In Logical Aspects of Computational Linguistics, 1996, Nancy, France.

Quantitative Constraint Logic Programming for Weighted Grammar Applications*

Stefan Riezler

Graduiertenkolleg ILS, Seminar für Sprachwissenschaft,
Universität Tübingen, Wilhelmstr. 113, 72074 Tübingen, Germany.
riezler@sfs.nphil.uni-tuebingen.de

Abstract. Constraint logic grammars provide a powerful formalism for expressing complex logical descriptions of natural language phenomena in exact terms. Describing some of these phenomena may, however, require some form of graded distinctions which are not provided by such grammars. Recent approaches to weighted constraint logic grammars attempt to address this issue by adding numerical calculation schemata to the deduction scheme of the underlying CLP framework.
Currently, these extralogical extensions are not related to the model-theoretic counterpart of the operational semantics of CLP, i.e., they do not come with a formal semantics at all.
The aim of this paper is to present a clear formal semantics for weighted constraint logic grammars, which abstracts away from specific interpretations of weights, but nevertheless gives insights into the parsing problem for such weighted grammars. Building on the formalization of constraint logic grammars in the CLP scheme of [11], this formal semantics will be given by a quantitative version of CLP. Such a quantitative CLP scheme can also be valuable for CLP tasks independent of grammars.

1 Introduction

Constraint logic grammars (CLGs) provide a powerful formalism for complex logical description and efficient processing of natural language phenomena. Linguistic description and computational practice may, however, often require some form of graded distinctions which are not provided by such grammars.

One such issue is the task of ambiguity resolution. This problem can be illustrated for formal grammars describing a nontrivial domain of natural language as follows: For such grammars every input of reasonable length may receive a large number of different analyses, many of which are not in accord with human perceptions. Clearly there is a need to distinguish more plausible analyses of an input from less plausible or even totally spurious ones.

* I am greatly indebted to Steven Abney, Thilo Götz and Paul King for their valuable comments on this paper. Furthermore, I would like to thank Graham Katz, Frank Morawietz and two anonymous LACL referees for their helpful suggestions.

This problem has successfully been addressed by the use of weighted grammars for disambiguation in regular and context-free grammars. Weighted grammars assign numerical values, or weights, to the structure-building components of the grammars and calculate the weight of an analysis from the weights of the structural features that make it up. The correct analysis is chosen from among the in-principle possible analyses by assuming the analysis with the greatest weight to be the correct analysis. This approach also allows parsing to be speeded up by pruning low-weighted subanalyses.

The idea of weighted grammars recently has been transferred to highly expressive weighted CLGs by [8, 9] and [7]. The approaches of Erbach and Eisele are based on the feature-based constraint formalism CUF ([5, 4]), which can be seen as an instance of the constraint logic programming (CLP) scheme of [11]. These approaches extend the underlying formalism by assigning weights to program clauses, but differ with respect to an interpretation of weights in a preference-based versus probabilistic framework. Erbach calculates a preference value of analyses from the preference values of the clauses used in the analyses, whereas Eisele assigns application probabilities to clauses from which a probability distribution over analyses is calculated.

There is an obvious problem with these approaches, however. Even if the formal foundation of the underlying framework is clear enough, there is no well-defined semantics for the weighted extensions. This means that these extralogical extensions of the deduction scheme of the underlying constraint logic program are not related to the model-theoretic counterpart of this operational semantics, i.e., they do not come with a formal semantics at all. This is clearly an undesirable state of affairs. Rather, in the same way as CLGs allow for a clear model-theoretic characterization of linguistic objects coupled with the operational parsing system, one would prefer to base a quantitative deduction system on a clear quantitative model-theory in a sound and complete way.

The aim of this paper is to present a clear formal semantics for weighted CLGs, which abstracts away from specific interpretations of weights, but gives insight into the parsing problem for weighted CLGs. Building on the formalization of CLGs in the CLP scheme of [11], this formal semantics will be given by a quantitative version of CLP. Such a quantitative CLP scheme can also be valuable for CLP tasks independent of grammars.

Previous work on related topics has been confined to quantitative extensions of conventional logic programming. A quantitative deduction scheme based on a fixpoint semantics for sets of numerically annotated conventional definite clauses was first presented by van Emden in [26]. In this approach numerical weights are associated with definite clauses as a whole. The semantics of such quantitative rule sets is based upon concepts of fuzzy set algebra and crucially deals with the truth-functional "propagation" of weights across definite clauses. Van Emden's approach initialized research into a now extensively explored area of quantitative logic programming. For example, annotated logic programming as introduced by [25] extends the expressive power of quantitative rule sets by allowing variables and evaluable function terms as annotations. Such annotations can be attached to components of the language formula and come with more complex mappings as a foundation for a multivalued logical semantics. Such extended theories are interpreted in frameworks of lattice-based logics for generalized annotated logic programming ([15]), possibilistic logic for possibilistic logic programming ([6]) or logics of subjective probability for probabilistic logic programming ([20, 21]) and probabilistic deductive databases ([17, 18]).

