}$ as $\mathbf{b}$ is the bottom element of the lattice. Hence *2*. also holds. □

The examples in this report will use a particular qualification domain $\mathcal{U}$ whose values represent certainty degrees in the sense of fuzzy logic. Formally, $\mathcal{U} = \langle U, \leq, 0, 1, \times \rangle$, where $U = [0, 1] = \{d \in \mathbb{R} \mid 0 \leq d \leq 1\}$, $\leq$ is the usual numerical ordering, and $\times$ is the multiplication operation. In this domain, the bottom and top elements are $\mathbf{b} = 0$ and $\mathbf{t} = 1$, and the infimum of a finite $S \subseteq U$ is the minimum value $\min(S)$, understood as 1 if $S = \emptyset$. The class of qualification domains is closed under cartesian products. For a proof of this fact and other examples of qualification domains, the reader is referred to [19,18].

Constraint domains are used in Constraint Logic Programming and its extensions as a tool to provide data values, primitive operations and constraints

[1] The authors are thankful to G. Gerla for pointing out this fact.

tailored to domain-oriented applications. Various formalizations of this notion are known. In this report, constraint domains are related to signatures of the form $\Sigma = \langle DC, PF, DF \rangle$ where $DC = \bigcup_{n\in\mathbb{N}} DC^n$, $PF = \bigcup_{n\in\mathbb{N}} PF^n$ and $DF = \bigcup_{n\in\mathbb{N}} DF^n$ are mutually disjoint sets of *data constructor* symbols, *primitive function* symbols and *defined function* symbols, respectively, ranked by arities. Given a signature Σ, a symbol $\perp$ to note the *undefined value*, a set B of *basic values* u and a countably infinite set $\mathcal{V}ar$ of variables X, we define the notions listed below, where $\overline{o}_n$ abbreviates the n-tuple of syntactic objects $o_1, \ldots, o_n$.

- *Expressions* $e \in \mathrm{Exp}_\perp(\Sigma, B, \mathcal{V}ar)$ have the syntax $e ::= \perp|X|u|h(\overline{e}_n)$, where $h \in DC^n \cup PF^n \cup DF^n$. In the case $n = 0$, $h(\overline{e}_n)$ is written simply as h.
- *Constructor Terms* $t \in \mathrm{Term}_\perp(\Sigma, B, \mathcal{V}ar)$ have the syntax $e ::= \perp|X|u|c(\overline{t}_n)$, where $c \in DC^n$. They will be called just terms in the sequel.
- *Total Expressions* $e \in \mathrm{Exp}(\Sigma, B, \mathcal{V}ar)$ and *Total Terms* $t \in \mathrm{Term}(\Sigma, B, \mathcal{V}ar)$ have a similar syntax, with the $\perp$ case omitted.
- An expression or term (total or not) is called *ground* iff it includes no occurrences of variables. $\mathrm{Exp}_\perp(\Sigma, B)$ stands for the set of all ground expressions. The notations $\mathrm{Term}_\perp(\Sigma, B)$, $\mathrm{Exp}(\Sigma, B)$ and $\mathrm{Term}(\Sigma, B)$ have a similar meaning.
- We note as $\sqsubseteq$ the *information ordering*, defined as the least partial ordering over $\mathrm{Exp}_\perp(\Sigma, B, \mathcal{V}ar)$ compatible with contexts and verifying $\perp \sqsubseteq e$ for all $e \in \mathrm{Exp}_\perp(\Sigma, B, \mathcal{V}ar)$.
- Substitutions are defined as mappings $\sigma : \mathcal{V}ar \to \mathrm{Term}_\perp(\Sigma, B, \mathcal{V}ar)$ assigning not necessarily total terms to variables. They can be represented as sets of bindings $X \mapsto t$ and extended to act over other syntactic objects o. The *domain* $\mathrm{vdom}(\sigma)$ and *variable range* $\mathrm{vran}(\sigma)$ are defined in the usual way. We will write $o\sigma$ for the result of applying σ to o. The *composition* $\sigma\sigma'$ of two substitutions is such that $o(\sigma\sigma')$ equals $(o\sigma)\sigma'$.

By adapting the definition found in Section 2.2 of [13] to a first-order setting, we obtain: [2]

Definition 2 (Constraint Domains). *A* Constraint Domain *of signature* Σ *is any algebraic structure of the form* $\mathcal{C} = \langle C, \{p^\mathcal{C} \mid p \in PF\}\rangle$ *such that:*

1. *The carrier set* C *is* $\mathrm{Term}_\perp(\Sigma, B)$ *for a certain set* B *of* basic values*. When convenient, we note* B *and* C *as* $B_\mathcal{C}$ *and* $C_\mathcal{C}$*, respectively.*
2. $p^\mathcal{C} \subseteq C^n \times C$*, written simply as* $p^\mathcal{C} \subseteq C$ *in the case* $n = 0$*, is called the* interpretation *of* p *in* $\mathcal{C}$*. We will write* $p^\mathcal{C}(\overline{t}_n) \to t$ *(or simply* $p^\mathcal{C} \to t$ *if* $n = 0$*) to indicate that* $(\overline{t}_n, t) \in p^\mathcal{C}$*.*
3. *Each primitive interpretation* $p^\mathcal{C}$ *has* monotonic *and* radical *behavior w.r.t. the information ordering* $\sqsubseteq$*. More precisely:*
   (a) **Monotonicity***: For all* $p \in PF^n$*,* $p^\mathcal{C}(\overline{t}_n) \to t$ *behaves monotonically w.r.t. the arguments* $\overline{t}_n$ *and antimonotonically w.r.t. the result* t*. Formally: For all* $\overline{t}_n, \overline{t'}_n, t, t' \in C$ *such that* $p^\mathcal{C}(\overline{t}_n) \to t$*,* $\overline{t}_n \sqsubseteq \overline{t'}_n$ *and* $t \sqsupseteq t'$*,* $p^\mathcal{C}(\overline{t'}_n) \to t'$ *also holds.*

[2] We slightly modify the statement of the *radicality* property, rendering it simpler than in [13] but sufficient for practical purposes.

(b) **Radicality**: *For all $p \in PF^n$, as soon as the arguments given to $p^{\mathcal{C}}$ have enough information to return a result other than $\bot$, the same arguments suffice already for returning a simple total result. Formally: For all $\overline{t}_n, t \in C$, if $p^{\mathcal{C}}(\overline{t}_n) \to t$ then $t = \bot$ or else $t \in B \cup DC^0$.*

Note that symbols $h \in DC \cup DF$ are given no interpretation in $\mathcal{C}$. As we will see in Section 3, symbols in $c \in DC$ are interpreted as free constructors, and the interpretation of symbols $f \in DF$ is program-dependent. We assume that any signature Σ includes two nullary constructors *true* and *false* for the boolean values, and a binary symbol $== \in PF^2$ used in infix notation and interpreted as *strict equality*; see [13] for details. For the examples in this report we will use a constraint domain $\mathcal{R}$ whose set of basic elements is $C_{\mathcal{R}} = \mathbb{R}$ and whose primitives functions correspond to the usual arithmetic operations $+, \times, \ldots$ and the usual boolean-valued comparison operations $\leq, <, \ldots$ over $\mathbb{R}$. Other useful instances of constraint domains can be found in [13].

Atomic constraints over $\mathcal{C}$ have the form $p(\overline{e}_n) == v$ [3] with $p \in PF^n$, $e_i \in \mathrm{Exp}_\bot(\Sigma, B, \mathcal{V}ar)$ and $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$. Atomic constraints of the form $p(\overline{e}_n) == true$ are abbreviated as $p(\overline{e}_n)$. In particular, $(e_1 == e_2) == true$ is abbreviated as $e_1 == e_2$. Atomic constraints of the form $(e_1 == e_2) == false$ are abbreviated as $e_1 \mathrel{/=} e_2$.

Compound constraints are built from atomic constraints using logical conjunction, existential quantification, and sometimes other logical operations. Constraints without occurrences of symbols $f \in DF$ are called *primitive*. We will note atomic constraints as δ, sets of atomic constraints as Δ, atomic primitive constraints as π, and sets of atomic primitive constraints as Π. When interpreting set of constraints, we will treat them as the conjunction of their members.

Ground substitutions η such that $X\eta \in \mathrm{Term}_\bot(\Sigma, B)$ for all $X \in \mathrm{vdom}(\eta)$ are called *variable valuations* over $\mathcal{C}$. The set of all possible variable valuations is noted $\mathrm{Val}_{\mathcal{C}}$. The *solution set* $\mathrm{Sol}_{\mathcal{C}}(\Pi) \subseteq \mathrm{Val}_{\mathcal{C}}$ includes as members those valuations η such that $\pi\sigma$ is true in $\mathcal{C}$ for all $\pi \in \Pi$; see [13] for a formal definition. In case that $\mathrm{Sol}_{\mathcal{C}}(\Pi) = \emptyset$ we say that Π is *unsatisfiable* and we write $\mathrm{Unsat}_{\mathcal{C}}(\Pi)$. In case that $\mathrm{Sol}_{\mathcal{C}}(\Pi) \subseteq \mathrm{Sol}_{\mathcal{C}}(\pi)$ we say that π is *entailed* by Π in $\mathcal{C}$ and we write $\Pi \models_{\mathcal{C}} \pi$. Note that the notions defined in this paragraph only make sense for primitive constraints.

In this report we are interested in pairs consisting of a qualification domain and a constraint domain that are related in the following technical sense:

Definition 3 (Expressing $\mathcal{D}$ in $\mathcal{C}$). *A qualification domain $\mathcal{D}$ with carrier set $D_{\mathcal{D}}$ is expressible in a constraint domain $\mathcal{C}$ with carrier set $C_{\mathcal{C}}$ if $D_{\mathcal{D}} \setminus \{\mathbf{b}\} \subseteq C_{\mathcal{C}}$ and the two following requirements are satisfied:*

1. *There is a primitive $\mathcal{C}$-constraint $\mathsf{qVal}(X)$ depending on the variable X, such that $Sol_{\mathcal{C}}(\mathsf{qVal}(X)) = \{\eta \in Val_{\mathcal{C}} \mid \eta(X) \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}\}$.*
2. *There is a primitive $\mathcal{C}$-constraint $\mathsf{qBound}(X, Y, Z)$ depending on the variables X, Y, Z, such that any $\eta \in Val_{\mathcal{C}}$ such that $\eta(X)$, $\eta(Y)$, $\eta(Z) \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $\eta \in Sol_{\mathcal{C}}(\mathsf{qBound}(X, Y, Z)) \iff \eta(X) \trianglelefteq \eta(Y) \circ \eta(Z)$.* □

[3] Written as $p(\overline{e}_n) \to! v$ in [13].

Intuitively, $\mathsf{qBound}(X, Y, Z)$ encodes the $\mathcal{D}$-statement $X \trianglelefteq Y \circ Z$ as a $\mathcal{C}$-constraint. As convenient notations, we will write $\ulcorner X \trianglelefteq Y \circ Z \urcorner$, $\ulcorner X \trianglelefteq Y \urcorner$ and $\ulcorner X \trianglerighteq Y \urcorner$ in place of $\mathsf{qBound}(X, Y, Z)$, $\mathsf{qBound}(X, \mathbf{t}, Y)$ and $\mathsf{qBound}(Y, \mathbf{t}, X)$, respectively. In the sequel, $\mathcal{C}$-constraints of the form $\ulcorner \kappa \urcorner$ are called *qualification constraints*, and Ω is used as notation for sets of qualification constraints. We also write $\mathrm{Val}_\mathcal{D}$ for the set of all $\mu \in \mathrm{Val}_\mathcal{C}$ such that $X\mu \in D_\mathcal{D} \setminus \{\mathbf{b}\}$ for all $X \in \mathrm{vdom}(\mu)$, called *$\mathcal{D}$-valuations*.

Note that $\mathcal{U}$ can be expressed in $\mathcal{R}$, because $D_\mathcal{U} \setminus \{0\} = (0, 1] \subseteq \mathbb{R} \subseteq C_\mathcal{R}$, $\mathsf{qVal}(X)$ can be built as the $\mathcal{R}$-constraint $0 < X \wedge X \leq 1$ and $\ulcorner X \trianglelefteq Y \circ Z \urcorner$ can be built as the $\mathcal{R}$-constraint $X \leq Y \times Z$. Other instances of qualification domains presented in [19] are also expressible in $\mathcal{R}$.

3 A Qualified Declarative Programming Scheme

In this section we present the scheme QCFLP($\mathcal{D}, \mathcal{C}$) announced in the Introduction, and we develop alternative characterizations of its declarative semantics using an interpretation transformer and a rewriting logic. The parameters $\mathcal{D}$ and $\mathcal{C}$ respectively stand for a qualification domain and a constraint domain with certain signature Σ. By convention, we only allow those instances of the scheme verifying that $\mathcal{D}$ is expressible in $\mathcal{C}$ in the sense of Definition 3. For example, QCFLP($\mathcal{U}, \mathcal{R}$) is an allowed instance.

Technically, the results presented here extend similar ones known for the CFLP($\mathcal{C}$) sheme [13], omitting higher-order functions and adding a suitable treatment of qualifications. In particular, the qc-interpretations for QCFLP($\mathcal{D}, \mathcal{C}$)-programs are a natural extension of the c-interpretations for CFLP($\mathcal{C}$)-programs introduced in [13]. In turn, these were inspired by the π-interpretations for the CLP($\mathcal{C}$) scheme proposed by Dore, Gabbrielli and Levi [7,6].

3.1 Programs, Interpretations and Models

A QCFLP($\mathcal{D}, \mathcal{C}$)-program is a set $\mathcal{P}$ of program rules. A program rule has the form $f(\bar{t}_n) \xrightarrow{\alpha} r \Leftarrow \Delta$ where $f \in DF^n$, $\bar{t}_n$ is a lineal sequence of Σ-terms, $\alpha \in D_\mathcal{D} \setminus \{\mathbf{b}\}$ is an attenuation factor, r is a Σ-expression and Δ is a sequence of atomic $\mathcal{C}$-constraints δ_j $(1 \leq j \leq m)$, interpreted as conjunction. The undefined symbol $\perp$ is not allowed to occur in program rules.

The library program shown in Figure 1 is an example of QCFLP($\mathcal{U}, \mathcal{R}$)-program. Leaving aside the attenuation factors, this is clearly not a confluent conditional term rewriting system. Certain program rules, as e.g. those for `guessGenre`, are intended to specify the behavior of *non-deterministic functions*. As argued elsewhere [17], the semantics of non-deterministic functions for the purposes of Functional Logic Programming is not suitably described by ordinary rewriting. Inspired by the approach in [13], we will overcome this difficulty by designing special inference mechanisms to derive semantically meaningful statements from programs. The kind of statements that we will consider are defined below:

Definition 4 (qc-Statements). *Assume partial Σ-expression e, partial Σ-terms $t, t', \bar{t}_n$, a qualification value $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$, an atomic $\mathcal{C}$-constraint δ and a finite set of atomic primitive $\mathcal{C}$-constraints Π. A* qualified constrained statement *(briefly, qc-statement) φ must have one of the following two forms:*

1. qc-production $(e \to t)\sharp d \Leftarrow \Pi$. *Such a qc-statement is called* trivial *iff either t is $\bot$ or else $Unsat_{\mathcal{C}}(\Pi)$. Its intuitive meaning is that a rewrite sequence $e \to^* t'$ using program rules and with attached qualification value d is allowed in our intended semantics for some $t' \sqsupseteq t$, under the assumption that Π holds. By convention, qc-productions of the form $(f(\bar{t}_n) \to t)\sharp d \Leftarrow \Pi$ with $f \in DF^n$ are called* qc-facts.
2. qc-atom $\delta\sharp d \Leftarrow \Pi$. *Such a qc-statement is called* trivial *iff $Unsat_{\mathcal{C}}(\Pi)$. Its intuitive meaning is that δ is entailed by the program rules with attached qualification value d, under the assumption that Π holds.* □

Our semantics will use program interpretations defined as sets of qc-facts with certain closure properties. As an auxiliary tool we need the following technical notion:

Definition 5 ($(\mathcal{D}, \mathcal{C})$-Entailment). *Given two qc-statements φ and φ', we say that φ $(\mathcal{D}, \mathcal{C})$-entails φ' (in symbols, $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$) iff one of the following two cases hold:*

1. *$\varphi = (e \to t)\sharp d \Leftarrow \Pi$, $\varphi' = (e' \to t')\sharp d' \Leftarrow \Pi'$, and there is some substitution σ such that $\Pi' \models_{\mathcal{C}} \Pi\sigma$, $d \trianglerighteq d'$, $e\sigma \sqsubseteq e'$ and $t\sigma \sqsupseteq t'$.*
2. *$\varphi = \delta\sharp d \Leftarrow \Pi$, $\varphi' = \delta'\sharp d' \Leftarrow \Pi'$, and there is some substitution σ such that $\Pi' \models_{\mathcal{C}} \Pi\sigma$, $d \trianglerighteq d'$, $\delta\sigma \sqsubseteq \delta'$.* □