Aiming at a formal foundation of weighted CLGs in a framework of quantitative CLP, we can start from the ideas developed in the simple and elegant framework of [26], but transfer them to the general CLP scheme of [11]. This means that the form of weighted CLGs under consideration allows us to restrict our attention to numerical weights associated with CLP clauses as a whole. Furthermore, the simple concepts of fuzzy set algebra can also provide a basis for an intuitive formal semantics for quantitative CLP. Such a formal semantics will be sufficiently general in that it is itself not restricted by a specific interpretation of weights. Further extensions should be straightforward, but have to be deferred to future work. Our scheme will straightforwardly transfer the nice properties of the CLP scheme of [11] into a quantitative version of CLP.

2 Constraint Logic Programming and Constraint Logic Grammars

Before discussing the details of our quantitative extension of CLP, some words on the underlying CLP scheme and grammars formulated by these means are necessary. In the following we will rely on the CLP scheme of [11], which generalizes conventional logic programming (see [19]) and also the CLP scheme of [12] to a scheme of definite clause specifications over arbitrary constraint languages. A very general characterization of the concept of constraint language can be given as follows.

Definition 1 **($\mathcal{L}$).** *A constraint language $\mathcal{L}$ consists of*

1. *an $\mathcal{L}$-signature, specifying the non-logical elements of the alphabet of the language,*
2. *a decidable infinite set VAR whose elements are called variables,*
3. *a decidable set CON of $\mathcal{L}$-constraints which are pieces of syntax with unknown internal structure,*
4. *a computable function V assigning to every constraint $\phi \in \mathsf{CON}$ a finite set $\mathsf{V}(\phi)$ of variables, the variables constrained by ϕ,*
5. *a nonempty set of $\mathcal{L}$-interpretations INT, where each $\mathcal{L}$-interpretation $\mathcal{I} \in \mathsf{INT}$ is defined w.r.t. a nonempty set $\mathcal{D}$, the domain of $\mathcal{I}$, and a set ASS of variable assignments $\mathsf{VAR} \to \mathcal{D}$,*
6. *a function $[\![\cdot]\!]^{\mathcal{I}}$ mapping every constraint $\phi \in \mathsf{CON}$ to a set $[\![\phi]\!]^{\mathcal{I}}$ of variable assignments, the solutions of ϕ in $\mathcal{I}$.*
7. *Furthermore, a constraint ϕ constrains only the variables in $\mathsf{V}(\phi)$, i.e., if $\alpha \in [\![\phi]\!]^{\mathcal{I}}$ and β is a variable assignment that agrees with α on $\mathsf{V}(\phi)$, then $\beta \in [\![\phi]\!]^{\mathcal{I}}$.*

To obtain constraint logic programs, a given constraint language $\mathcal{L}$ has to be extended to a constraint language $\mathcal{R}(\mathcal{L})$ providing for the necessary relational atoms and propositional connectives.

Definition 2 ($\mathcal{R}(\mathcal{L})$). *A constraint language $\mathcal{R}(\mathcal{L})$ extending a constraint language $\mathcal{L}$ is defined as follows:*

1. *The signature of $\mathcal{R}(\mathcal{L})$ is an extension of the signature of $\mathcal{L}$ with a decidable set $\mathcal{R}$ of relation symbols and an arity function $\mathsf{Ar} : \mathcal{R} \to \mathbb{N}$.*
2. *The variables of $\mathcal{R}(\mathcal{L})$ are the variables of $\mathcal{L}$.*
3. *The set of $\mathcal{R}(\mathcal{L})$-constraints is the smallest set s.t.*
   - *ϕ is an $\mathcal{R}(\mathcal{L})$-constraint if ϕ is an $\mathcal{L}$-constraint,*
   - *$r(\boldsymbol{x})$ is an $\mathcal{R}(\mathcal{L})$-constraint, called an* **atom**, *if $r \in \mathcal{R}$ is a relation symbol with arity n and $\boldsymbol{x}$ is an n-tuple of pairwise distinct variables,*
   - *$\emptyset$, $F \,\&\, G$, $F \to G$ are $\mathcal{R}(\mathcal{L})$-constraints, if F and G are $\mathcal{R}(\mathcal{L})$-constraints,*
   - *$\phi \,\&\, B_1 \,\&\, \dots \,\&\, B_n \to A$ is an $\mathcal{R}(\mathcal{L})$-constraint, called a* **definite clause**, *if $A, B_1, \dots, B_n$ are atoms and ϕ is an $\mathcal{L}$-constraint. We may write a definite clause also as $A \leftarrow \phi \,\&\, B_1 \,\&\, \dots \,\&\, B_n$.*
4. *The variables constrained by an $\mathcal{R}(\mathcal{L})$-constraint are defined as follows: If ϕ is an $\mathcal{L}$-constraint, then $\mathsf{V}(\phi)$ is defined as in $\mathcal{L}$; $\mathsf{V}(r(x_1, \dots, x_n)) := \{x_1, \dots, x_n\}$; $\mathsf{V}(\emptyset) := \emptyset$; $\mathsf{V}(F \,\&\, G) := \mathsf{V}(F) \cup \mathsf{V}(G)$; $\mathsf{V}(F \to G) := \mathsf{V}(F) \cup \mathsf{V}(G)$.*
5. *For each $\mathcal{L}$-interpretation $\mathcal{I}$, an $\mathcal{R}(\mathcal{L})$-interpretation $\mathcal{A}$ is an extension of an $\mathcal{L}$-interpretation $\mathcal{I}$ with relations $r^{\mathcal{A}}$ on the domain $\mathcal{D}$ of $\mathcal{A}$ with appropriate arity for every $r \in \mathcal{R}$ and the domain of $\mathcal{A}$ is the domain of $\mathcal{I}$.*
6. *For each $\mathcal{R}(\mathcal{L})$-interpretation $\mathcal{A}$, for each $\mathcal{L}$-interpretation $\mathcal{I}$, $[\![\cdot]\!]^{\mathcal{A}}$ is a function mapping every $\mathcal{R}(\mathcal{L})$-constraint to a set of variable assignments s.t.*
   - *$[\![\phi]\!]^{\mathcal{A}} = [\![\phi]\!]^{\mathcal{I}}$ if ϕ is an $\mathcal{L}$-constraint,*
   - *$[\![r(\boldsymbol{x})]\!]^{\mathcal{A}} = \{\alpha \in \mathsf{ASS} \mid \alpha(\boldsymbol{x}) \in r^{\mathcal{A}}\}$,*
   - *$[\![\emptyset]\!]^{\mathcal{A}} = \mathsf{ASS}$,*
   - *$[\![F \,\&\, G]\!]^{\mathcal{A}} = [\![F]\!]^{\mathcal{A}} \cap [\![G]\!]^{\mathcal{A}}$,*
   - *$[\![F \to G]\!]^{\mathcal{A}} = (\mathsf{ASS} \setminus [\![F]\!]^{\mathcal{A}}) \cup [\![G]\!]^{\mathcal{A}}$.*

A constraint logic program then is defined as a definite clause specification over a constraint language.

Definition 3 (definite clause specification). *A definite clause specification $\mathcal{P}$ over a constraint language $\mathcal{L}$ is a set of definite clauses from a constraint language $\mathcal{R}(\mathcal{L})$ extending $\mathcal{L}$.*

Relying on terminology well-known for conventional logic programming, Höhfeld and Smolka's generalization of the key result of conventional logic programming can be stated as follows:[1] First, for every definite clause specification $\mathcal{P}$

[1] Further conditions for this generalization to hold are decidability of the satisfiability problem, closure under variable renaming and closure under intersection for the constraint languages under consideration.

in the extension of an arbitrary constraint language $\mathcal{L}$, every interpretation of $\mathcal{L}$ can be extended to a minimal model of $\mathcal{P}$. Second, the SLD-resolution method for conventional logic programming can be generalized to a sound and complete operational semantics for definite clause specifications not restricted to Horn theories. In contrast to [12], in this scheme constraint languages are not required to be sublanguages of first order predicate logic and do not have to be interpreted in a single fixed domain. This makes this scheme usable for a wider range of applications. Instead, a constraint is satisfiable if there is at least one interpretation in which it has a solution. Moreover, such interpretations do not have to be solution compact. This was necessary in [12] to provide a sound and complete treatment of negation as failure, which is not addressed in [11].

The term constraint logic grammars expresses the connection between CLP and constraint-based grammars. Constraint-based grammars allow for a clear model-theoretic characterization of linguistic objects by stating grammars as sets of axioms of suitable logical languages. However, such approaches do not necessarily provide an operational interpretation of their purely declarative specifications. This may lead to problems with an operational treatment of declaratively well-defined problems such as parsing. CLP provides one possible approach to an operational treatment of various such declarative frameworks by an embedding of arbitrary logical languages into constraint logic programs. CLGs thus are grammars formulated by means of a suitable logical language which can be used as a constraint language in the sense of [11].[2]

For example, for feature based grammars such as HPSG ([23]), a quite direct embedding of a logical language close to that of [24] into the CLP scheme of [11] is done in the formalism CUF ([5,4]). This approach directly offers the operational properties of the CLP scheme by simply redefining grammars as constraint logic programs, but is questionable in losing the connection to the model-theoretic specifications of the underlying feature-based grammars. A different approach is given by [10] where a compilation of a logical language close to that of [16] into constraint logic programs is defined. This translation procedure preserves important model-theoretic properties by generating a constraint logic program $\mathcal{P}(\mathcal{G})$ from a feature-based grammar $\mathcal{G}$ in an explicit way.