The intended meaning of $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$ is that φ' follows from φ, regardless of the interpretation of the defined function symbols $f \in DF$ occurring in φ, φ'. Intuitively, this is the case because the interpretations of defined function symbols are expected to satisfy the monotonicity properties stated for the case of primitive function symbols in Definition 2. The following example may help to understand the idea:

Example 1 ($(\mathcal{U}, \mathcal{R})$-entailment). Let φ, φ' be defined as:

$$\begin{aligned} \varphi\ &: (f(X\!:\!Xs) \to Xs)\sharp 0.8 \Leftarrow X \times X \neq 0 \\ \varphi'\ &: (f(A\!:\!(B\!:\![\,])) \to \bot\!:\!\bot)\sharp 0.7 \Leftarrow A < 0 \end{aligned}$$

Then $\varphi \succcurlyeq_{\mathcal{U},\mathcal{R}} \varphi'$ with $\sigma = \{X \mapsto A,\ Xs \mapsto B\!:\!\bot\}$ because:

- $\Pi' \models_{\mathcal{R}} \Pi\sigma$, since $\Pi' = \{A < 0\}$, $\Pi\sigma = \{X \times X \neq 0\}\sigma = \{A \times A \neq 0\}$, and $A \times A \neq 0$ is entailed by $A < 0$ in $\mathcal{R}$.
- $d \trianglerighteq d'$ holds in $\mathcal{U}$, since $d = 0.8 \geq 0.7 = d'$.
- $e\sigma \sqsubseteq e'$, since $e\sigma = f(X\!:\!Xs)\sigma = f(A\!:\!(B\!:\!\bot)) \sqsubseteq f(A\!:\!(B\!:\![\,])) = e'$.
- $t\sigma \sqsupseteq t'$, since $t\sigma = Xs\sigma = B\!:\!\bot \sqsupseteq \bot\!:\!\bot = t'$. □

Now we can define program interpretations as follows:

Definition 6 (qc-Interpretations). *A* qualified constrained interpretation *(or* qc-interpretation*) over $\mathcal{D}$ and $\mathcal{C}$ is a set $\mathcal{I}$ of qc-facts including all trivial and entailed qc-facts. In other words, a set $\mathcal{I}$ of qc-facts such that $cl_{\mathcal{D},\mathcal{C}}(\mathcal{I}) \subseteq \mathcal{I}$, where the closure over $\mathcal{D}$ and $\mathcal{C}$ of $\mathcal{I}$ is defined as:*

$$cl_{\mathcal{D},\mathcal{C}}(\mathcal{I}) =_{\text{def}} \{\varphi \mid \varphi \textit{ trivial}\} \cup \{\varphi' \mid \varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi' \textit{ for some } \varphi \in \mathcal{I}\} \ .$$

We write $Int_{\mathcal{D},\mathcal{C}}$ for the set of all qc-interpretations over $\mathcal{D}$ and $\mathcal{C}$.

QTI $\quad \dfrac{}{\varphi}$ if φ is a trivial qc-statement.

QRR $\quad \dfrac{}{(v \to v)\sharp d \Leftarrow \Pi}$ if $v \in \mathcal{V}ar \cup B_{\mathcal{C}}$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$.

QDC $\quad \dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(c(\overline{e}_n) \to c(\overline{t}_n))\sharp d \Leftarrow \Pi}$ if $c \in DC^n$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(1 \le i \le n)$.

QDF$_{\mathcal{I}}$ $\quad \dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi}$

if $f \in DF^n$, non-trivial $((f(\overline{t}_n) \to t)\sharp d_0 \Leftarrow \Pi) \in \mathcal{I}$
and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(0 \le i \le n)$.

QPF $\quad \dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{e}_n) \to v)\sharp d \Leftarrow \Pi}$ if $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$,

$\Pi \models_{\mathcal{C}} p(\overline{t}_n) \to v$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(1 \le i \le n)$.

QAC $\quad \dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{e}_n) == v)\sharp d \Leftarrow \Pi}$ if $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$,

$\Pi \models_{\mathcal{C}} p(\overline{t}_n) == v$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(1 \le i \le n)$.

Fig. 2. Qualified Constrained Rewriting Logic for Interpretations

Given a qc-interpretation $\mathcal{I}$, the inference rules displayed in Fig. 2 are used to derive qc-statements from the qc-facts belonging to $\mathcal{I}$. The inference system consisting of these rules is called *Qualified Constrained Rewriting Logic for Interpretations* and noted as $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$. The notation $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ is used to indicate that φ can be derived from $\mathcal{I}$ in $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$. By convention, we agree that no other inference rule is used whenever **QTI** is applicable. Therefore, trivial qc-statements can only be inferred by rule **QTI**. As usual in formal inference systems, $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$ proofs can be represented as trees whose nodes correspond to inference steps.

In the sequel, the inference rules $\mathbf{QDF}_{\mathcal{I}}$, **QPF** and **QAC** will be called *crucial*. The notation $|\mathcal{T}|$ will denote the number of inference steps within the proof tree $\mathcal{T}$ that are *not crucial*. Proof trees with no crucial inferences (i.e. such that $|\mathcal{T}| = 0$) will be called *easy*. The following lemma states some technical properties of $\mathcal{I}$-QCRWL$(\mathcal{D}, \mathcal{C})$.

Lemma 1 (Some properties of $\mathcal{I}$-QCRWL$(\mathcal{D}, \mathcal{C})$).

1. *Approximation property: For any non-trivial φ of the form $(t \to t')\sharp d \Leftarrow \Pi$ where $t, t' \in Term_{\perp}(\Sigma, B, \mathcal{V}ar)$, the three following affirmations are equivalent: (a) $t \sqsupseteq t'$; (b) $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with an easy proof tree; and (c) $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$.*
2. *Primitive c-atoms: For any primitive c-atom $p(\overline{t}_n) == v$, one has $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}}$ $(p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi \iff \Pi \models_{\mathcal{C}} p(\overline{t}_n) == v$.*
3. *Entailment property: $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with a proof tree $\mathcal{T}$ and $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi' \Longrightarrow$ $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi'$ with a proof tree $\mathcal{T}'$ such that $|\mathcal{T}'| \le |\mathcal{T}|$.*
4. *Conservation property: For any qc-fact φ, one has $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi \iff \varphi \in \mathcal{I}$.*

Proof. We argue separately for each of the four properties:

[*1.*] (*Approximation property*). The terms t and t' involve neither defined nor primitive function symbols. Due to the form of the $\mathcal{I}$-QCRWL$(\mathcal{D}, \mathcal{C})$ inference rules, a proof of the qc-statement $(t \to t')\sharp d \Leftarrow \Pi$ will involve no crucial inferences and it will succeed iff $t \sqsupseteq t'$. A formal proof can be easily obtained reasoning by induction on the syntactic size of t, similarly as in item 3. of Lemma 1 from [13].

[*2.*] (*Primitive c-atoms*). Let φ be $(p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi$. If φ is trivial, then $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi$ can be proved with just one **QTI** inference, and $\Pi \models_{\mathcal{C}} p(\overline{t}_n) == v$ also holds because of $\mathrm{Unsat}_{\mathcal{C}}(\Pi)$. If φ is not trivial, then:

- ($\Leftarrow$) Assume $\Pi \models_{\mathcal{C}} p(\overline{t}_n) == v$. Then $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi$ can be obtained with a proof of the form

$$\frac{(\ (t_i \to t_i)\sharp\mathbf{t} \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi}\ \mathrm{QAC}$$

  where each of the n premises has an easy $\mathcal{I}$-QCRWL$(\mathcal{D}, \mathcal{C})$-proof due to the approximation property (since $t_i \sqsupseteq t_i$).
- ($\Rightarrow$) Assume now $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi$. The $\mathcal{I}$-QCRWL$(\mathcal{D}, \mathcal{C})$-proof will have the form

$$\frac{(\ (t_i \to t'_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi}\ \mathrm{QAC}$$

  where $\Pi \models_{\mathcal{C}} p(\overline{t'}_n) == v$ and $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (t_i \to t_i)\sharp d_i \Leftarrow \Pi$, $d \trianglelefteq d_i$ hold for all $1 \le i \le n$. Due to the approximation property, we can conclude that $t_i \sqsupseteq t'_i$ holds for $1 \le i \le n$, which implies $\Pi \models_{\mathcal{C}} p(\overline{t}_n) == v$ because of the monotonic behavior of primitive functions in constraint domains.

[*3.*] (*Entailment property*). Assume $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with a $\mathcal{I}$-QCRWL($\mathcal{D},\mathcal{C}$)-proof tree $\mathcal{T}$. We must prove that $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi'$ with some proof tree $\mathcal{T}'$ such that $|\mathcal{T}'| \leq |\mathcal{T}|$. If φ' results trivial, then it is proved with just one **QTI** inference step, and therefore $|\mathcal{T}'| = 0 \leq |\mathcal{T}|$. In the sequel, we assume φ' non-trivial and we reason by induction on the number of inference steps within $\mathcal{T}$. We distinguish cases according to the inference step at the root of $\mathcal{T}$:

- **QTI**: From Definition 5 it is easy to check that φ' must be trivial whenever $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$ and φ is trivial. Since we are assuming that φ' is not trivial, this case cannot happen.
- **QRR**: In this case φ is of the form $(v \to v)\sharp d \Leftarrow \Pi$ with either $v \in B_{\mathcal{C}}$ or $v \in \mathcal{V}ar$. Since $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$, we assume $\varphi' : (v' \to v')\sharp d' \Leftarrow \Pi'$ with $\Pi' \models_{\mathcal{C}} \Pi\sigma$, $d \trianglerighteq d'$ and $v\sigma = v'$ for some substitution σ. If $v \in B_{\mathcal{C}}$, then also $v' \in B_{\mathcal{C}}$ and $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi'$ can be proved with a proof tree $\mathcal{T}'$ consisting of just one **QRR** inference step. If $v \in \mathcal{V}ar$, then $v' \in \mathrm{Term}_{\perp}(\Sigma, B, \mathcal{V}ar)$, and $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi'$ can be proved with a proof tree $\mathcal{T}'$ consisting only of **QDC** and **QRR** inferences. In both cases, $|\mathcal{T}'| = 0 \leq |\mathcal{T}|$.
- **QDC**: In this case $\varphi : (c(\overline{e}_n) \to c(\overline{t}_n))\sharp d \Leftarrow \Pi$ and $\mathcal{T}$ has the form

$$\frac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(c(\overline{e}_n) \to c(\overline{t}_n))\sharp d \Leftarrow \Pi}\ \mathrm{QDC}$$

  where $c \in DC^n$, $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (e_i \to t_i)\sharp d_i \Leftarrow \Pi$ with proof tree $\mathcal{T}_i$, and $d \trianglelefteq d_i$ $(1 \leq i \leq n)$. Since $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$, we can assume that φ' has the form $(c(\overline{e'}_n) \to c(\overline{t'}_n))\sharp d' \Leftarrow \Pi'$ with $e_i\sigma \sqsubseteq e'_i$ $(1 \leq i \leq n)$, $c(\overline{t}_n)\sigma \sqsupseteq c(\overline{t'}_n)$, $d \trianglerighteq d'$ and $\Pi' \models_{\mathcal{C}} \Pi\sigma$ for some substitution σ. For $1 \leq i \leq n$, we clearly obtain $(e_i \to t_i)\sharp d_i \Leftarrow \Pi \succcurlyeq_{\mathcal{D},\mathcal{C}} (e'_i \to t'_i)\sharp d_i \Leftarrow \Pi'$, and by induction hypothesis we can assume $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (e'_i \to t'_i)\sharp d_i \Leftarrow \Pi'$ with proof tree $\mathcal{T'}_i$ such that $|\mathcal{T'}_i| \leq |\mathcal{T}_i|$. Then we get $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (c(\overline{e'}_n) \to c(\overline{t'}_n))\sharp d' \Leftarrow \Pi'$ with a proof tree $\mathcal{T}'$ such that $|\mathcal{T}'| \leq |\mathcal{T}|$. More precisely, $\mathcal{T}'$ has the form

$$\frac{(\ (e'_i \to t'_i)\sharp d_i \Leftarrow \Pi'\ )_{i=1\ldots n}}{(c(\overline{e'}_n) \to c(\overline{t'}_n))\sharp d' \Leftarrow \Pi'}\ \mathrm{QDC}$$

  where $d' \trianglelefteq d_i$ follows from $d' \trianglelefteq d \trianglelefteq d_i$ $(1 \leq i \leq n)$ and each premise is proved by $\mathcal{T'}_i$.
- **QDF$_{\mathcal{I}}$**: In this case $\varphi : (f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi$ and $\mathcal{T}$ has the form

$$\frac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi}\ \mathrm{QDF}_{\mathcal{I}}$$

  where $f \in DF^n$ and there is some non-trivial $\psi = (f(\overline{t}_n) \to t)\sharp d_0 \Leftarrow \Pi)$ such that $\psi \in \mathcal{I}$, $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (e_i \to t_i)\sharp d_i \Leftarrow \Pi$ with proof tree $\mathcal{T}_i$ and $d \trianglelefteq d_i$ $(0 \leq i \leq n)$. Since $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$, we can assume $\varphi' = (f(\overline{e'}_n) \to t')\sharp d' \Leftarrow \Pi'$ with $e_i\sigma \sqsubseteq e'_i$ $(1 \leq i \leq n)$, $t\sigma \sqsupseteq t'$, $d \trianglerighteq d'$ and $\Pi' \models_{\mathcal{C}} \Pi\sigma$ for some substitution σ. For $1 \leq i \leq n$, we get $(e_i \to t_i)\sharp d_i \Leftarrow \Pi \succcurlyeq_{\mathcal{D},\mathcal{C}} (e'_i \to t_i\sigma)\sharp d_i \Leftarrow \Pi'$, and by induction hypothesis we can assume $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (e'_i \to t_i\sigma)\sharp d_i \Leftarrow \Pi'$ with proof tree $\mathcal{T}'_i$ such that $|\mathcal{T}'_i| \le |\mathcal{T}_i|$. Consider now $\psi' = ((f(\bar{t}_n)\sigma \to t')\sharp d_0 \Leftarrow \Pi')$. Clearly, $\psi \succcurlyeq_{\mathcal{D},\mathcal{C}} \psi'$ and therefore $\psi' \in \mathcal{I}$ because $\mathcal{I}$ is closed under $(\mathcal{D},\mathcal{C})$-entailment. Using this ψ' we get $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (f(\overline{e'}_n) \to t')\sharp d' \Leftarrow \Pi'$ with a proof tree $\mathcal{T}'$ such that $|\mathcal{T}'| \le |\mathcal{T}|$. More precisely, $\mathcal{T}'$ has the form

$$\frac{(\ (e'_i \to t_i\sigma)\sharp d_i \Leftarrow \Pi'\ )_{i=1\dots n}}{(f(\overline{e'}_n) \to t')\sharp d' \Leftarrow \Pi'}\ \mathrm{QDF}_{\mathcal{I}}$$

where $d' \trianglelefteq d_i$ follows from $d' \trianglelefteq d \trianglelefteq d_i$ $(0 \le i \le n)$ and each premise is proved by $\mathcal{T}'_i$.

– **QPF**: In this case $\varphi : (p(\bar{e}_n) \to v)\sharp d \Leftarrow \Pi$ and $\mathcal{T}$ has the form

$$\frac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\dots n}}{(p(\bar{e}_n) \to v)\sharp d \Leftarrow \Pi}\ \mathrm{QPF}$$

where $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$, $\Pi \models_{\mathcal{C}} p(\bar{t}_n) \to v$, $d \trianglelefteq d_i$ and $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (e_i \to t_i)\sharp d_i \Leftarrow \Pi$ with proof tree $\mathcal{T}_i$ $(1 \le i \le n)$. Since $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$, we can assume φ' to be of the form $(p(\overline{e'}_n) \to v')\sharp d' \Leftarrow \Pi'$ with $e_i\sigma \sqsubseteq e'_i$ $(1 \le i \le n)$, $v\sigma \sqsupseteq v'$, $d \trianglerighteq d'$ and $\Pi' \models_{\mathcal{C}} \Pi\sigma$ for some substitution σ. For $1 \le i \le n$, we get $(e_i \to t_i)\sharp d_i \Leftarrow \Pi \succcurlyeq_{\mathcal{D},\mathcal{C}} (e'_i \to t_i\sigma)\sharp d_i \Leftarrow \Pi'$, and by induction hypothesis we can assume $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (e'_i \to t_i\sigma)\sharp d_i \Leftarrow \Pi'$ with proof tree $\mathcal{T}'_i$ such that $|\mathcal{T}'_i| \le |\mathcal{T}_i|$. Moreover, we can also assume $v' \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$ because p is a primitive function symbol and φ' is not trivial. From $v, v' \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$ and $v\sigma \sqsupseteq v'$ we can conclude that $v\sigma = v'$. Then, from $\Pi \models_{\mathcal{C}} p(\bar{t}_n) \to v$ and $\Pi' \models_{\mathcal{C}} \Pi\sigma$ we can deduce $\Pi' \models_{\mathcal{C}} p(\bar{t}_n)\sigma \to v'$. Putting everything together, we get $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (p(\overline{e'}_n) \to v')\sharp d' \Leftarrow \Pi'$ with a proof tree $\mathcal{T}'$ such that $|\mathcal{T}'| \le |\mathcal{T}|$. More precisely, $\mathcal{T}'$ has the form

$$\frac{(\ (e'_i \to t_i\sigma)\sharp d_i \Leftarrow \Pi'\ )_{i=1\dots n}}{(p(\overline{e'}_n) \to v')\sharp d' \Leftarrow \Pi'}\ \mathrm{QPF}$$

where $d' \trianglelefteq d_i$ follows from $d' \trianglelefteq d \trianglelefteq d_i$ $(1 \le i \le n)$ and each premise is proved by $\mathcal{T}'_i$.