The parsing/generation problem for CLGs then is as follows. Given a program $\mathcal{P}$ (encoding a grammar) and a definite goal G (encoding the string/logical form we want to parse/generate from), we ask if we can infer an answer φ of G (which is a satisfiable $\mathcal{L}$ -constraint encoding an analysis) proving the implication $\varphi \rightarrow G$ to be a logical consequence of $\mathcal{P}$.

[2] Clearly, a direct definition of an operational semantics for specific constraint-based grammars is possible and may even better suit the particular frameworks. However, such approaches have to rely directly on the syntactic properties of the logical languages in question. Under the CLP approach, arbitrary constraint-based grammars can receive a unique operational semantics by an embedding into definite clause specifications. The main advantage of this approach is the possibility to put constraint-based grammar processing into the well-understood paradigm of logic programming. This allows the resulting programs to run on existing architectures and to use well-known optimization techniques worked out in this area.

3 Quantitative Constraint Logic Programming

3.1 Syntax and Declarative Semantics of Quantitative Definite Clause Specifications

Building upon the definitions in [11], we can define the syntax of a quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ very quickly. A definite clause specification $\mathcal{P}$ in $\mathcal{R}(\mathcal{L})$ can be extended to a quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ in $\mathcal{R}(\mathcal{L})$ simply by adding numerical factors to program clauses.

The following definitions are made with respect to implicit constraint languages $\mathcal{L}$ and $\mathcal{R}(\mathcal{L})$.

Definition 4 **($\mathcal{P}_{\mathrm{F}}$).** *A quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ in $\mathcal{R}(\mathcal{L})$ is a finite set of quantitative formulae, called quantitative definite clauses, of the form: ϕ & B_1 & ... & B_n $_f\!\rightarrow A$, where $A, B_1, \ldots, B_n$ are $\mathcal{R}(\mathcal{L})$ -atoms, ϕ is an $\mathcal{L}$ -constraint, $n \geq 0$, $f \in (0, 1]$. We may write a quantitative formula also as $A \leftarrow_f \phi$ & B_1 & ... & B_n.*

Such factors should be thought of as abstract weights which receive a concrete interpretation in specific instantiations of $\mathcal{P}_{\mathrm{F}}$ by weighted CLGs.

In the following the notation $\mathcal{R}(\mathcal{L})$ will be used more generally to denote relationally extended constraint languages which possibly include quantitative formulae of the above form.

To obtain a formal semantics for $\mathcal{P}_{\mathrm{F}}$, first we have to introduce an appropriate quantitative measure into the set-theoretic specification of $\mathcal{R}(\mathcal{L})$ -interpretations. One possibility to obtain quantitative $\mathcal{R}(\mathcal{L})$ -interpretations is to base the set algebra of $\mathcal{R}(\mathcal{L})$ -interpretations on the simple and well-defined concepts of fuzzy set algebra (see [27]).

Relying on Höhfeld and Smolka's specification of base equivalent $\mathcal{R}(\mathcal{L})$ -interpretations, i.e., $\mathcal{R}(\mathcal{L})$ -interpretations extending the same $\mathcal{L}$ -interpretation, in terms of the denotations of the relation symbols in these interpretations, we can "fuzzify" such interpretations by regarding the denotations of their relation symbols as fuzzy subsets of the set of tuples in the common domain.

Given constraint languages $\mathcal{L}$ and $\mathcal{R}(\mathcal{L})$, we interpret each n-ary relation symbol $r \in \mathcal{R}$ as a fuzzy subset of $\mathcal{D}^n$, for each $\mathcal{R}(\mathcal{L})$ -interpretation $\mathcal{A}$ with domain $\mathcal{D}$. That is, we identify the denotation of r under $\mathcal{A}$ with a total function: $\mu(\_\,; r^{\mathcal{A}}) : \mathcal{D}^n \rightarrow [0, 1]$, which can be thought of as an abstract membership function. Classical set membership is coded in this context by membership functions taking only 0 and 1 as values.

Next, we have to give a model-theoretic characterization of quantitative definite clauses. Clearly, any monotonous mapping could be used for the model-theoretic specification of the interaction of weights in quantitative definite clauses and accordingly for the calculation of weights in the proof-theory of quantitative CLP. For concreteness, we will instantiate such a mapping to the specific case of Definition 5 resembling [26]'s mode of rule application. This will allow us to state the proof-theory of quantitative CLP in terms of min/max trees which in turn enables strategies such as alpha/beta pruning to be used for efficient searching. However, this choice is not crucial for the substantial claims of this paper and generalizations of this particular combination mode to specific applications should be straightforward, but are beyond the scope of this paper.

The following definition of model corresponds to the definition of model in classical logic when considering only clauses with $f = 1$ and mappings $\mathcal{D}^n \rightarrow \{0, 1\}$.

Definition 5 (model). *An $\mathcal{R}(\mathcal{L})$ -interpretation $\mathcal{A}$ extending some $\mathcal{L}$ -interpretation $\mathcal{I}$ is a model of a quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ iff for each $\alpha \in \mathsf{ASS}$, for each quantitative formula $r(\boldsymbol{x}) \leftarrow_f \phi$ & $q_1(\boldsymbol{x}_1)$ & ... & $q_k(\boldsymbol{x}_k)$ in $\mathcal{P}_{\mathrm{F}}$: If $\alpha \in [\![\phi]\!]^{\mathcal{I}}$, then $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \geq f \times min\{\mu(\alpha(\boldsymbol{x}_j); q_j^{\mathcal{A}}) |\ 1 \leq j \leq k\}$.*

Note that the notation of an $\mathcal{R}(\mathcal{L})$ -interpretation $\mathcal{A}$ will be used more generally to include interpretations of quantitative formulae. $\mathcal{R}(\mathcal{L})$ -solutions of a quantitative formula are defined as $[\![r(\boldsymbol{x}) \leftarrow_f \phi \,\&\, q_1(\boldsymbol{x}_1) \,\&\, \ldots \,\&\, q_k(\boldsymbol{x}_k)]\!]^{\mathcal{A}} = \{\alpha \in \mathsf{ASS} \mid \text{If } \alpha \in [\![\phi]\!]^{\mathcal{I}}, \text{ then } \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \geq f \times min\{\mu(\alpha(\boldsymbol{x}_j); q_j^{\mathcal{A}}) |\ 1 \leq j \leq k\}\}$.

The concept of logical consequence is defined as usual.

Definition 6 (logical consequence). *A quantitative formula $r(\boldsymbol{x}) \leftarrow_f \phi$ is a logical consequence of a quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ iff for each $\mathcal{R}(\mathcal{L})$ -interpretation $\mathcal{A}$, $\mathcal{A}$ is a model of $\mathcal{P}_{\mathrm{F}}$ implies that $\mathcal{A}$ is a model of $\{r(\boldsymbol{x}) \leftarrow_f \phi\}$.*

Furthermore, we have that $r(\boldsymbol{x}) \leftarrow_f \phi$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$ implies that $r(\boldsymbol{x}) \leftarrow_{f'} \phi$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$ for every $f' \leq f$.

A **goal** G is defined to be a (possibly empty) conjunction of $\mathcal{R}(\mathcal{L})$ -atoms and $\mathcal{L}$ -constraints. We can, without loss of generality, restrict goals to be of the form $r(\boldsymbol{x})$ & ϕ, i.e., a (possibly empty) conjunction of a single relational atom $r(\boldsymbol{x})$ and an $\mathcal{L}$ -constraint ϕ. This is possible as for each goal $G = r_1(\boldsymbol{x}_1)$ & ... & $r_k(\boldsymbol{x}_k)$ & ϕ containing more than one relational atom, we can complete the program with a new clause $C = r(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k) \leftarrow_1 r_1(\boldsymbol{x}_1)$ & ... & $r_k(\boldsymbol{x}_k)$ & ϕ with G as antecedent and a new predicate with all variables in G as arguments as consequent. Submitting the new predicate $r(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k)$ as query yields the same results as would be obtained when querying with the compound goal G.

Given some $\mathcal{P}_{\mathrm{F}}$ and some goal G, a $\mathcal{P}_{\mathrm{F}}$ **-answer** φ of G is defined to be a satisfiable $\mathcal{L}$ -constraint φ s.t. $\varphi\ {}_f\!\rightarrow G$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$. A quantitative formula $\varphi\ {}_f\!\rightarrow r(\boldsymbol{x})$ & ϕ is defined to be a logical consequence of $\mathcal{P}_{\mathrm{F}}$ iff every model of $\mathcal{P}_{\mathrm{F}}$ is a model of $\{\varphi\ {}_f\!\rightarrow r(\boldsymbol{x}) \,\&\, \phi\}$. An $\mathcal{R}(\mathcal{L})$ -interpretation $\mathcal{A}$ is defined to be a a model of $\{\varphi\ {}_f\!\rightarrow r(\boldsymbol{x}) \,\&\, \phi\}$ iff $[\![\varphi]\!]^{\mathcal{A}} \subseteq [\![\phi]\!]^{\mathcal{A}}$ and $\mathcal{A}$ is a model of $\{r(\boldsymbol{x}) \leftarrow_f \varphi\}$.