– **QAC**: Similar to the case for **QPF**.

[*4.*] (*Conservation property*). Assume $\varphi : (f(\bar{t}_n) \to t)\sharp d \Leftarrow \Pi$. In the case that φ is a trivial qc-fact, it is true by definition of qc-interpretation that $\varphi \in \mathcal{I}$, and $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ follows by rule **QTI**. Therefore the property is satisfied for trivial qc-facts. If φ is not trivial, we prove each implication as follows:

– ($\Leftarrow$) Assume $\varphi \in \mathcal{I}$. Then $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with a $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$-proof tree of the form:

$$\frac{(\ (t_i \to t_i)\sharp \mathbf{t} \Leftarrow \Pi\ )_{i=1\dots n}}{(f(\bar{t}_n) \to t)\sharp d \Leftarrow \Pi}\ \mathrm{QDF}_{\mathcal{I}} \text{ using } \varphi \in \mathcal{I}$$

where each premise has an easy $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$-proof tree due to the approximation property, and $d \trianglelefteq d, \mathbf{t}$ hold trivially.

– ($\Rightarrow$) Assume $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$. As φ is not trivial, there is a $\mathcal{I}$-QCRWL($\mathcal{D},\mathcal{C}$)-proof tree of the form:

$$\frac{(\ (t_i \to t'_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(f(\overline{t}_n) \to t)\sharp d \Leftarrow \Pi}\ \mathrm{QDF}_{\mathcal{I}} \text{ using } \varphi' = (f(\overline{t'}_n) \to t)\sharp d' \Leftarrow \Pi) \in \mathcal{I}$$

where $d \trianglelefteq d', d_i$ and $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (t_i \to t'_i)\sharp d_i \Leftarrow \Pi\,(1 \le i \le n)$. For each $1 \le i \le n$, we claim that $t'_i \sqsubseteq t_i$. If $t'_i = \bot$ the claim is trivial. If $t'_i \neq \bot$, then $(t_i \to t'_i)\sharp d_i \Leftarrow \Pi$ is a non-trivial qc-production and the claim follows from $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (t_i \to t'_i)\sharp d_i \Leftarrow \Pi$ and the approximation property. Now, the claim together with $\Pi \models_{\mathcal{C}} \Pi$, $d' \trianglerighteq d$ and $t \sqsupseteq t$ yields $\varphi' \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi$. Since $\varphi' \in \mathcal{I}$ and $\mathcal{I}$ is closed under $(\mathcal{D},\mathcal{C})$-entailment, we can conclude that $\varphi \in \mathcal{I}$. □

Next, we can define program models and semantic consequence, adapting ideas from the so-called *strong semantics* of [13]. [4]

Definition 7 (Models and semantic consequence). *Let a QCFLP($\mathcal{D},\mathcal{C}$)-program $\mathcal{P}$ be given.*

1. *A qc-interpretation $\mathcal{I}$ is a model of $R_l : (f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \overline{\delta}_m) \in \mathcal{P}$ (in symbols, $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} R_l$) iff for every substitution θ, for every set of atomic primitive $\mathcal{C}$-constraints Π, for every c-term $t \in Term_\bot(\Sigma, B, \mathcal{V}ar)$ and for all $d, d_0, \ldots, d_m \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ such that $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \delta_i\theta\sharp d'_i \Leftarrow \Pi\,(1 \le i \le m)$, $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (r\theta \to t)\sharp d'_0 \Leftarrow \Pi$ and $d \trianglelefteq \alpha \circ d_i\,(0 \le i \le m)$, one has $((f(\overline{t}_n)\theta \to t)\sharp d \Leftarrow \Pi) \in \mathcal{I}$.*
2. *A qc-interpretation $\mathcal{I}$ is a* model *of $\mathcal{P}$ (in symbols, $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P}$) iff $\mathcal{I}$ is a model of every program rule belonging to $\mathcal{P}$.*
3. *A qc-statement φ is a semantic consequence of $\mathcal{P}$ (in symbols, $\mathcal{P} \models_{\mathcal{D},\mathcal{C}} \varphi$) iff $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ holds for every qc-interpretation $\mathcal{I}$ such that $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P}$.* □

3.2 Least Models

We will now present two different characterizations for the least model of a given program $\mathcal{P}$: in the first place as a least fixpoint of an interpretation transformer and in the second place as the set of qc-facts derivable from $\mathcal{P}$ in a special rewriting logic.

A fixpoint characterization of least models.

A well-known way of characterizing least program models is to exploit the lattice structure of the family of all program interpretations and to obtain the least model of a give program $\mathcal{P}$ as the least fixpoint of an interpretation transformer related to $\mathcal{P}$. Such characterizations are know for logic programming [11,2], constraint logic programming [7,6,10], constraint functional logic programming [13]

[4] Weak models and weak semantic consequence could be also defined similarly as in [13], but strong semantics suffices for the purposes of this report.

and qualified logic programming [19]. Our approach here extends that in [13] by adding qualification values.

The next result, whose easy proof is omitted, provides a lattice structure of program interpretations:

Proposition 2 (Interpretations Lattice). *$Int_{\mathcal{D},\mathcal{C}}$ defined as the set of all qc-interpretations over the qualification domain $\mathcal{D}$ and the constraint domain $\mathcal{C}$ is a complete lattice w.r.t. the set inclusion ordering ($\subseteq$). Moreover, the bottom element $\perp\!\!\!\perp$ and the top element $\top\!\!\!\top$ of this lattice are characterized as $\perp\!\!\!\perp = cl_{\mathcal{D},\mathcal{C}}(\{\varphi \mid \varphi$ is a trivial qc-fact$\})$ and $\top\!\!\!\top = \{\varphi \mid \varphi$ is any qc-fact$\}$.*

Now we define an *interpretations transformer $ST_{\mathcal{P}}$* intended to formalize the computation of immediate consequences from the qc-facts belonging to a given qc-interpretation.

Definition 8 (Interpretations transformers). *Assuming a QCFLP($\mathcal{D},\mathcal{C}$)-program $\mathcal{P}$ and a qc-interpretation $\mathcal{I}$, $ST_{\mathcal{P}} : Int_{\mathcal{D},\mathcal{C}} \rightarrow Int_{\mathcal{D},\mathcal{C}}$ is defined as $ST_{\mathcal{P}}(\mathcal{I}) =_{\mathrm{def}} cl_{\mathcal{D},\mathcal{C}}(preST_{\mathcal{P}}(\mathcal{I}))$ where the closure operator $cl_{\mathcal{D},\mathcal{C}}()$ is defined as in Def. 6 and the auxiliary interpretation pre-transformer $preST_{\mathcal{P}}$ acts as follows:*

$preST_{\mathcal{P}}(\mathcal{I}) =_{\mathrm{def}} \{(f(\overline{t}_n)\theta \rightarrow t)\sharp d \Leftarrow \Pi \mid$ *there are*
some $(f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \overline{\delta}_m) \in \mathcal{P}$,
some substitution θ,
some set Π *of primitive atomic* $\mathcal{C}$*-constraints* ,
some c-term $t \in Term_{\perp}(\Sigma, B, \mathcal{V}ar)$, *and*
some qualification values $d_0, d_1, \ldots, d_m \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ *such that*
– $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \delta_i\theta\sharp d_i \Leftarrow \Pi$ $(1 \leq i \leq m)$,
– $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (r\theta \rightarrow t)\sharp d_0 \Leftarrow \Pi$, *and*
– $d \trianglelefteq \alpha \circ d_i$ $(0 \leq i \leq m)$
$\}$.

Proposition 3 below shows that $preST_{\mathcal{P}}(\mathcal{I})$ is closed under $(\mathcal{D},\mathcal{C})$-entailment. Its proof relies on the next technical, but easy result:

Lemma 2 (Auxiliary Result). *Given terms $t, t' \in Term_{\perp}(\Sigma, B, \mathcal{V}ar)$ and a substitution η such that t is linear and $t\eta \sqsubseteq t'$, there is some substitution η' such that:*

1. $t\eta' = t'$,
2. $\eta \sqsubseteq \eta'$ *(i.e. $X\eta \sqsubseteq X\eta'$ for all $X \in \mathcal{V}ar$)* , *and*
3. $\eta = \eta'$ $[\backslash var(t)]$.

Proof. Since t is linear, for each variable X occurring in t there is one single position p such that X occurs in t at position p. Let p_X be this position. Since $t\theta \sqsubseteq t'$, there must be a subterm t'_X occurring in t' at position p_X such that $X\eta \sqsubseteq t'_X$. Let η' be a substitution such that $X\eta' = t'_X$ for each variable X occurring in t, and $Y\eta' = Y\theta$ for each variable Y not occurring in t. It is easy to check that η' has all the desired properties. $\square$

Proposition 3 ($preST_{\mathcal{P}}(\mathcal{I})$ **is closed under** $(\mathcal{D},\mathcal{C})$**-entailment).** *Assume two qc-facts* φ *and* φ'*. If* $\varphi \in preST_{\mathcal{P}}(\mathcal{I})$ *and* $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$*, then* $\varphi' \in preST_{\mathcal{P}}(\mathcal{I})$*.*

Proof. Since $\varphi \in preST_{\mathcal{P}}(\mathcal{I})$, there are some $R_l : (f(\bar{t}_n) \xrightarrow{\alpha} r \Leftarrow \bar{\delta}_m) \in \mathcal{P}$ and some substitution θ such that $\varphi : (f(\bar{t}_n)\theta \to t)\sharp d \Leftarrow \Pi$ and

- (1) $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \delta_i\theta\sharp d_i \Leftarrow \Pi$ $(1 \le i \le m)$,
- (2) $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (r\theta \to t)\sharp d_0 \Leftarrow \Pi$, and
- (3) $d \trianglelefteq \alpha \circ d_i$ $(0 \le i \le m)$.

Since $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi'$, we can assume $\varphi' : (f(\overline{t'}_n) \to t')\sharp d' \Leftarrow \Pi'$ and a substitution σ such that $t_i\theta\sigma \sqsubseteq t'_i$ $(1 \le i \le n)$, $t\sigma \sqsupseteq t'$, (4) $d \trianglerighteq d'$ and $\Pi' \models_{\mathcal{C}} \Pi\sigma$.

Given that $\bar{t}_n$ is a linear tuple of terms, and applying Lemma 2 with $\eta = \theta\sigma$, we obtain a substitution η' satisfying $t_i\eta' = t'_i$ $(1 \le i \le n)$, $\theta\sigma \sqsubseteq \eta'$ and $\theta\sigma = \eta'$ $[\backslash\mathrm{var}(\bar{t}_n)]$. Now, in order to prove $\varphi' \in preST_{\mathcal{P}}(\mathcal{I})$ it suffices to consider R_l, η' and some some $d'_0, d'_1, \ldots, d'_m \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ satisfying:

- (1') $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \delta_i\eta'\sharp d'_i \Leftarrow \Pi'$ $(1 \le i \le m)$,
- (2') $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (r\eta' \to t')\sharp d'_0 \Leftarrow \Pi'$, and
- (3') $d' \trianglelefteq \alpha \circ d'_i$ $(0 \le i \le m)$.

Let us see that (1'), (2') and (3') hold when choosing $d'_i = d_i$ $(0 \le i \le m)$:

[1'] For any $1 \le i \le m$ we have $\delta_i\theta\sharp d_i \Leftarrow \Pi \succcurlyeq_{\mathcal{D},\mathcal{C}} \delta_i\eta'\sharp d_i \Leftarrow \Pi'$ using σ, because $\delta_i\theta\sigma \sqsubseteq \delta_i\eta'$, $d_i \trianglerighteq d_i$ and $\Pi' \models_{\mathcal{C}} \Pi\sigma$. Therefore (1) $\Rightarrow$ (1') by the entailment property (Lemma 1(3)).

[2'] Similarly as for (1'), $(r\theta \to t)\sharp d_0 \Leftarrow \Pi \succcurlyeq_{\mathcal{D},\mathcal{C}} (r\theta' \to t')\sharp d_0\mu \Leftarrow \Pi'$ using σ, because $r\theta\sigma \sqsubseteq r\eta'$, $t\sigma \sqsupseteq t'$, $d_0 \trianglerighteq d_0$ and $\Pi' \models_{\mathcal{C}} \Pi\sigma$. Therefore (2) $\Rightarrow$ (2') again by the entailment property (Lemma 1(3)).

[3'] From (3) and (4) we trivially get $d' \trianglelefteq \alpha \circ d_i$ $(0 \le i \le m)$. Therefore, (3') holds when choosing $d'_i = d_i$ $(0 \le i \le m)$. ☐

As a consequence of the previous proposition, we can establish a stronger relation between $ST_{\mathcal{P}}(\mathcal{I})$ and $preST_{\mathcal{P}}(\mathcal{I})$ for non-trivial qc-facts, as given in the following lemma.

Lemma 3 ($ST_{\mathcal{P}}(\mathcal{I})$ **versus** $preST_{\mathcal{P}}(\mathcal{I})$**).** *For any non-trivial qc-fact* φ *one has:* $\varphi \in ST_{\mathcal{P}}(\mathcal{I}) \Longrightarrow \varphi \in preST_{\mathcal{P}}(\mathcal{I})$*.*

Proof. From $\varphi \in ST_{\mathcal{P}}(\mathcal{I})$ it follows by definition of $ST_{\mathcal{P}}$ that $\varphi \in \mathrm{cl}_{\mathcal{D},\mathcal{C}}(preST_{\mathcal{P}}(\mathcal{I}))$. As we are assuming that φ is not trivial, there must be some $\psi \in preST_{\mathcal{P}}(\mathcal{I})$ such that $\psi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi$. Then $\varphi \in preST_{\mathcal{P}}(\mathcal{I})$ follows from Proposition 3. ☐

The main properties of the interpretation transformer $ST_{\mathcal{P}}$ are given in the following proposition.

Proposition 4 (Properties of interpretation transformers). *Let* $\mathcal{P}$ *be a QCFLP*$(\mathcal{D},\mathcal{C})$*-program. Then:*

1. $ST_{\mathcal{P}}$ *is monotonic and continuous.*

2. *For any* $\mathcal{I} \in Int_{\mathcal{D},\mathcal{C}}$: $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P} \Longleftrightarrow ST_{\mathcal{P}}(\mathcal{I}) \subseteq \mathcal{I}$.

Proof. Monotonicity and continuity are well-known results for similar semantics; see e.g. Prop. 3 in [13]. Item 2 can be proved as follows: as an easy consequence of Def. 7, $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P} \Longleftrightarrow preST_{\mathcal{P}}(\mathcal{I}) \subseteq \mathcal{I}$. Moreover, $preST_{\mathcal{P}}(\mathcal{I}) \subseteq \mathcal{I} \Longleftrightarrow \mathrm{cl}_{\mathcal{D},\mathcal{C}}(preST_{\mathcal{P}}(\mathcal{I})) \subseteq \mathrm{cl}_{\mathcal{D},\mathcal{C}}(\mathcal{I}) \Longleftrightarrow ST_{\mathcal{P}}(\mathcal{I}) \subseteq \mathcal{I}$, where the first equivalence is obvious and the second equivalence is due to the equalities $\mathrm{cl}_{\mathcal{D},\mathcal{C}}(preST_{\mathcal{P}}(\mathcal{I})) = ST_{\mathcal{P}}(\mathcal{I})$ and $\mathrm{cl}_{\mathcal{D},\mathcal{C}}(\mathcal{I}) = \mathcal{I}$. Therefore, $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P} \Longleftrightarrow ST_{\mathcal{P}}(\mathcal{I}) \subseteq \mathcal{I}$, as desired. □

Finally, we can conclude that the least fixpoint of $ST_{\mathcal{P}}$ characterizes the least model of any given QCFLP($\mathcal{D},\mathcal{C}$)-program $\mathcal{P}$, as stated in the following theorem.