Aiming to generalize the key result in the declarative semantics of CLP—the minimal model semantics of definite clause specifications over arbitrary constraint languages—to our quantitative CLP scheme, first we have to associate a complete lattice of interpretations with quantitative definite clause specifications.

Adopting Zadeh's definitions for set operations, we can define a partial ordering on the set of base equivalent $\mathcal{R}(\mathcal{L})$ -interpretations. This is done by defining set operations on these interpretations with reference to set operations on the denotations of relation symbols in these interpretations. We get for all base equivalent $\mathcal{R}(\mathcal{L})$ -interpretations $\mathcal{A}$, $\mathcal{A}'$:

- $\mathcal{A} \subseteq \mathcal{A}'$ iff for each n-ary relation symbol $r \in \mathcal{R}$, for each $\alpha \in \mathsf{ASS}$, for each $\boldsymbol{x} \in \mathsf{VAR}^n$: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}'})$,
- $\mathcal{A} = \bigcup X$ iff for each n-ary relation symbol $r \in \mathcal{R}$, for each $\alpha \in \mathsf{ASS}$, for each $\boldsymbol{x} \in \mathsf{VAR}^n$: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) = sup\{\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}'}) | \mathcal{A}' \in X\}$,
- $\mathcal{A} = \bigcap X$ iff for each n-ary relation symbol $r \in \mathcal{R}$, for each $\alpha \in \mathsf{ASS}$, for each $\boldsymbol{x} \in \mathsf{VAR}^n$: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) = inf\{\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}'}) | \mathcal{A}' \in X\}$,
- $sup\, \emptyset = 0$, $inf\, \emptyset = 1$.

Clearly, the set of all base equivalent $\mathcal{R}(\mathcal{L})$ -interpretations is a **complete lattice** under the partial ordering of set inclusion.

Next we have to apply the syntactic notions of renaming and variant to the quantitative case. A **renaming** is a bijection $\mathsf{VAR} \rightarrow \mathsf{VAR}$ which is the identity except for finitely many exceptions and VAR is a decidable infinite set of variables.

A quantitative formula κ' is a **ρ-variant** of a quantitative formula κ under a renaming ρ iff $\mathsf{V}(\kappa') = \rho(\mathsf{V}(\kappa))$, where V is a computable function assigning to every quantitative formula κ the set $\mathsf{V}(\kappa)$ of variables occurring in κ; $\kappa' = \kappa\rho$, i.e., κ' is the quantitative formula obtained from κ by simultaneously replacing each occurrence of a variable X in κ by $\rho(X)$ for all variables in $\mathsf{V}(\kappa)$; and $[\![\kappa]\!]^{\mathcal{A}} = [\![\kappa']\!]^{\mathcal{A}}\rho := \{\alpha \circ \rho | \alpha \in [\![\kappa']\!]^{\mathcal{A}}\}$ for each interpretation $\mathcal{A}$.

A quantitative formula κ' is a **variant** of a quantitative formula κ if there exists a renaming ρ s.t. κ' is a ρ-variant of κ.

Using these definitions, we can state the central equations which link the declarative and procedural semantics of $\mathcal{P}_{\mathrm{F}}$.

Definition 7. *Let $\mathcal{P}_{\mathrm{F}}$ be a quantitative definite clause specification in $\mathcal{R}(\mathcal{L})$, $\mathcal{I}$ be an $\mathcal{L}$ -interpretation. Then the countably infinite sequence $\langle \mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots \rangle$ of $\mathcal{R}(\mathcal{L})$ -interpretations extending $\mathcal{I}$ is a $\mathcal{P}_{\mathrm{F}}$ -chain iff for each n-ary relation symbol $r \in \mathcal{R}$, for each $\alpha \in \mathsf{ASS}$, for each $\boldsymbol{x} \in \mathsf{VAR}^n$:*

$\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_0}) := 0,$
$\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{i+1}}) := max\{f \times min\{\mu(\alpha(\boldsymbol{x}_j); q_j^{\mathcal{A}_i}) | \ 1 \leq j \leq n\} \mid$ *there is a variant* $r(\boldsymbol{x}) \leftarrow_f \phi \ \& \ q_1(\boldsymbol{x}_1) \ \& \ \ldots \ \& \ q_n(\boldsymbol{x}_n)$ *of a clause in* $\mathcal{P}_{\mathrm{F}}$ *and* $\alpha \in [\![\phi]\!]^{\mathcal{A}_i}\}$.

Before stating the central theorem concerning the declarative semantics of quantitative definite clause specifications, we have to prove the following useful lemma (cf. [26], Lemmata 2.10', 2.11'):

Lemma 1. *For each $\mathcal{P}_{\mathrm{F}}$, for each $\mathcal{P}_{\mathrm{F}}$ -chain $\langle \mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots \rangle$, for each n-ary relation symbol $r \in \mathcal{R}$, for each $\alpha \in \mathsf{ASS}$, for each $\boldsymbol{x} \in \mathsf{VAR}^n$, there exists some $n \in \mathbb{N}$ s.t. $\mu(\alpha(\boldsymbol{x}); r^{\bigcup_{i \geq 0} \mathcal{A}_i}) = \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n})$.*

Proof. We have to show that the supremum $v = sup\{\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i})|\ i \geq 0\}$ can be attained for some $n \in \mathbb{N}$.

$v = 0$: For $v = 0$, we have $n = 0$.

$v > 0$: For $v > 0$, we have to show for any real ϵ, $0 < \epsilon < v$: $\{\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i})|\ i \geq 0 \textit{ and } \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i}) \geq \epsilon\}$ is finite.
Let F be the finite set of real numbers of factors of clauses in $\mathcal{P}_{\mathrm{F}}$, m be the greatest element in F s.t. $m < 1$ and let q be the smallest integer s.t. $m^q < \epsilon$.
Then, since each real number $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i})$ is a product of a sequence of elements of F, the number of different products $\geq \epsilon$ is not greater than $|F|^q$ (in combinatorics' talk, the permutation of $|F|$ different things taken q at a time with repetitions) and thus finite.
Hence, the supremum is the maximum attained for some $n \in \mathbb{N}$. □

Now we can obtain minimal model properties for quantitative definite clause specifications similar to those for the non-quantitative case of [11]. Theorem 1 states that we can construct a minimal model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$ for each quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ in the extension of an arbitrary constraint language $\mathcal{L}$ and for each $\mathcal{L}$ -interpretation. This means that—due to the definiteness of $\mathcal{P}_{\mathrm{F}}$ —we can restrict our attention to a minimal model semantics of $\mathcal{P}_{\mathrm{F}}$.

Theorem 1 (definiteness). *For each quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ in $\mathcal{R}(\mathcal{L})$, for each $\mathcal{L}$ -interpretation $\mathcal{I}$, for each $\mathcal{P}_{\mathrm{F}}$ -chain $\langle \mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots \rangle$ of $\mathcal{R}(\mathcal{L})$ -interpretations extending some $\mathcal{L}$ -interpretation $\mathcal{I}$:*

(i) $\mathcal{A}_0 \subseteq \mathcal{A}_1 \subseteq \ldots$,
(ii) *the union* $\mathcal{A} := \bigcup_{i \geq 0} \mathcal{A}_i$ *is a model of* $\mathcal{P}_{\mathrm{F}}$ *extending* $\mathcal{I}$,
(iii) $\mathcal{A}$ *is the minimal model of* $\mathcal{P}_{\mathrm{F}}$ *extending* $\mathcal{I}$.

Proof. **(i)** We have to show that $\mathcal{A}_i \subseteq \mathcal{A}_{i+1}$. We prove by induction on i showing for each constraint language $\mathcal{L}$, for each quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ in $\mathcal{R}(\mathcal{L})$, for each $\mathcal{L}$ -interpretation $\mathcal{I}$, for each $\mathcal{P}_{\mathrm{F}}$ -chain $\langle \mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots \rangle$ of $\mathcal{R}(\mathcal{L})$ -interpretations extending some $\mathcal{L}$ -interpretation $\mathcal{I}$, for each n-ary relation symbol $r \in \mathcal{R}$, for each $\alpha \in \mathsf{ASS}$, for each $\boldsymbol{x} \in \mathsf{VAR}^n$, for each $i \in \mathbb{N}$: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{i+1}})$.

Base: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_0}) = 0 \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_1})$.

Hypothesis: Suppose $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{n-1}}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n})$.

Step: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n}) = v > 0$

$\Longrightarrow$ there exists a variant $r(\boldsymbol{x}) \leftarrow_f \phi \ \&\ q_1(\boldsymbol{x}_1) \ \&\ \ldots \ \&\ q_k(\boldsymbol{x}_k)$ of a clause in $\mathcal{P}_{\mathrm{F}}$ s.t. $v = f \times min\{\mu(\alpha(\boldsymbol{x}_1); {q_1}^{\mathcal{A}_{n-1}}), \ldots, \mu(\alpha(\boldsymbol{x}_k); {q_k}^{\mathcal{A}_{n-1}})\}$ and $\alpha \in [\![\phi]\!]^{\mathcal{A}_{n-1}}$, by Definition 7