Theorem 1. *For every QCFLP($\mathcal{D},\mathcal{C}$)-program $\mathcal{P}$ there exists the* least model $S_{\mathcal{P}} = lfp(ST_{\mathcal{P}}) = \bigcup_{k\in\mathbb{N}} ST_{\mathcal{P}}\!\uparrow^{k}(\perp\!\!\!\perp)$.

Proof. Due to a well-known theorem by Knaster and Tarski [22], a monotonic mapping from a complete lattice into itself always has a least fixpoint which is also its least pre-fixpoint. In the case that the mapping is continuous, its least fixpoint can be characterized as the lub of the sequence of lattice elements obtained by reiterated application of the mapping to the bottom element. Combining these results with Prop. 4 trivially proves the theorem. □

A qualified constraint rewriting logic.

In order to obtain a logical view of program semantics and an alternative characterization of least program models, we define the *Qualified Constrained Rewriting Logic for Programs* QCRWL($\mathcal{D},\mathcal{C}$) as the formal system consisting of the six inference rules displayed in Fig. 3. Note that QCRWL($\mathcal{D},\mathcal{C}$) is very similar Qualified Constrained Rewriting Logic for Interpretations $\mathcal{I}$-QCRWL($\mathcal{D},\mathcal{C}$) (see Fig. 2), except that the inference rule $\mathbf{QDF}_{\mathcal{I}}$ from $\mathcal{I}$-QCRWL($\mathcal{D},\mathcal{C}$) is replaced by the inference rule $\mathbf{QDF}_{\mathcal{P}}$ in QCRWL($\mathcal{D},\mathcal{C}$). The inference rules in QCRWL($\mathcal{D},\mathcal{C}$) formalize provability of qc-statements from a given program $\mathcal{P}$ according to their intuitive meanings. In particular, $\mathbf{QDF}_{\mathcal{P}}$ formalizes the behavior of program rules and attenuation factors that was informally explained in the Introduction, using the set $[\mathcal{P}]_{\perp}$ of *program rule instances.*

In the sequel we use the notation $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ to indicate that φ can be inferred from $\mathcal{P}$ in QCRWL($\mathcal{D},\mathcal{C}$). By convention, we agree that no other inference rule is used whenever **QTI** is applicable. Therefore, trivial qc-statements can only be inferred by rule **QTI**. As usual in formal inference systems, QCRWL($\mathcal{D},\mathcal{C}$) proofs can be represented as trees whose nodes correspond to inference steps. For example, if $\mathcal{P}$ is the library program, Π is empty, and ψ is

```
(guessGenre(book(4,"Beim Hauten der Zwiebel","Gunter Grass",
    "German","Biography", medium, 432)) --> "Essay")#0.7
```

QTI $\dfrac{}{\varphi}$ if φ is a trivial qc-statement.

QRR $\dfrac{}{(v \to v)\sharp d \Leftarrow \Pi}$ if $v \in \mathcal{V}ar \cup B_{\mathcal{C}}$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$.

QDC $\dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(c(\overline{e}_n) \to c(\overline{t}_n))\sharp d \Leftarrow \Pi}$ if $c \in DC^n$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(1 \le i \le n)$.

QDF$_{\mathcal{P}}$ $\dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n} \quad (r \to t)\sharp d'_0 \Leftarrow \Pi \quad (\delta_j \sharp d'_j \Leftarrow \Pi)_{j=1\ldots m}}{(f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi}$

if $f \in DF^n$ and $(f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \delta_1, \ldots, \delta_m) \in [\mathcal{P}]_\perp$
where $[\mathcal{P}]_\perp = \{R_l\theta \mid R_l$ is a rule in $\mathcal{P}$ and θ is a substitution$\}$,
and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(1 \le i \le n)$, $d \trianglelefteq \alpha \circ d'_j$ $(0 \le j \le m)$.

QPF $\dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{e}_n) \to v)\sharp d \Leftarrow \Pi}$ if $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$,

$\Pi \models_{\mathcal{C}} p(\overline{t}_n) \to v$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(1 \le i \le n)$.

QAC $\dfrac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{e}_n) == v)\sharp d \Leftarrow \Pi}$ if $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$,

$\Pi \models_{\mathcal{C}} p(\overline{t}_n) == v$ and $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ verifies $d \trianglelefteq d_i$ $(1 \le i \le n)$.

Fig. 3. Qualified Constrained Rewriting Logic for Programs

then $\mathcal{P} \vdash_{\mathcal{U},\mathcal{R}} \psi \Leftarrow \Pi$ with a proof tree whose root inference may be chosen as **QDF**$_{\mathcal{P}}$ using a suitable instance of the fourth program rule for `guessGenre`.

The following lemma states the main properties of QCRWL$(\mathcal{D}, \mathcal{C})$. The proof is similar to that of Lemma 1 and omitted here. The interested reader is also referred to the proof of Lemma 2 in [13].

Lemma 4 (Some properties of QCRWL$(\mathcal{D}, \mathcal{C})$). *The three first items of Lemma 1 also hold for QCRWL$(\mathcal{D}, \mathcal{C})$, with the natural reformulation of their statements. More precisely:*

1. *Approximation property: For any non-trivial φ of the form $(t \to t')\sharp d \Leftarrow \Pi$ where $t, t' \in Term_\perp(\Sigma, B, \mathcal{V}ar)$, the three following affirmations are equivalent: (a) $t \sqsupseteq t'$; (b) $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ with an easy proof tree; and (c) $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$.*
2. *Primitive c-atoms: For any primitive c-atom $p(\overline{t}_n) == v$, one has $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} (p(\overline{t}_n) == v)\sharp d \Leftarrow \Pi \iff \Pi \models_{\mathcal{C}} p(\overline{t}_n) == v$.*
3. *Entailment property: $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ with a proof tree $\mathcal{T}$ and $\varphi \succcurlyeq_{\mathcal{D},\mathcal{C}} \varphi' \implies \mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi'$ with a proof tree $\mathcal{T}'$ such that $|\mathcal{T}'| \le |\mathcal{T}|$.*

The next theorem is the main result in this section. It provides a nice equivalence between QCRWL$(\mathcal{D}, \mathcal{C})$-derivability and semantic consequence in the sense of Definition 7 (*soundness* and *completeness* properties), as well as a characterization of least program models in terms of QCRWL$(\mathcal{D},\mathcal{C})$-derivability (*canonicity property*).

Theorem 2 (QCRWL$(\mathcal{D},\mathcal{C})$ characterizes program semantics). *For any QCFLP$(\mathcal{D},\mathcal{C})$-program $\mathcal{P}$ and any qc-statement φ, the following three conditions are equivalent:*

$$(a) \quad \mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi \qquad (b) \quad \mathcal{P} \models_{\mathcal{D},\mathcal{C}} \varphi \qquad (c) \quad S_{\mathcal{P}} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$$

Moreover, we also have:

1. *Soundness: for any qc-statement φ, $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi \Longrightarrow \mathcal{P} \models_{\mathcal{D},\mathcal{C}} \varphi$.*
2. *Completeness: for any qc-statement φ, $\mathcal{P} \models_{\mathcal{D},\mathcal{C}} \varphi \Longrightarrow \mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$.*
3. *Canonicity: $S_{\mathcal{P}} = \{\varphi \mid \varphi$ is a qc-fact and $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi\}$.*

Proof. Assuming the equivalence between (a), (b) and (c), soundness and completeness are a trivial consequence of the equivalence between (a) and (b), and canonicity holds because of the equivalences $\varphi \in S_{\mathcal{P}} \iff S_{\mathcal{P}} \Vdash_{\mathcal{D},\mathcal{C}} \varphi \iff \mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$, which follow from the conservation property from Lemma 1 and the equivalence between (c) and (a). The rest of the proof consists of separate proofs for the three implications $(a) \Rightarrow (b)$, $(b) \Rightarrow (c)$ and $(c) \Rightarrow (a)$.

$[(a) \Rightarrow (b)]$ We assume (a), i.e., $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ with a QCRWL$(\mathcal{D},\mathcal{C})$-proof tree $\mathcal{T}_{\mathcal{P}}$ including $k \geq 1$ QCRWL$(\mathcal{D},\mathcal{C})$-inference steps. In order to prove (b) we also assume a qc-interpretation $\mathcal{I}$ such that $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P}$. We must prove $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with some QCRWL$(\mathcal{D},\mathcal{C})$-proof tree $\mathcal{T}_{\mathcal{I}}$. This follows easily by induction on k, using the fact that each QCRWL$(\mathcal{D},\mathcal{C})$-inference rule **QRL** is *sound* in the following sense: each inference step

$$\frac{\varphi_1 \quad \cdots \quad \varphi_n}{\varphi} \ \mathrm{QRL}$$

verifying $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi_i$ $(1 \leq i \leq n)$ (i.e., the premises are valid in $\mathcal{I}$) also verifies $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ (i.e., the conclusion is valid in $\mathcal{I}$). For **QRL** other than **QDF**$_{\mathcal{P}}$, soundness of **QRL** does not depend on the assumption $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P}$; it can be easily proved by using the homonomous $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$-inference rule **QRL**. In the case of **QDF**$_{\mathcal{P}}$, φ has the form $f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi$ and the validity of the premises in $\mathcal{I}$ means the following:

- (1) $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (e_i \to t_i)\sharp d_i \Leftarrow \Pi$ $(1 \leq i \leq n)$,
- (2) $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (r \to t)\sharp d'_0 \Leftarrow \Pi$, and
- (3) $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} \delta_j\sharp d'_j \Leftarrow \Pi$ $(1 \leq j \leq m)$

with $f \in DF^n$, $(f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \delta_1, \cdots, \delta_m) \in [\mathcal{P}]_\perp$, $d \triangleleft d_i$ $(1 \leq i \leq n)$ and $d \triangleleft \alpha \circ d'_j$ $(0 \leq j \leq m)$. Then, from the assumption $\mathcal{I} \models_{\mathcal{D},\mathcal{C}} \mathcal{P}$ and Def. 7 we obtain

- (4) $((f(\overline{t}_n) \to t)\sharp d \Leftarrow \Pi) \in \mathcal{I}$.

Finally, from (1), (4) we conclude that $(f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi$ can de derived by means of a $\mathbf{QDF}_\mathcal{I}$-inference step from premises $(e_i \to t_i)\sharp d_i \Leftarrow \Pi$ $(1 \le i \le n)$. Therefore, $\mathcal{I} \Vdash_{\mathcal{D},\mathcal{C}} (f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi$, as desired.

$[(b) \Rightarrow (c)]$ Straightforward, given that $S_\mathcal{P} \models_{\mathcal{D},\mathcal{C}} \mathcal{P}$, as proved in Th. 1.

$[(c) \Rightarrow (a)]$ Let φ be any c-statement and assume $S_\mathcal{P} \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with proof tree $\mathcal{T}$. Note that $\mathcal{T}$ includes a finite number of $\mathbf{QDF}_\mathcal{I}$-inference steps with $\mathcal{I} = S_\mathcal{P}$, relying on finitely many qc-facts $\psi_i \in S_\mathcal{P}$ $(1 \le i \le p)$. As $S_\mathcal{P} = \bigcup_{k\in\mathbb{N}} ST_\mathcal{P}\!\uparrow^k(\perp\!\!\!\perp)$ because of Th. 1, there must exist some $k \in \mathbb{N}$ such that all the ψ_i $(1 \le i \le p)$ belong to $ST_\mathcal{P}\!\uparrow^k(\perp\!\!\!\perp)$ and thus $ST_\mathcal{P}\!\uparrow^k(\perp\!\!\!\perp) \Vdash_{\mathcal{D},\mathcal{C}} \varphi$. Therefore, it is enough to prove by induction on k that

$$ST_\mathcal{P}\!\uparrow^k(\perp\!\!\!\perp) \Vdash_{\mathcal{D},\mathcal{C}} \varphi \Longrightarrow \mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$$

Basis (k=0). Assume $ST_\mathcal{P}\!\uparrow^0(\perp\!\!\!\perp) \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$-proof tree $\mathcal{T}$. As $ST_\mathcal{P}\!\uparrow^0(\perp\!\!\!\perp) = \perp\!\!\!\perp$, which only includes trivial qc-facts and $\mathbf{QDF}_\mathcal{I}$ always uses non-trivial qc-facts, $\mathcal{T}$ cannot include $\mathbf{QDF}_\mathcal{I}$-inference steps. Hence, $\mathcal{T}$ also serves as a QCRWL$(\mathcal{D},\mathcal{C})$-proof tree which includes no $\mathbf{QDF}_\mathcal{P}$-inference steps and proves $ST_\mathcal{P}\!\uparrow^0(\perp\!\!\!\perp) \vdash_{\mathcal{D},\mathcal{C}} \varphi$.

Inductive step (k>0). Assume $ST_\mathcal{P}\!\uparrow^{k+1}(\perp\!\!\!\perp) \Vdash_{\mathcal{D},\mathcal{C}} \varphi$ with $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$-proof tree $\mathcal{T}$. Then $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ can be proved by an auxiliary induction on the size of $\mathcal{T}$, measured as its number of nodes. The reasoning must distinguish six cases, according to the $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$-inference rule $\mathbf{QRL}$ used to infer φ at the root of $\mathcal{T}$. Here we present only the most interesting case, when $\mathbf{QRL}$ is $\mathbf{QDF}_\mathcal{I}$. In this case, φ is a non-trivial qc-statement of the form $(f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi$, and $\mathcal{T}$ has the form

$$\frac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\dots n}}{\varphi : (f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi}\ \mathrm{QDF}_\mathcal{I}$$

with non-trivial $\psi : ((f(\overline{t}_n) \to t)\sharp d_0 \Leftarrow \Pi) \in ST_\mathcal{P}\!\uparrow^{k+1}(\perp\!\!\!\perp)$, $d \trianglelefteq d_i$ $(0 \le i \le n)$, and $ST_\mathcal{P}\!\uparrow^{k+1}(\perp\!\!\!\perp) \Vdash_{\mathcal{D},\mathcal{C}} (e_i \to t_i)\sharp d_i \Leftarrow \Pi$ proved by $\mathcal{I}$-QCRWL$(\mathcal{D},\mathcal{C})$-proof trees $\mathcal{T}_i$ wit sizes smaller than the size of $\mathcal{T}$ $(1 \le i \le n)$. Therefore, the inductive hypothesis of the nested induction guarantees

- (1) $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} (e_i \to t_i)\sharp d_i \Leftarrow \Pi$ with QCRWL$(\mathcal{D},\mathcal{C})$-proof trees $\hat{\mathcal{T}}_i$ $(1 \le i \le n)$

On the other hand, Lemma 3 ensures $\psi \in preST_\mathcal{P}(ST_\mathcal{P}\!\uparrow^k(\perp\!\!\!\perp))$. Therefore, recalling Def. 8, there must exist $f(\overline{s}_n) \xrightarrow{\alpha} r \Leftarrow \overline{\delta}_m \in \mathcal{P}$, a substitution θ and qualification values $d'_0, d'_1, \dots, d'_m$ satisfying $s_i\theta = t_i$ $(1 \le i \le n)$ and

- (2) $ST_\mathcal{P}\!\uparrow^k(\perp\!\!\!\perp) \Vdash_{\mathcal{D},\mathcal{C}} \delta_j\theta\sharp d'_j \Leftarrow \Pi$ $(1 \le j \le m)$
- (3) $ST_\mathcal{P}\!\uparrow^k(\perp\!\!\!\perp) \Vdash_{\mathcal{D},\mathcal{C}} (r\theta \to t)\sharp d'_0 \Leftarrow \Pi$
- (4) $d_0 \trianglelefteq \alpha \circ d'_j$ $(0 \le j \le m)$

By the inductive hypothesis of the main induction, applied to (2) and (3), we get:

- (5) $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \delta_j\theta\sharp d'_j \Leftarrow \Pi$ with QCRWL$(\mathcal{D},\mathcal{C})$-proof trees $\hat{\mathcal{T}}'_j$ $(1 \le j \le m)$

– (6) $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} (r\theta \to t)\sharp d'_0 \Leftarrow \Pi$ with QCRWL($\mathcal{D},\mathcal{C}$)-proof tree $\hat{\mathcal{T}}'$

From $d \trianglelefteq d_i$ $(0 \le i \le n)$ and (4) we also obtain:

– (7) $d \trianglelefteq d_i$ $(0 \le i \le n)$, $d \trianglelefteq \alpha \circ d'_j$ $(0 \le j \le m)$

Finally, we can prove $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ with a QCRWL($\mathcal{D},\mathcal{C}$)-proof tree $\hat{\mathcal{T}}$ of the form:

$$\frac{((e_i \to s_i\theta)\sharp d_i \Leftarrow \Pi)_{i=1\ldots n} \quad (r\theta \to t)\sharp d'_0 \Leftarrow \Pi \quad (\delta_j\theta\sharp d'_j \Leftarrow \Pi)_{j=1\ldots m}}{\varphi : (f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi}\ \mathrm{QDF}_{\mathcal{P}}$$

using the program rule instance $(f(\overline{s}_n) \xrightarrow{\alpha} r \Leftarrow \overline{\delta}_m)\theta \in [\mathcal{P}]_\perp$, where (5) and (6) provide proof trees for deriving the premises and (7) ensures the additional conditions required by the $\mathbf{QDF}_{\mathcal{P}}$ inference at the root of $\hat{\mathcal{T}}$. □

3.3 Goals and their Solutions

In all declarative programming paradigms, programs are generally used by placing goals and computing answers for them. In this brief subsection we define the syntax of QCFLP($\mathcal{D},\mathcal{C}$)-goals and we give a declarative characterization of goal solutions, based on the QCRWL($\mathcal{D},\mathcal{C}$) logic. This will allow formal proofs of correctness for the goal solving methods presented in Section 4.

Definition 9 (QCFLP($\mathcal{D},\mathcal{C}$)-Goals and their Solutions). *Assume a a countable set $\mathcal{W}ar$ of so-called* qualification variables W, *disjoint from $\mathcal{V}ar$ and $\mathcal{C}$'s signature Σ, and a QCFLP($\mathcal{D},\mathcal{C}$)-program $\mathcal{P}$. Then:*

1. *A* goal *G for $\mathcal{P}$ has the form $\delta_1\sharp W_1, \ldots, \delta_m\sharp W_m \;[]\; W_1 \trianglerighteq \beta_1, \ldots, W_m \trianglerighteq \beta_m$, abbreviated as $(\ \delta_i\sharp W_i,\ W_i \trianglerighteq \beta_i\ )_{i=1\ldots m}$, where $\delta_j\sharp W_j$ $(1 \le j \le m)$ are atomic $\mathcal{C}$-constraints annotated with different qualification variables W_i, and $W_i \trianglerighteq \beta_i$ are so-called* threshold conditions, *with $\beta_i \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ $(1 \le i \le m)$.*
2. *A* solution *for G is any triple $\langle \sigma, \mu, \Pi \rangle$ such that σ is a substitution, μ is a $\mathcal{D}$-valuation, Π is a finite set of atomic primitive $\mathcal{C}$-constraints, and the following two conditions hold for all $1 \le i \le m$: $W_i\mu = d_i \trianglerighteq \beta_i$, and $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} (\delta_i\sigma)\sharp d_i \Leftarrow \Pi$. The set of all solutions for G is noted $Sol_{\mathcal{P}}(G)$.* □

Thanks to the *Canonicity* property of Theorem 2, solutions of $\mathcal{P}$ are valid in the least model $S_{\mathcal{P}}$ and hence in all models of $\mathcal{P}$. A goal for the library program and one solution for it have been presented in the Introduction. In this particular example, $\Pi = \emptyset$ and the QCRWL($\mathcal{U},\mathcal{R}$) proof needed to check the solution according to Definition 9 can be formalized by following the intuitive ideas sketched in the Introduction.

4 Implementation by Program Transformation

Goal solving in instances of the CFLP($\mathcal{C}$) scheme from [13] has been formalized by means of *constrained narrowing* procedures as e.g. [12,5], and is supported by systems such as Curry [9] and $\mathcal{TOY}$ [3]. In this section we present a semantically correct transformation from QCFLP($\mathcal{D},\mathcal{C}$) into the first-order fragment of CFLP($\mathcal{C}$) which can be used for implementing goal solving in QCFLP($\mathcal{D},\mathcal{C}$).

By abuse of notation, the first-order fragment of the CFLP($\mathcal{C}$) scheme will be noted simply as CFLP($\mathcal{C}$) in the sequel. A formal description of CFLP($\mathcal{C}$) can be found in [13]; it is easily derived from the previous Section 3 by simply omitting everything related to qualification domains and values. Programs $\mathcal{P}$ are sets of program rules of the form $f(\bar{t}_n) \to r \Leftarrow \Delta$, with no attenuation factors attached. Program semantics relies on inference mechanisms for deriving *c-staments* from programs. In analogy to Def. 4, a c-statement φ may be a c-production $e \to t \Leftarrow \Pi$ or a c-atom $\delta \Leftarrow \Pi$. In analogy to Def. 6, *c-interpretations* are defined as sets of c-statements closed under a $\mathcal{C}$-entailment relation. Program models and semantic consequence are defined similarly as in Def. 7. Results similar to Th. 1 and Th. 2 can be obtained to characterize program semantics in terms of an interpretation transformer and a rewriting logic CRWL($\mathcal{C}$), respectively.

For the purposes of this section it is enough to focus on CRWL($\mathcal{C}$), which is a formal system consisting of the six inference rules displayed in Fig. 4. They are quite similar to the QCRWL($\mathcal{D},\mathcal{C}$)-inference rules from Fig. 3, except that attenuation factors and qualification values are absent.

TI $\quad \dfrac{}{\varphi}$ if φ is a trivial c-statement.

RR $\quad \dfrac{}{v \to v \Leftarrow \Pi}$ if $v \in \mathcal{V}ar \cup B_\mathcal{C}$.

DC $\quad \dfrac{(\ e_i \to t_i \Leftarrow \Pi\ )_{i=1\ldots n}}{c(\bar{e}_n) \to c(\bar{t}_n) \Leftarrow \Pi}$ if $c \in DC^n$.

DF$_\mathcal{P}$ $\quad \dfrac{(\ e_i \to t_i \Leftarrow \Pi\ )_{i=1\ldots n} \quad r \to t \Leftarrow \Pi \quad (\ \delta_j \Leftarrow \Pi\ )_{j=1\ldots m}}{f(\bar{e}_n) \to t \Leftarrow \Pi}$

if $f \in DF^n$ and $(f(\bar{t}_n) \xrightarrow{\alpha} r \Leftarrow \delta_1, \ldots, \delta_m) \in [\mathcal{P}]_\perp$
where $[\mathcal{P}]_\perp = \{R_l\theta \mid R_l$ is a rule in $\mathcal{P}$ and θ is a substitution$\}$.

PF $\quad \dfrac{(\ e_i \to t_i \Leftarrow \Pi\ )_{i=1\ldots n}}{p(\bar{e}_n) \to v \Leftarrow \Pi}$ if $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_\mathcal{C}$ and $\Pi \models_\mathcal{C} p(\bar{t}_n) \to v$.

AC $\quad \dfrac{(\ e_i \to t_i \Leftarrow \Pi\ )_{i=1\ldots n}}{p(\bar{e}_n) == v \Leftarrow \Pi}$ if $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_\mathcal{C}$ and $\Pi \models_\mathcal{C} p(\bar{t}_n) == v$.

Fig. 4. First Order Constrained Rewriting Logic

The notation $\mathcal{P} \vdash_\mathcal{C} \varphi$ indicates that φ can be inferred from $\mathcal{P}$ in CRWL($\mathcal{C}$). In analogy to the Canonicity Property from Th. 2, it is possible to prove that the least model of $\mathcal{P}$ w.r.t. set inclusion can be characterized as $S_{\mathcal{P}} = \{\varphi \mid \varphi$ is a c-fact and $\mathcal{P} \vdash_{\mathcal{C}} \varphi\}$. Therefore, working with formal inference in the rewrite logics QCRWL($\mathcal{D},\mathcal{C}$) and CRWL($\mathcal{C}$) is sufficient for proving the semantic correctness of the transformations presented in the rest of this section.

The following definition is similar to Def. 9. It will be useful for proving the correctness of the goal solving procedure for QCFLP($\mathcal{D},\mathcal{C}$)-goals discussed in the final part of this section.

Definition 10 (CFLP($\mathcal{C}$)-Goals and their Solutions). *Assume a CFLP($\mathcal{C}$)-program $\mathcal{P}$. Then:*

1. *A* goal *G for $\mathcal{P}$ has the form $\delta_1, \ldots, \delta_m$ where δ_j are atomic $\mathcal{C}$-constraints.*
2. *A* solution *for G is any pair $\langle \sigma, \Pi \rangle$ such that σ is a substitution, Π is a finite set of atomic primitive $\mathcal{C}$-constraints, and $\mathcal{P} \vdash_{\mathcal{C}} \delta_j\sigma \Leftarrow \Pi$ holds for $1 \leq j \leq m$. The set of all solutions for G is noted $Sol_{\mathcal{P}}(G)$.* □

Now we are ready to describe a semantically correct transformation from QCFLP($\mathcal{D},\mathcal{C}$) into CFLP($\mathcal{C}$). The transformation goes from a source signature Σ into a target signature Σ' such that each $f \in DF^n$ in Σ becomes $f' \in DF^{n+1}$ in Σ', and all the other symbols in Σ remain the same in Σ'. There are four group of transformation rules displayed in Figure 5 and designed to transform expressions, qc-statements, program rules and goals, respectively. The transformation works by introducing fresh qualification variables W to represent the qualification values attached to the results of calls to defined functions, as well as qualification constraints to be imposed on the values of qualification variables. Let us comment the four groups of rules in order.

Transforming any expression e yields a triple $e^{\mathcal{T}} = (e', \Omega, \mathcal{W})$, where Ω is a set of qualification constraints and $\mathcal{W}$ is the set of qualification variables occurring in e' at outermost positions. This set is relevant because the qualification value attached to e cannot exceed the infimum in $\mathcal{D}$ of the values of the variables $W \in \mathcal{W}$, and $e^{\mathcal{T}}$ is computed by recursion on e's syntactic structure as specified by the transformation rules **TAE**, **TCE**$_1$ and **TCE**$_2$. Note that **TCE**$_2$ introduces a new qualification variable W for each call to a defined function $f \in DF^n$ and builds a set Ω' of qualification constraints ensuring that W must be interpreted as a qualification value not greater than the qualification values attached to f's arguments. **TCE**$_1$ deals with calls to constructors and primitive functions just by collecting information from the arguments, and **TAE** is self-explanatory.

Unconditional productions and atomic constraints are transformed by means of **TP** and **TA**, respectively, relying on the transformation of expressions in the obvious way. Relying on **TP** and **TA**, **TCS** transforms any qc-statement of the form $\psi\sharp d \Leftarrow \Pi$ into a c-statement whose conditional part includes, in addition to Π, the qualification constraints Ω coming from $\psi^{\mathcal{T}}$ and extra qualification constraints ensuring that d is not greater than allowed by ψ's qualification.

Program rules are transformed by **TPR**. Transforming the left-hand side $f(\overline{t}_n)$ introduces a fresh symbol $f' \in DF^{n+1}$ and a fresh qualification variable W. The transformed right-hand side r' comes from $r^{\mathcal{T}}$, and the transformed conditions are obtained from the constraints coming from $r^{\mathcal{T}}$ and ${\delta_i}^{\mathcal{T}} (1 \leq i \leq m)$

Transforming Expressions

TAE $$\frac{}{v^{\mathcal{T}} = (v, \emptyset, \emptyset)} \quad \text{if } v \in \mathcal{V}ar \cup B_{\mathcal{C}}.$$

TCE$_1$ $$\frac{(\ e_i{}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)\ )_{i=1\ldots n}}{h(\overline{e}_n)^{\mathcal{T}} = (h(\overline{e'}_n),\ \bigcup_{i=1}^n \Omega_i,\ \bigcup_{i=1}^n \mathcal{W}_i)} \quad \text{if } h \in DC^n \cup PF^n.$$

TCE$_2$ $$\frac{(\ e_i{}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)\ )_{i=1\ldots n}}{f(\overline{e}_n)^{\mathcal{T}} = (f'(\overline{e'}_n, W), \Omega', \{W\})}$$

if $f \in DF^n$ and W is a fresh variable,
where $\Omega' = (\bigcup_{i=1}^n \Omega_i) \cup \{\mathsf{qVal}(W)\} \cup \{\ulcorner W \trianglelefteq W' \urcorner \mid W' \in \bigcup_{i=1}^n \mathcal{W}_i\}$.

Transforming qc-Statements

TP $$\frac{e^{\mathcal{T}} = (e', \Omega, \mathcal{W})}{(e \to t)^{\mathcal{T}} = (e' \to t,\ \Omega, \mathcal{W})}$$

TA $$\frac{(\ e_i{}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)\ )_{i=1\ldots n}}{(p(\overline{e}_n) == v)^{\mathcal{T}} = (\ p(\overline{e'}_n) == v, \bigcup_{i=1}^n \Omega_i, \bigcup_{i=1}^n \mathcal{W}_i\ )}$$

if $p \in PF^n$, $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$.

TCS $$\frac{\psi^{\mathcal{T}} = (\psi', \Omega, \mathcal{W})}{(\psi\sharp d \Leftarrow \Pi)^{\mathcal{T}} = (\psi' \Leftarrow \Pi, \Omega \cup \{\ulcorner d \trianglelefteq W \urcorner \mid W \in \mathcal{W}\})}$$

if ψ is of the form $e \to t$ or $p(\overline{e}_n) == v$ and $d \in D_{\mathcal{D}}$.

Transforming Program Rules

TPR $$\frac{r^{\mathcal{T}} = (r', \Omega_r, \mathcal{W}_r) \qquad (\ \delta_i{}^{\mathcal{T}} = (\delta'_i, \Omega_i, \mathcal{W}_i)\ )_{i=1\ldots m}}{\begin{array}{l}(f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \delta_1, \ldots, \delta_m)^{\mathcal{T}} = \\ \quad f(\overline{t}_n, W) \to r' \Leftarrow \mathsf{qVal}(W),\ \Omega_r,\ (\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)_{W' \in \mathcal{W}_r}, \\ \qquad (\ \Omega_i,\ (\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)_{W' \in \mathcal{W}_i},\ \delta'_i\ )_{i=1\ldots m}\end{array}}$$

where W is a fresh variable.

Transforming Goals

TG $$\frac{(\ \delta_i{}^{\mathcal{T}} = (\delta'_i, \Omega'_i, \mathcal{W}'_i)\ )_{i=1\ldots m}}{\begin{array}{l}((\ \delta_i\sharp W_i, W_i \trianglerighteq \beta_i\ )_{i=1\ldots m})^{\mathcal{T}} = \\ \quad (\ \Omega'_i,\ \mathsf{qVal}(W_i),\ (\ulcorner W_i \trianglelefteq W' \urcorner)_{W' \in \mathcal{W}'_i},\ \ulcorner W_i \trianglerighteq \beta_i \urcorner,\ \delta'_i\ )_{i=1\ldots m}\end{array}}$$

Fig. 5. Transformation rules

by adding extra qualification constraints to be imposed on W, namely $\mathsf{qVal}(W)$ and $(\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)_{W' \in \mathcal{W}'}$, for $\mathcal{W}' = \mathcal{W}_r$ and $\mathcal{W}' = \mathcal{W}_i$ $(1 \le i \le m)$. By convention, $(\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)_{W' \in \mathcal{W}'}$ is understood as $\ulcorner W \trianglelefteq \alpha \urcorner$ in case that $\mathcal{W}' = \emptyset$. The idea is that W's value cannot exceed the infimum in $\mathcal{D}$ of all the values $\alpha \circ \beta$, for the different β coming from the qualifications of r and δ_i $(1 \le i \le m)$.

Finally, **TG** transforms a goal $(\ \delta_i \sharp W_i,\ W_i \trianglerighteq \beta_i\ )_{i=1\dots m}$ by transforming each atomic constraint δ_i and adding $\mathsf{qVal}(W_i)$, $(\ulcorner W_i \trianglelefteq W' \urcorner)_{W' \in \mathcal{W}'_i}$ and $\ulcorner W_i \trianglerighteq \beta_i \urcorner$ $(1 \le i \le m)$ to ensure that each W_i is interpreted as a qualification value not bigger than the qualification computed for δ_i and satisfying the threshold condition $W_i \trianglerighteq \beta_i$. In case that $\mathcal{W}'_i = \emptyset$, $(\ulcorner W_i \trianglelefteq W' \urcorner)_{W' \in \mathcal{W}'_i}$ is understood as $\ulcorner W_i \trianglelefteq \mathbf{t} \urcorner$.

The result of applying **TPR** to all the program rules of a program $\mathcal{P}$ will be noted as $\mathcal{P}^{\mathcal{T}}$. The following theorem proves that QCRWL($\mathcal{D},\mathcal{C}$)-derivability from $\mathcal{P}$ corresponds to CRWL($\mathcal{C}$)-derivability from $\mathcal{P}^{\mathcal{T}}$. Since program semantics in QCFLP($\mathcal{D},\mathcal{C}$) and in CFLP($\mathcal{C}$) is characterized by, respectively, derivability in QCRWL($\mathcal{D},\mathcal{C}$) and in CRWL($\mathcal{C}$), the program transformation is semantically correct. The theorem uses an auxiliary lemma we are proving first which indicates that the constraints obtained when transforming a qc-statement always admit a solution.

Lemma 5. *Let $\varphi = \psi \sharp d \Leftarrow \Pi$ be a qc-statement such that $\varphi^{\mathcal{T}} = (\psi' \Leftarrow \Pi, \Omega')$. Then exists $\rho : var(\Omega') \to D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ solution of Ω'.*

Proof. $\varphi^{\mathcal{T}}$ is obtained by the transformation rule **TCS** of Figure 5. This rule needs to obtain $\psi^{\mathcal{T}}$ which can be done using either the transformation rule **TP** or **TA** of the same figure. In the case of using **TP**, ψ must be of the form $(e \to t)$ and Ω' will be of the form $\Omega \cup \{\ulcorner d \trianglelefteq W \urcorner \mid W \in \mathcal{W}\}$, with $\Omega, \mathcal{W}$ such that $e^{\mathcal{T}} = (e', \Omega, \mathcal{W})$. Checking the transformation rules for expressions (again Figure 5) we see that Ω is a set of constraints where each element is either of the form $\ulcorner W \trianglelefteq W' \urcorner$ or $\mathsf{qVal}(W)$, with $W, W' \in \mathcal{W}ar$. Then ρ can be defined assigning $\mathbf{t}$ to every variable W occurring in either Ω' or $\mathcal{W}$. The case corresponding to the transformation rule **TA** is analogous. □

Theorem 3. *Let $\mathcal{P}$ be a QCFLP($\mathcal{D},\mathcal{C}$)-program and $\psi \sharp d \Leftarrow \Pi$ a qc-statement such that $(\psi \sharp d \Leftarrow \Pi)^{\mathcal{T}} = (\psi' \Leftarrow \Pi, \Omega')$. Then the two following statements are equivalent:*

1. *$\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \psi \sharp d \Leftarrow \Pi$.*
2. *$\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \psi'\rho \Leftarrow \Pi$ for some $\rho \in Sol_{\mathcal{C}}(\Omega')$ such that $vdom(\rho) = var(\Omega')$.*

Proof. We prove the equivalence separately proving each implication.

[*1.* $\Rightarrow$ *2.*] (*Transformation completeness*). Assume $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \psi \sharp d \Leftarrow \Pi$ by means of a QCRWL($\mathcal{D},\mathcal{C}$) proof tree T with k nodes. By induction on k we show the existence of a CRWL($\mathcal{C}$) proof tree T' witnessing $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \psi'\rho \Leftarrow \Pi$ for some $\rho \in \mathrm{Sol}_{\mathcal{C}}(\Omega')$ such that $\mathrm{vdom}(\rho) = \mathrm{var}(\Omega')$.

Basis (k=1). If T contains only one node the QCRWL($\mathcal{D},\mathcal{C}$) inference step applied at the root must be one of the following:

– **QTI**. In this case $\psi \sharp d \Leftarrow \Pi$ is a trivial qc-statement, and we take ρ as the substitution defined in Lemma 5. By Def. 4, $\psi \sharp d \Leftarrow \Pi$ trivial implies either $\psi = e \rightarrow \perp$ or $\mathrm{Unsat}_{\mathcal{C}}(\Pi)$. In the first case $\psi' = e' \rightarrow \perp$ and therefore $\psi'\rho \Leftarrow \Pi$ is trivial. Analogously, if $\mathrm{Unsat}_{\mathcal{C}}(\Pi)$ then $\psi'\rho \Leftarrow \Pi$ is trivial as well. Hence T' consists of a single node $\psi'\rho \Leftarrow \Pi$ with a **TI** inference step at its root.
– **QRR**. In this case $\psi = t \rightarrow t$ for some $t \in \mathcal{V}ar \cup \mathcal{B}_{\mathcal{C}}$, and $(\psi \sharp d \Leftarrow \Pi)^{\mathcal{T}} = (t \rightarrow t \Leftarrow \Pi, \emptyset)$ (applying the transformation rules **TCS**, **TP** and **TAE** to obtain $t^{\mathcal{T}} = (t, \emptyset, \emptyset)$). Therefore ρ can be defined as the identity substitution and prove $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \psi'\rho \Leftarrow \Pi$ by using a single **RR** inference step.
– **QDC**. In this case $\psi = c \rightarrow c$ and $(\psi \sharp d \Leftarrow \Pi)^{\mathcal{T}} = (c \rightarrow c \Leftarrow \Pi, \emptyset)$ (applying the transformation rules **TCS**, **TP** and **TCE**$_1$ for $c^{\mathcal{T}} = (c, \emptyset, \emptyset)$). Therefore ρ can be defined as the identity substitution and prove $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \psi'\rho \Leftarrow \Pi$ by using a single **DC** inference step.

Inductive step ($k>1$). The QCRWL($\mathcal{D},\mathcal{C}$) inference step applied at the root must be one of the following:

– **QDC**. In this case $\psi = c(\overline{e}_n) \rightarrow c(\overline{t}_n)$ and the first inference step is of the form

$$\frac{(\ (e_i \rightarrow t_i) \sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(c(\overline{e}_n) \rightarrow c(\overline{t}_n)) \sharp d \Leftarrow \Pi}$$

with $d \trianglelefteq d_i$ ($1 \leq i \leq n$). In order to obtain $\psi \sharp d \Leftarrow \Pi^{\mathcal{T}}$ we apply the transformation rules as follows:
- By the transformation rule **TCE**$_1$,
$$c(\overline{e}_n)^{\mathcal{T}} = (c(\overline{e'}_n), \bigcup_{i=1}^{n} \Omega_i, \bigcup_{i=1}^{n} \mathcal{W}_i)$$
with $e_i{}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)$ for $i = 1 \ldots n$.
- By **TP** and with the result of the previous step,
$$\psi^{\mathcal{T}} = (c(\overline{e}_n) \rightarrow c(\overline{t}_n))^{\mathcal{T}} = (c(\overline{e'}_n) \rightarrow c(\overline{t}_n), \bigcup_{i=1}^{n} \Omega_i, \bigcup_{i=1}^{n} \mathcal{W}_i)\ .$$
- And finally from $\psi^{\mathcal{T}}$ and by **TCS**,
$$(\psi \sharp d \Leftarrow \Pi)^{\mathcal{T}} = (c(\overline{e'}_n) \rightarrow c(\overline{t}_n) \Leftarrow \Pi, \Omega')\ ,$$
with
$$\Omega' = \bigcup_{i=1}^{n} \Omega_i \cup \{\ulcorner d \trianglelefteq W \urcorner \mid W \in \bigcup_{i=1}^{n} \mathcal{W}_i\}\ .$$

From the premises $(\ (e_i \rightarrow t_i) \sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}$ of the **QDC** step and by the induction hypothesis we have that $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (e'_i \rightarrow t_i)\rho_i \Leftarrow \Pi$, $i = 1 \ldots n$ for some substitutions $\rho_i : \mathrm{var}(\Omega'_i) \rightarrow D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ solution of

$$\Omega'_i = \Omega_i \cup \{\ulcorner d_i \trianglelefteq W \urcorner \mid W \in \mathcal{W}_i\}$$

for $i = 1 \dots n$. Since $\mathrm{var}(\Omega'_i) \cap \mathrm{var}(\Omega'_j) = \emptyset$ for every $1 \le i, j \le n$, $i \neq j$, and $\mathrm{var}(\Omega') = \bigcup_{i=1}^{n} \mathrm{var}(\Omega'_i)$, we can define a new substitution $\rho : \mathrm{var}(\Omega') \to D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ as $\rho = \biguplus_{i=1}^{n} \rho_i$. It is easy to check that ρ is solution of Ω':

- It is solution of every Ω'_i for $i = 1 \dots n$, since $\rho \upharpoonright \mathrm{var}(\Omega'_i) = \rho_i$. Therefore it is solution of $\bigcup_{i=1}^{n} \Omega_i$.
- It is a solution of $\{\ulcorner d \trianglelefteq W \urcorner \mid W \in \bigcup_{i=1}^{n} \mathcal{W}_i\}$ because as solution of Ω'_i for $i = 1 \dots n$, ρ is solution of $\{\ulcorner d_i \trianglelefteq W \urcorner \mid W \in \mathcal{W}_i\}$, and by the hypothesis of **QDC** $d \trianglelefteq d_i$.

Therefore we prove $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (c(\overline{e'}_n)\rho \to c(\overline{t}_n))\rho \Leftarrow \Pi$ with a proof tree T' which starts with a **DC** inference rule of the form

$$\frac{((\ e'_i \to t_i)\rho \Leftarrow \Pi\ )_{i=1\dots n}}{(c(\overline{e'}_n) \to c(\overline{t}_n))\rho \Leftarrow \Pi} \ .$$

In order to justify that $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (e'_i \to t_i)\rho \Leftarrow \Pi$ for each $i = 1 \dots n$, we observe that the only variables of $e'_i \to t_i$ that can be affected by ρ are those introduced in e'_i by the transformation, and that therefore $(e'_i \to t_i)\rho = (e'_i \to t_i)\rho_i$ for $i = 1 \dots n$, and these premises correspond to the inductive hypotheses of this case.

– $\mathbf{QDF}_{\mathcal{P}}$. In this case $\psi = f(\overline{e}_n) \to t$ and the inference step applied at the root is of the form

$$\frac{(\ (e_i \to t_i\theta)\sharp d_i \Leftarrow \Pi\ )_{i=1\dots n} \quad (r\theta \to t)\sharp d'_0 \Leftarrow \Pi \quad (\ \delta_j\theta\sharp d'_j \Leftarrow \Pi\ )_{j=1\dots m}}{(f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi}$$

for some program rule $R_l = (f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \overline{\delta}_m) \in \mathcal{P}$ and substitution θ such that $R_l\theta \in [\mathcal{P}]_\perp$, and with $d \trianglelefteq d_i$ $(1 \le i \le n)$ and $d \trianglelefteq \alpha \circ d'_j$ $(0 \le j \le m)$.
The inductive hypotheses in this case are:

1. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (e'_i \to t_i\theta)\rho_i \Leftarrow \Pi$ for $i = 1 \dots n$, with ${e_i}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)$ and ρ_i solution of $\Omega'_i = \Omega_i \cup \{\ulcorner d_i \trianglelefteq W' \urcorner \mid W' \in \mathcal{W}_i\}$, for $i = 1 \dots n$.
2. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (r'\theta \to t)\rho'_0 \Leftarrow \Pi$, with $r^{\mathcal{T}} = (r', \Omega_r, \mathcal{W}'_0)$ (it is easy to check that if $r^{\mathcal{T}} = (r', \Omega_r, \mathcal{W}'_0)$ then $(r\theta)^{\mathcal{T}} = (r'\theta, \Omega_r, \mathcal{W}'_0)$ for every substitution θ), and ρ'_0 solution of $\Omega'_r = \Omega_r \cup \{\ulcorner d'_0 \trianglelefteq W' \urcorner \mid W' \in \mathcal{W}'_0\}$.
3. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\delta'_j\theta)\rho'_j \Leftarrow \Pi$ with ${\delta_j}^{\mathcal{T}} = (\delta'_j, \Omega_{\delta_j}, \mathcal{W}'_j)$ for $j = 1 \dots k$ (it is easy to check that if ${\delta_j}^{\mathcal{T}} = (\delta'_j, \Omega_{\delta_j}, \mathcal{W}'_j)$ then $(\delta_j\theta)^{\mathcal{T}} = (\delta'_j\theta, \Omega_{\delta_j}, \mathcal{W}'_j)$ for every substitution θ and $j = 1 \dots k$). The substitution ρ'_j is solution of $\Omega'_{\delta_j} = \Omega_{\delta_j} \cup \{\ulcorner d'_j \trianglelefteq W' \urcorner \mid W' \in \mathcal{W}'_j\}$ for $j = 1 \dots m$.

In this case, $(\psi\sharp d \Leftarrow \Pi)^{\mathcal{T}}$ is obtained by means of the transformation rule **TCS**. This rule asks first for the transformation of the qualified statement $(f(\overline{e}_n) \to t)\sharp d$, which can be obtained by rule **TP**, and this one requires the transformation of $f(\overline{e}_n)$, provided by rule rule $\mathbf{TCE}_2$. Let's see it:

$$\frac{(\ e_i{}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)\ )_{i=1\dots n}}{\begin{array}{l} f(\overline{e}_n)^{\mathcal{T}} = (\ f(\overline{e'}_n, W), \\ \quad (\bigcup_{i=1}^n \Omega_i) \cup \{\mathsf{qVal}(W)\} \cup \\ \quad \{\ulcorner W \trianglelefteq W' \urcorner \mid W' \in \bigcup_{i=1}^n \mathcal{W}_i\},\ \{W\}\ ) \end{array}}\ \mathbf{TCE_2}$$

$$\frac{}{\begin{array}{l} (f(\overline{e}_n) \to t)^{\mathcal{T}} = (\ f(\overline{e'}_n, W) \to t, \\ \quad (\bigcup_{i=1}^n \Omega_i) \cup \{\mathsf{qVal}(W)\} \cup \\ \quad \{\ulcorner W \trianglelefteq W' \urcorner \mid W' \in \bigcup_{i=1}^n \mathcal{W}_i\},\ \{W\}\ ) \end{array}}\ \mathbf{TP}$$

$$\frac{}{\begin{array}{l} ((f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi)^{\mathcal{T}} = (\ f(\overline{e'}_n, W) \to t \Leftarrow \Pi, \\ \quad (\bigcup_{i=1}^n \Omega_i) \cup \{\mathsf{qVal}(W)\} \cup \\ \quad \{\ulcorner W \trianglelefteq W' \urcorner \mid W' \in \bigcup_{i=1}^n \mathcal{W}_i\} \cup \{\ulcorner d \trianglelefteq W \urcorner\}\ ) \end{array}}\ \mathbf{TCS}$$

Therefore

$$\Omega' = (\bigcup_{i=1}^{n} \Omega_i) \cup \{\mathsf{qVal}(W)\} \cup \{\ulcorner W \trianglelefteq W' \urcorner \mid W' \in \bigcup_{i=1}^{n} \mathcal{W}_i\} \cup \{\ulcorner d \trianglelefteq W \urcorner\}\ .$$

We define a new substitution

$$\rho = \biguplus_{i=1}^{n} \rho_i \uplus \rho'_0 \uplus \biguplus_{j=1}^{m} \rho'_j \uplus \{W \mapsto d\}\ .$$

It is straightforward to check that ρ is a solution for Ω' because ρ is solution of:

- Each Ω_i $(1 \leq i \leq n)$, because ρ_i is solution of Ω'_i which contains Ω_i (see inductive hypothesis 1) and ρ is an extension of ρ_i.
- $\{\mathsf{qVal}(W)\}$ because $\mathsf{qVal}(W)\rho = \mathsf{qVal}(d)$ which holds by definition.
- $\{\ulcorner W \trianglelefteq W' \urcorner \mid W' \in \bigcup_{i=1}^n \mathcal{W}_i\}$ because $W\rho = d$, ρ is solution of $\{\ulcorner d_i \trianglelefteq W' \urcorner \mid W' \in \mathcal{W}_i\}$ for each $i = 1 \dots n$ (see inductive hypothesis 1), and $d \trianglelefteq d_i$ $(1 \leq i \leq n)$ by the hypotheses of the inference rule $\mathbf{QDP}_{\mathcal{P}}$.
- $\{\ulcorner d \trianglelefteq W \urcorner\}$ since $W\rho = d$ and trivially $d \trianglelefteq d$.

The transformed of the program rule $R_l = (f(\bar{t}_n) \xrightarrow{\alpha} r \Leftarrow \overline{\delta}_m) \in \mathcal{P}$ will be a program rule in $\mathcal{P}^{\mathcal{T}}$ of the form:

$$\begin{array}{rl} (R_l)^{\mathcal{T}} = (f(\bar{t}_n, W) \to r' \Leftarrow & \mathsf{qVal}(W), \Omega_r, (\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)_{W' \in \mathcal{W}'_0}, \\ & \Omega_{\delta_1}, (\ulcorner W \trianglelefteq \alpha \circ W'_1 \urcorner)_{W'_1 \in \mathcal{W}'_1}, \delta'_1 \\ & \vdots \\ & \Omega_{\delta_m}, (\ulcorner W \trianglelefteq \alpha \circ W'_m \urcorner)_{W'_m \in \mathcal{W}'_m}, \delta'_m \end{array}$$

with $r^{\mathcal{T}} = (r', \Omega_r, \mathcal{W}'_0)$ and $(\ \delta_j{}^{\mathcal{T}} = (\delta'_j, \Omega_{\delta_j}, \mathcal{W}'_j)\ )_{j=1\ldots m}$.
Then we prove $(f(\overline{e'}_n, W) \to t)\rho \Leftarrow \Pi$ in CFLP($\mathcal{C}$) with a $\mathbf{DF}_{\mathcal{P}}$ root inference step using the program rule $(R_l)^{\mathcal{T}}$ and the substitution $\theta' = \theta \uplus \rho$ to instantiate the program rule. We next check that every premise of this inference can be proven in CRWL($\mathcal{C}$):

- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (e'_i\rho \to t_i(\theta \uplus \rho)) \Leftarrow \Pi$ for $i = 1 \ldots n$. We observe that the only variables of e'_i that can be affected by ρ are those in ρ_i. Moreover, ρ cannot affect t_i because the program transformation does not introduce new variables in terms. Therefore $(e'_i\rho \to t_i(\theta \uplus \rho)) = (e'_i \to t_i\theta)\rho_i$ and $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (e'_i \to t_i\theta)\rho_i \Leftarrow \Pi$ for $i = 1 \ldots n$ follows from inductive hypothesis number 1.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (W\rho \to W(\theta \uplus \rho)) \Leftarrow \Pi$. By construction of ρ, $(W\rho \to W(\theta \uplus \rho)) = d \to d$ and one **RR** inference step proves this statement.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (r'(\theta \uplus \rho) \to t\rho) \Leftarrow \Pi$. In this case $t\rho = t$ because t it contains no variables introduced during the transformation, and $r'(\theta \uplus \rho) = r'(\theta\rho'_0)$ since ρ'_0 is the only part of ρ that can affect r' and the range of θ does not include any of the new variables in the domain of ρ'_0. Now, $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (r'\theta \to t)\rho'_0 \Leftarrow \Pi$ follows from inductive hypothesis number 2.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \mathsf{qVal}(W)(\theta \uplus \rho) \Leftarrow \Pi$. W is a fresh variable and, by construction of ρ, $\mathsf{qVal}(W)(\theta \uplus \rho) = \mathsf{qVal}(d)$. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \mathsf{qVal}(d) \Leftarrow \Pi$ trivially holds.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \Omega_r(\theta \uplus \rho) \Leftarrow \Pi$. $\Omega_r(\theta \uplus \rho) = \Omega_r\rho = \Omega_r\rho'_0$ and, by construction, ρ'_0 is solution of Ω_r.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)(\theta \uplus \rho) \Leftarrow \Pi$ for each $W' \in \mathcal{W}'_0$. We have $(\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)(\theta \uplus \rho) = (\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)\rho = \ulcorner W\rho \trianglelefteq \alpha \circ W'\rho'_0 \urcorner = \ulcorner d \trianglelefteq \alpha \circ W'\rho'_0 \urcorner$. And $\ulcorner d \trianglelefteq \alpha \circ W'\rho'_0 \urcorner$ holds because $d \trianglelefteq \alpha \circ d'_0$ by the hypotheses of the inference rule $\mathbf{QDP}_{\mathcal{P}}$, and $\ulcorner d'_0 \trianglelefteq W' \urcorner$ by inductive hypothesis number 2.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \Omega_{\delta_j}(\theta \uplus \rho) \Leftarrow \Pi$ for $j = 1 \ldots m$. As in the previous premises $\Omega_{\delta_j}(\theta \uplus \rho) = \Omega_{\delta_j}\rho = \Omega_{\delta_j}\rho'_j$ and ρ'_j is solution of Ω_{δ_j} as a consequence of the inductive hypothesis number 3.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\ulcorner W \trianglelefteq \alpha \circ W'_j \urcorner)(\theta \uplus \rho) \Leftarrow \Pi$ for every $W'_j \in \mathcal{W}'_j$ and $j = 1 \ldots m$. We have $(\ulcorner W \trianglelefteq \alpha \circ W'_j \urcorner)(\theta \uplus \rho) = (\ulcorner W \trianglelefteq \alpha \circ W'_j \urcorner)\rho = \ulcorner W\rho \trianglelefteq \alpha \circ W'_j\rho \urcorner = \ulcorner d \trianglelefteq \alpha \circ W'_j\rho'_j \urcorner$. Now, from the hypotheses of the inference rule $\mathbf{QDP}_{\mathcal{P}}$ follows $d \trianglelefteq \alpha \circ d'_j$ for $j = 1 \ldots m$, and from inductive hypothesis number 3, ρ'_j is solution of $\ulcorner d'_j \trianglelefteq W'_j \urcorner$. Hence $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \ulcorner d \trianglelefteq \alpha \circ W'_j\rho'_j \urcorner \Leftarrow \Pi$ for $j = 1 \ldots k$.
- $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \delta'_j(\theta \uplus \rho) \Leftarrow \Pi$ for $j = 1 \ldots m$. In this case δ'_j can contain variables from both θ and ρ'_j. Hence $\delta'_j(\theta \uplus \rho) = (\delta'_j\theta)\rho'_j$. And $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\delta'_j\theta)\rho'_j \Leftarrow \Pi$ follows from the inductive hypothesis number 3.

– **QPF**. In this case $\psi = p(\overline{e}_n) \to v$ and the inference step applied at the root is of the form

$$\frac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{e}_n) \to v)\sharp d \Leftarrow \Pi}$$

with $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$, $\Pi \models_{\mathcal{C}} p(\overline{t}_n) \to v$ and $d \trianglelefteq d_i$ $(1 \le i \le n)$. In order to obtain $(\psi\sharp d \Leftarrow \Pi)^{\mathcal{T}}$ one has to:

- First, apply the transformation rule $\mathbf{TCE}_1$,

$$p(\overline{e}_n)^{\mathcal{T}} = (p(\overline{e'}_n), \bigcup_{i=1}^{n} \Omega_i, \bigcup_{i=1}^{n} \mathcal{W}_i)$$

  where ${e_i}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)$ for $i = 1 \ldots n$.
- Second, apply the transformation rule **TP**,

$$(p(\overline{e}_n) \to v)^{\mathcal{T}} = (p(\overline{e'}_n) \to v, \bigcup_{i=1}^{n} \Omega_i, \bigcup_{i=1}^{n} \mathcal{W}_i) \ .$$

- And finally, apply the transformation rule **TCS**,

$$(\psi \sharp d \Leftarrow \Pi)^{\mathcal{T}} = (p(\overline{e'}_n) \to v \Leftarrow \Pi, \bigcup_{i=1}^{n} \Omega_i \cup \{\ulcorner d \trianglelefteq W \urcorner \mid W \in \bigcup_{i=1}^{n} \mathcal{W}_i\}) \ .$$

Therefore

$$\Omega' = \bigcup_{i=1}^{n} \Omega_i \cup \{\ulcorner d \trianglelefteq W \urcorner \mid W \in \bigcup_{i=1}^{n} \mathcal{W}_i\} \ .$$

From the premises $(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}$ of the inference rule **QPF**, and by the inductive hypothesis we have $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (e'_i \to t_i)\rho_i \Leftarrow \Pi$ $(1 \le i \le n)$ for some substitutions $\rho_i : \mathrm{var}(\Omega'_i) \to D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ solution of

$$\Omega'_i = \Omega_i \cup \{\ulcorner d_i \trianglelefteq W \urcorner \mid W \in \mathcal{W}_i\}$$

for $i = 1 \ldots n$. We define a new substitution $\rho : \mathrm{var}(\Omega') \to D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ as $\rho = \biguplus_{i=1}^{n} \rho_i$. It is easy to check that ρ is solution of Ω':
- It is solution of every Ω'_i for $i = 1 \ldots n$, since $\rho\restriction\mathrm{var}(\Omega'_i) = \rho_i$. Therefore it is solution of $\bigcup_{i=1}^{n} \Omega_i$.
- It is a solution of $\{\ulcorner d \trianglelefteq W \urcorner \mid W \in \bigcup_{i=1}^{n} \mathcal{W}_i\}$ because as solution of Ω'_i for $i = 1 \ldots n$, ρ is solution of $\{\ulcorner d_i \trianglelefteq W \urcorner \mid W \in \mathcal{W}_i\}$, and by the hypothesis of the inference rule **QPF**, $d \trianglelefteq d_i$ $(1 \le i \le n)$.

We now prove $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (p(\overline{e'}_n) \to v)\rho \Leftarrow \Pi$ with a proof tree T' with a **PF** root inference of the form:

$$\frac{(\ (e'_i \to t_i)\rho \Leftarrow \Pi\ )_{i=1\ldots n}}{(p(\overline{e'}_n)\rho \to v) \Leftarrow \Pi}$$

The rule can be applied because the requirements $v \in \mathcal{V}ar \cup DC^0 \cup B_{\mathcal{C}}$ and $\Pi \models_{\mathcal{C}} p(\overline{t}_n) \to v$ are ensured by the hypothesis of the inference rule **QPF**. In order to justify that $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (e'_i \to t_i)\rho \Leftarrow \Pi$ for each $i = 1 \ldots n$, we observe that the only variables of $(e'_i \to t_i)$ that can be affected by ρ are those introduced in e'_i by the transformation, and that therefore $(e'_i \to t_i)\rho = (e'_i \to t_i)\rho_i$ for $i = 1 \ldots n$, and it is easy to check that these premises correspond to the inductive hypotheses of this case.

– **QAC**. This case is analogous to the previous proof, with the only differences being:

- The inference rule applied at the root of the proof tree is a **QAC** inference rule instead of a **QPF** inference rule.
- In order to obtain the $(\psi\sharp d \Leftarrow \Pi)^{\mathcal{T}}$, the transformation rules applied are **TA** and **TCS** instead of **TCE**$_1$, **TP** and **TCS**.
- The proof tree T' will have an **AC** inference step at its root instead of a **PF** inference step.

[*2.* $\Rightarrow$ *1.*] *(Transformation soundness).* Assume $\rho \in \mathrm{Sol}_{\mathcal{C}}(\Omega')$ such that $\mathrm{vdom}(\rho) = \mathrm{var}(\Omega')$ and $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \psi'\rho \Leftarrow \Pi$ by means of a CRWL($\mathcal{C}$) proof tree T with k nodes. Reasoning by induction on k we show the existence of a QCRWL($\mathcal{D},\mathcal{C}$) proof tree T' witnessing $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \psi\sharp d \Leftarrow \Pi$.

Basis (k=1). If T contains only one node the QCRWL($\mathcal{D},\mathcal{C}$) inference step applied at the root must be any of the following:

- **TI**. In this case $\psi'\rho \Leftarrow \Pi$ is a trivial c-statement. Then $\psi'\rho$ is either of the form $e' \to \bot$ or $\mathrm{Unsat}_{\mathcal{C}}(\Pi)$. In the first case, since the transformation introduces no new variables at the right-hand side of a production, ψ' is of the form $e'' \to \bot$ with $e' = e''\rho$, and ψ is of the form $e \to \bot$, hence $\psi\sharp d \Leftarrow \Pi$ is trivial. Analogously, if $\mathrm{Unsat}_{\mathcal{C}}(\Pi)$ then $\psi\sharp d \Leftarrow \Pi$ is trivial as well. Therefore T' consists of a single node $\psi\sharp d \Leftarrow \Pi$ with d any value in $D_{\mathcal{D}} \setminus \{\mathbf{b}\}$, with a **QTI** inference step at its root.
- **RR**. In this case $\psi'\rho = v \to v$ with $v \in \mathcal{V}ar \cup B_{\mathcal{C}}$. Then $\psi' = v_1 \to v_2$ for some $v_1, v_2 \in \mathcal{V}ar \cup B_{\mathcal{C}}$ such that $\psi'\rho = v \to v$. Since ψ' cannot contain new variables introduced by the transformation (by the transformation rules), this means $\psi'\rho = \psi'$, and then $\psi' = v \to v$. Therefore $\psi = v \to v$, and T' consists of a single node containing $(v \to v)\sharp d \Leftarrow \Pi$ for any $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ as the conclusion of a **QRR** inference step.
- **DC**. Then $\psi'\rho = c \to c$, which means that ψ' can be either of the form $c \to c$, $X \to c$, or $X \to Y$ with X, Y variables. In every case ψ' does not include new variables introduced by the transformation, and therefore $\psi'\rho = \psi'$, which means that $\psi' = c \to c$ is the only possibility. Therefore $\psi = c \to c$, and T' consists of a single node containing $(c \to c)\sharp d \Leftarrow \Pi$ for some $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ as the conclusion of a **QDC** inference step.

Inductive step (k>1). The CRWL($\mathcal{C}$) inference step applied at the root must be any of the following:

- **DC**. Then $\psi'\rho = c(\overline{e''}_n) \to c(\bar{t}_n)$ where $c \in DC^n$ and $n > 0$, which implies that $\psi = c(\overline{e}_n) \to c(\bar{t}_n)$ for values e_i verifying ${e_i}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)$ for $i = 1 \ldots n$, and $e''_i = e'_i\rho$ for $i = 1 \ldots n$. Then

$$\psi^{\mathcal{T}} = (c(\overline{e}_n) \to c(\bar{t}_n))^{\mathcal{T}} = (c(\overline{e'}_n) \to c(\bar{t}_n), \bigcup_{i=1}^{n} \Omega_i, \bigcup_{i=1}^{n} \mathcal{W}_i)$$

  and thus $\varphi = (c(\overline{e}_n) \to c(\bar{t}_n))\sharp d \Leftarrow \Pi$ for some $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ such that $\varphi^{\mathcal{T}} = (\psi' \Leftarrow \Pi, \Omega')$, with

$$\Omega' = \bigcup_{i=1}^{n} \Omega_i \cup \{\ulcorner d \trianglelefteq W \urcorner \mid W \in \bigcup_{i=1}^{n} \mathcal{W}_i\}$$

The substitution $\rho : \mathrm{var}(\Omega') \to D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ must be solution of Ω', and the inference step at the root must be of the form:

$$\frac{(\ e'_i\rho \to t_i \Leftarrow \Pi\ )_{i=1\ldots n}}{c(\overline{e'}_n)\rho \to c(\overline{t}_n) \Leftarrow \Pi}$$

In the premises we have the proofs T_i of $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} e'_i\rho \Leftarrow \Pi$ for $i = 1\ldots n$. Now, for each $1 \leq i \leq n$ we obtain a new value $d_i \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ as $d_i = \bigsqcap\{W\rho \mid W \in \mathcal{W}_i\}$. Then we will prove $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ applying the following **QDC** inference step at the root:

$$\frac{(\ (e_i \to t_i)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n}}{(c(\overline{e}_n) \to c(\overline{t}_n))\sharp d \Leftarrow \Pi}$$

In order to ensure that this step must be applied we must check that $d \trianglelefteq d_i$ $(1 \leq i \leq n)$. This holds because ρ is solution of Ω', in particular of $\{\ulcorner d \trianglelefteq W \urcorner \mid W \in \mathcal{W}_i\}$ for $i = 1\ldots n$. Therefore for each $i = 1\ldots n$ and $W \in \mathcal{W}_i$, $d \trianglelefteq W\rho$, which means that $d \trianglelefteq d_i = \bigsqcap\{W\rho \mid W \in \mathcal{W}_i\}$. To complete the proof we must check that there are proof trees for the premises, i.e. that $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi_i$ with $\varphi_i = (e_i \to t_i)\sharp d_i \Leftarrow \Pi$, $i = 1\ldots n$. This is a consequence of the inductive hypotheses since for each $i = 1\ldots n$:

- ${\varphi_i}^{\mathcal{T}} = (e'_i \to t_i \Leftarrow \Pi, \Omega'_i)$, with $\Omega'_i = \Omega_i \cup \{\ulcorner d_i \trianglelefteq W \urcorner \mid W \in \mathcal{W}_i\}$.
- ρ is solution of Ω'_i, since it is solution of Ω_i and by the definition of d_i, $d_i \trianglelefteq W\rho$ for every $W \in \mathcal{W}_i$.
- We have that $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} e'_i\rho \Leftarrow \Pi$ for $i = 1\ldots n$ (the premises of the **DC** step).

– $\mathbf{DF}_{\mathcal{P}}$. The inference step at the root of T will use an instance $({R_l}^{\mathcal{T}})\theta \in [\mathcal{P}^{\mathcal{T}}]_{\perp}$ of a program rule ${R_l}^{\mathcal{T}}$ of $\mathcal{P}^{\mathcal{T}}$. ${R_l}^{\mathcal{T}}$ will be the transformed of a program rule $R_l = (f(\overline{t}_n) \xrightarrow{\alpha} r \Leftarrow \overline{\delta}_m) \in \mathcal{P}$, and therefore will have the form:

$$\begin{aligned} {R_l}^{\mathcal{T}} = (f(\overline{t}_n, W) \to r' \Leftarrow\ & \mathsf{qVal}(W), \Omega_r, (\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)_{W' \in \mathcal{W}'_0}, \\ & \Omega_{\delta_1}, (\ulcorner W \trianglelefteq \alpha \circ W'_1 \urcorner)_{W'_1 \in \mathcal{W}'_1}, \delta'_1 \\ & \vdots \\ & \Omega_{\delta_m}, (\ulcorner W \trianglelefteq \alpha \circ W'_m \urcorner)_{W'_m \in \mathcal{W}'_m}, \delta'_m \end{aligned}$$

with $r^{\mathcal{T}} = (r', \Omega_r, \mathcal{W}'_0)$ and $({\delta_j}^{\mathcal{T}} = (\delta'_j, \Omega'_j, \mathcal{W}'_j))_{j=1\ldots m}$.
In this case, $\psi'\rho$ must be of the form $(f(\overline{e'}_{n+1}) \to t)\rho$. By the theorem premises, there exists a qc-statement $\psi\sharp d \Leftarrow \Pi$ such that $(\psi\sharp d \Leftarrow \Pi)^{\mathcal{T}} = (\psi' \Leftarrow \Pi, \Omega')$ for some Ω'. Examining the transformation program rules we observe that the only possibility for ψ is to be of the form $f(\overline{e}_n) \to t$ and that the **TCS** transformation rules should have been applied followed by **TP** and $\mathbf{TCE}_2$. This means in particular that $d \neq \mathbf{b}$ and that ${e_i}^{\mathcal{T}} = (e'_i, \Omega_i, \mathcal{W}_i)$ for $i = 1\ldots n$ and that $e'_{n+1} = V$ with V fresh variable. Hence

$$\begin{aligned} \psi^{\mathcal{T}} = (f(\overline{e'}_n, V) \to t, (\ & \textstyle\bigcup_{i=1}^{n} \Omega_i) \cup \{\mathsf{qVal}(V)\} \cup \\ & \{\ulcorner V \trianglelefteq W' \urcorner \mid W' \in \textstyle\bigcup_{i=1}^{n} \mathcal{W}_i\},\ \{V\}) \end{aligned}$$

and $\varphi = (f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi$ for some $d \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$. By hypotheses, ρ is solution of

$$\Omega' = (\bigcup_{i=1}^{n} \Omega_i) \cup \{\mathsf{qVal}(V)\} \cup \{\ulcorner V \trianglelefteq W' \urcorner \mid W' \in \bigcup_{i=1}^{n} \mathcal{W}_i\} \cup \{\ulcorner d \trianglelefteq V \urcorner\}$$

which means, in particular, that $V\rho \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$, since it must hold both $\mathsf{qVal}(V)$ and $\ulcorner d \trianglelefteq V \urcorner$.
Therefore the root of T will be $f(\overline{e'}_n, V)\rho \to t \Leftarrow \Pi$, with premises proof trees proving:

1. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\ e'_i\rho \to t_i\theta \Leftarrow \Pi\ )_{i=1\ldots n}$.
2. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\ V\rho \to W\theta \Leftarrow \Pi\ )$. Since $V\rho \in D_{\mathcal{D}} \setminus \{\mathbf{b}\}$ then either $W\theta = V\rho$ or $W\theta = \mathbf{b}$. By premise 4 below, $W\theta \neq \mathbf{b}$, therefore $W\theta = V\rho$.
3. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} r'\theta \to t \Leftarrow \Pi$.
4. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \mathsf{qVal}(W\theta) \Leftarrow \Pi$.
5. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \Omega_r\theta \Leftarrow \Pi$.
6. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\ulcorner W \trianglelefteq \alpha \circ W' \urcorner)_{W' \in \mathcal{W}'_0}\theta \Leftarrow \Pi$.
7. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \Omega_{\delta_j}\theta \Leftarrow \Pi$ for $j = 1\ldots m$.
8. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\ulcorner W \trianglelefteq \alpha \circ W'_j \urcorner)_{W'_j \in \mathcal{W}'_j}\theta \Leftarrow \Pi$ for $j = 1\ldots m$.
9. $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} \delta'_j\theta \Leftarrow \Pi$ for $j = 1\ldots m$.

Then we can prove $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ by applying a $\mathbf{QDF}_{\mathcal{P}}$ inference step of the form:

$$\frac{(\ (e_i \to t_i\theta)\sharp d_i \Leftarrow \Pi\ )_{i=1\ldots n} \quad (r\theta \to t)\sharp d'_0 \Leftarrow \Pi \quad (\ \delta_j\theta\sharp d'_j \Leftarrow \Pi\ )_{j=1\ldots m}}{(f(\overline{e}_n) \to t)\sharp d \Leftarrow \Pi}$$

where

- $d_i = \bigsqcap\{W\rho \mid W \in \mathcal{W}_i\}$ for $i = 1\ldots n$.
- $d'_0 = \bigsqcap\{W\theta \mid W \in \mathcal{W}'_0\}$.
- $d'_j = \bigsqcap\{W\theta \mid W \in \mathcal{W}'_j\}$ for $j = 1\ldots m$.

For proving $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \varphi$ we need to check that

- $d \trianglelefteq d_i$ $(1 \leq i \leq n)$. Since ρ is solution of Ω', $d \trianglelefteq W\rho$, and $W\rho \trianglelefteq W'\rho$ for every $W' \in \mathcal{W}_i$ and every $1 \leq i \leq n$. Therefore $d \trianglelefteq \bigsqcap\{\rho(W) \mid W \in \mathcal{W}_i\} = d_i$ for $i = 1\ldots n$.
- $d \trianglelefteq \alpha \circ d'_0$. Since ρ is solution of Ω', $d \trianglelefteq V\rho = W\theta$. From premise 6, $W\theta \trianglelefteq \alpha \circ W'\theta$ for every $W' \in \mathcal{W}'_0$. Therefore $d \trianglelefteq \bigsqcap\{W\theta \mid W \in \mathcal{W}'_0\} = d'_0$.
- $d \trianglelefteq \alpha \circ d'_j$ $(1 \leq j \leq m)$. Analogous to the previous point but using premise 8.

Finally, in order to justify the premises of the $\mathbf{QDF}_{\mathcal{P}}$ we must prove:

- $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} (e_i \to t_i\theta)\sharp d_i \Leftarrow \Pi$, which is a consequence of applying the inductive hypotheses to the premises 1, $(\ e'_i\rho \to t_i\theta \Leftarrow \Pi\ )_{i=1\ldots n}$, following the same reasoning we applied for the premises of the **DC** inference.
- $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} (r\theta \to t)\sharp d'_0 \Leftarrow \Pi$. Analogously, is a consequence of the inductive hypothesis and of premise 3.

  - $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} (\ \delta_j\theta\sharp d'_j \Leftarrow \Pi\ )_{j=1\ldots m}$. Again a consequence of the inductive hypothesis, this time applied to the premise 9.
- **PF**. Analogous to the proof for the **DC** inference step.
- **AC**. analogous to the proof for the **DC** inference step. □

Using Theorem 3 we can prove that the transformation of goals specified in Fig. 5 preserves solutions in the sense of the following result.

Theorem 4. *Let G be a goal for a given QCFLP($\mathcal{D}$,$\mathcal{C}$)-program $\mathcal{P}$. Then, the two following statements are equivalent:*

1. $\langle \sigma, \mu, \Pi\rangle \in Sol_{\mathcal{P}}(G)$.
2. $\langle \sigma \uplus \mu \uplus \rho, \Pi\rangle \in Sol_{\mathcal{P}^{\mathcal{T}}}(G^{\mathcal{T}})$ *for some* $\rho \in Val_{\mathcal{D}}$ *such that* $vdom(\rho)$ *is the set of new variables* W *introduced by the transformation of* G.

Proof. Let $G = (\ \delta_i\sharp W_i, W_i \trianglerighteq \beta_i\ )_{i=1\ldots m}$, σ and μ be given. For $i = 1\ldots m$, consider $\delta_i{}^{\mathcal{T}} = (\delta'_i, \Omega_i, \mathcal{W}_i)$ and $\Omega'_i = \Omega_i \cup \{\ulcorner W_i \trianglelefteq W\urcorner \mid W \in \mathcal{W}_i\}$. According to Fig. 5, $G^{\mathcal{T}} = (\Omega'_i,\ \mathsf{qVal}(W_i),\ \ulcorner W_i \trianglerighteq \beta_i\urcorner,\ \delta'_i)_{i=1\ldots m}$. Then, because of Def. 9(2) and the analogous notion of solution for CFLP($\mathcal{C}$) goals explained in Sect. 3, the two statements of the theorem can be reformulated as follows:

(a) $W_i\mu \trianglerighteq \beta_i$ and $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \delta_i\sigma\sharp W_i\mu \Leftarrow \Pi$ hold for $i = 1\ldots m$.
(b) There exists $\rho \in \mathrm{Val}_{\mathcal{D}}$ with $\mathrm{vdom}(\rho) = \bigcup_{i=1}^{m} \mathrm{var}(\Omega_i)$ such that $\rho \in \mathrm{Sol}_{\mathcal{C}}(\Omega'_i\mu)$, $W_i\mu \trianglerighteq \beta_i$ and $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\delta'_i\sigma)\rho \Leftarrow \Pi$ hold for $i = 1\ldots m$.

[$(a) \Rightarrow (b)$] Assume (a). Note that $\delta_i\sigma\sharp W_i\mu \Leftarrow \Pi^{\mathcal{T}}$ is $\delta'_i\sigma \Leftarrow \Pi, \Omega'_i\mu$. Applying Theorem 3 (with $\psi = \delta_i\sigma$, $d = W_i\mu$ and Π) we obtain $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\delta'_i\sigma)\rho_i \Leftarrow \Pi$ for some $\rho_i \in \mathrm{Sol}_{\mathcal{C}}(\Omega'_i\mu)$ with $\mathrm{vdom}(\rho_i) = \mathrm{var}(\Omega'_i\mu) = \mathrm{var}(\Omega_i)$. Then ($b$) holds for $\rho = \biguplus_{i=1}^{m} \rho_i$.
[$(b) \Rightarrow (a)$] Assume (b). Let $\rho_i = \rho{\upharpoonright}\mathrm{var}(\Omega_i)$, $i = 1\ldots m$. Note that (b) ensures $\mathcal{P}^{\mathcal{T}} \vdash_{\mathcal{C}} (\delta'_i\sigma)\rho_i \Leftarrow \Pi$ and $\rho \in \mathrm{Sol}_{\mathcal{C}}(\Omega'_i\mu)$. Then Theorem 3 can be applied (again with $\psi = \delta_i\sigma$, $d = W_i\mu$ and Π) to obtain $\mathcal{P} \vdash_{\mathcal{D},\mathcal{C}} \delta_i\sigma\sharp W_i\mu \Leftarrow \Pi$. Therefore, ($a$) holds. □

As an example of goal solving via the transformation, we consider again the *library program* $\mathcal{P}$ and the goal G discussed in the Introduction. Both belong to the instance QCFLP($\mathcal{U}$, $\mathcal{R}$) of our scheme. Their translation into CFLP($\mathcal{R}$) can be executed in the $\mathcal{TOY}$ system [3] after loading the Real Domain Constraints library (`cflpr`). The source and translated code are publicly available at `gpd.sip.ucm.es/cromdia/qlp`. Solving the transformed goal in $\mathcal{TOY}$ computes the answer announced in the Introduction as follows:

```
Toy(R)> qVal([W]), W>=0.65, search("German","Essay",intermediate,W) == R
      { R -> 4 }
      { W=<0.7, W>=0.65 }
sol.1, more solutions (y/n/d/a) [y]? no
```

The best qualification value for `W` provided by the answer constraints is `0.7`.

5 Conclusions

The work in this report is based on the scheme CFLP($\mathcal{C}$) for functional logic programming with constraints presented in [13]. Our main results are: a new programming scheme QCFLP($\mathcal{D},\mathcal{C}$) extending the first-order fragment of CFLP($\mathcal{C}$) with qualified computation capabilities; a rewriting logic QCRWL($\mathcal{D},\mathcal{C}$) characterizing QCFLP($\mathcal{D},\mathcal{C}$)-program semantics; and a transformation of QCFLP($\mathcal{D},\mathcal{C}$) into CFLP($\mathcal{C}$) preserving program semantics and goal solutions, that can be used as a correct implementation technique. Existing CFLP($\mathcal{C}$) systems such as $\mathcal{TOY}$ [3] and Curry [9] that use definitional trees as an efficient implementation tool can easily adopt the implementation, since the structure of definitional trees is quite obviously preserved by the transformation.

As argued in the Introduction, our scheme is more expressive than the main related approaches we are aware of. By means of an example dealing with a simplified library, we have shown that instances of QCFLP($\mathcal{D},\mathcal{C}$) can serve as a declarative language for flexible information retrieval problems, where qualified (rather than exact) answers to user's queries can be helpful.

As future work we plan to extend QCFLP($\mathcal{D},\mathcal{C}$) and the program transformation in order to provide explicit support for similarity-based reasoning, as well as the higher-order programming features available in CFLP($\mathcal{C}$). We also plan to automate the program transformation, which should be embedded as part of an enhanced version of the $\mathcal{TOY}$ system. Finally, we plan further research on flexible information retrieval applications, using different instances of our scheme.

References

1. S. Antoy, R. Echahed, and M. Hanus. A needed narrowing strategy. *Journal of the ACM*, 47(4):776–822, 2000.
2. K. R. Apt. Logic programming. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, pages 493–574. Elsevier and The MIT Press, 1990.
3. P. Arenas, A. J. Fernández, A. Gil, F. J. López-Fraguas, M. Rodríguez-Artalejo, and F. Sáenz-Pérez. $\mathcal{TOY}$, a multiparadigm declarative language. version 2.3.1, 2007. R. Caballero and J. Sánchez (Eds.), Available at `http://toy.sourceforge.net`.
4. R. Caballero, M. Rodríguez-Artalejo, and C. A. Romero-Díaz. Similarity-based reasoning in qualified logic programming. In *PPDP '08: Proceedings of the 10th international ACM SIGPLAN conference on Principles and Practice of Declarative Programming*, pages 185–194, New York, NY, USA, 2008. ACM.
5. R. del Vado-Vírseda. Declarative constraint programming with definitional trees. In B. Gramlich, editor, *Proceedings of the 5th International Conference on Frontiers of Combining Systems (FroCoS'05)*, volume 3717 of *LNCS*, pages 184–199. Springer Verlag, 2005.
6. M. Gabbrielli, G. M. Dore, and G. Levi. Observable semantics for constraint logic programs. *Journal of Logic and Computation*, 5(2):133–171, 1995.

7. M. Gabbrielli and G. Levi. Modeling answer constraints in constraint logic programs. In *Proceedings of the 8th International Conference on Logic Programming (ICLP'91)*, pages 238–252. The MIT Press, 1991.
8. S. Guadarrama, S. Muñoz, and C. Vaucheret. Fuzzy prolog: A new approach using soft constraint propagation. *Fuzzy Sets and Systems*, 144(1):127–150, 2004.
9. M. Hanus. Curry: an integrated functional logic language, version 0.8.2, 2006. M. Hanus (Ed.), Available at `http://www.informatik.uni-kiel.de/~curry/report.html`.
10. J. Jaffar, M. Maher, K. Marriott, and P. J. Stuckey. Semantics of constraints logic programs. *Journal of Logic Programming*, 37(1-3):1–46, 1998.
11. J. W. Lloyd. *Foundations of Logic Programming, Second Edition.* Springer, 1987.
12. F. J. López-Fraguas, M. Rodríguez-Artalejo, and R. del Vado-Virseda. A lazy narrowing calculus for declarative constraint programming. In *Proceedings of the 6th International ACM SIGPLAN Conference on Principles and Practice of Declarative Programming (PPDP'04)*, pages 43–54. ACM Press, 2004.
13. F. J. López-Fraguas, M. Rodríguez-Artalejo, and R. del Vado-Vírseda. A new generic scheme for functional logic programming with constraints. *Journal of Higher-Order and Symbolic Computation*, 20(1&2):73–122, 2007.
14. G. Moreno and V. Pascual. Formal properties of needed narrowing with similarity relations. *Electronic Notes in Theoretical Computer Science*, 188:21–35, 2007.
15. S. Riezler. Quantitative constraint logic programming for weighted grammar applications. In C. Retoré, editor, *Proceedings of the Logical Aspects of Computational Linguistics (LACL'96)*, volume 1328 of *LNCS*, pages 346–365. Springer Verlag, 1996.
16. S. Riezler. *Probabilistic Constraint Logic Programming.* PhD thesis, Neuphilologischen Fakultät del Universität Tübingen, 1998.
17. M. Rodríguez-Artalejo. Functional and constraint logic programming. In C. M. H. Comon and R. Treinen, editors, *Constraints in Computational Logics, Theory and Applications*, volume 2002 of *Lecture Notes in Computer Science*, pages 202–270. Springer Verlag, 2001.
18. M. Rodríguez-Artalejo and C. A. Romero-Díaz. A generic scheme for qualified logic programming. Technical Report SIC-1-08, Universidad Complutense, Departamento de Sistemas Informáticos y Computación, Madrid, Spain, 2008.
19. M. Rodríguez-Artalejo and C. A. Romero-Díaz. Quantitative logic programming revisited. In J. Garrigue and M. Hermenegildo, editors, *Functional and Logic Programming (FLOPS'08)*, volume 4989 of *LNCS*, pages 272–288. Springer Verlag, 2008.
20. M. I. Sessa. Approximate reasoning by similarity-based SLD resolution. *Theoretical Computer Science*, 275(1-2):389–426, 2002.
21. V. S. Subrahmanian. Uncertainty in logic programming: Some recollections. *Association for Logic Programming Newsletter*, 20(2), 2007.
22. A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285–309, 1955.