$\Longrightarrow$ $\mu(\alpha(\boldsymbol{x}_1); {q_1}^{\mathcal{A}_n}) \geq \mu(\alpha(\boldsymbol{x}_1); {q_1}^{\mathcal{A}_{n-1}})$,
$\ldots, \mu(\alpha(\boldsymbol{x}_k); {q_k}^{\mathcal{A}_n}) \geq \mu(\alpha(\boldsymbol{x}_k); {q_k}^{\mathcal{A}_{n-1}})$ and $\alpha \in [\![\phi]\!]^{\mathcal{A}_n}$, by the hypothesis

$\Longrightarrow\ \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{n+1}}) \geq v$, by definition of $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{i+1}})$

$\Longrightarrow\ \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{n+1}})$.

For $v = 0$ follows immediately $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{n+1}})$.

Claim (i) follows by arithmetic induction.

(ii) We have to show that $\mathcal{A} := \bigcup_{i\geq 0} \mathcal{A}_i$ is a model of $\mathcal{P}_{\mathrm{F}}$ extending $\mathcal{I}$. We prove that for each clause $r(\boldsymbol{x}) \leftarrow_f \phi \ \&\ q_1(\boldsymbol{x}_1)\ \&\ \ldots\ \&\ q_k(\boldsymbol{x}_k)$ in $\mathcal{P}_{\mathrm{F}}$, for each $\alpha \in \mathsf{ASS}$: If $\alpha \in [\![\phi]\!]^{\mathcal{A}}$, then $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \geq f \times min\{\mu(\alpha(\boldsymbol{x}_j); q_j{}^{\mathcal{A}})|\ 1 \leq j \leq k\}$.

Note that since every $\mathcal{A}_i$ is an $\mathcal{R}(\mathcal{L})$ -interpretation extending $\mathcal{I}$, $\mathcal{A}$ is an $\mathcal{R}(\mathcal{L})$ -interpretation extending $\mathcal{I}$.

Now let $r(\boldsymbol{x}) \leftarrow_f \phi\ \&\ q_1(\boldsymbol{x}_1)\ \&\ \ldots\ \&\ q_k(\boldsymbol{x}_k)$ be a clause in $\mathcal{P}_{\mathrm{F}}$ s.t. for some $\alpha \in \mathsf{ASS} : \alpha \in [\![\phi]\!]^{\mathcal{A}}$ and $\mu(\alpha(\boldsymbol{x}_i); q_i{}^{\mathcal{A}}) = min\{\mu(\alpha(\boldsymbol{x}_j); q_j{}^{\mathcal{A}})|\ 1 \leq j \leq k\} = v$.

Then there exists some $n \in \mathbb{N}$ s.t. $v = \mu(\alpha(\boldsymbol{x}_i); q_i{}^{\mathcal{A}_n}) = min\{\mu(\alpha(\boldsymbol{x}_j); q_j{}^{\mathcal{A}_n})|$ $1 \leq j \leq k\}$, by Lemma 1 and since for all j s.t. $1 \leq j \leq k : \mu(\alpha(\boldsymbol{x}_j); q_j{}^{\mathcal{A}}) =$ $sup\{\mu(\alpha(\boldsymbol{x}_j); q_j{}^{\mathcal{A}_i})|\ i \geq 0\}$

$\Longrightarrow\ \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{n+1}}) \geq f \times v$, by Definition 7

$\Longrightarrow\ \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \geq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{n+1}})$, since $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) =$ $sup\{\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i})|\ i \geq 0\}$

$\Longrightarrow \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \geq f \times min\{\mu(\alpha(\boldsymbol{x}_j); q_j{}^{\mathcal{A}})|\ 1 \leq j \leq k\}$.

This completes the proof for claim (ii).

(iii) We have to show that $\mathcal{A}$ is the minimal model of $\mathcal{P}_{\mathrm{F}}$ extending $\mathcal{I}$. We prove for every base equivalent model $\mathcal{B}$ of $\mathcal{P}_{\mathrm{F}} : \mathcal{A}_i \subseteq \mathcal{B}$, which gives $\mathcal{A} \subseteq \mathcal{B}$, by induction on i showing for each constraint language $\mathcal{L}$, for each quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ in $\mathcal{R}(\mathcal{L})$, for each $\mathcal{L}$ -interpretation $\mathcal{I}$, for each $\mathcal{P}_{\mathrm{F}}$ -chain $\langle \mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots \rangle$ of $\mathcal{R}(\mathcal{L})$ -interpretations extending some $\mathcal{L}$ -interpretation $\mathcal{I}$, for each n-ary relation symbol $r \in \mathcal{R}$, for each $\alpha \in \mathsf{ASS}$, for each $\boldsymbol{x} \in \mathsf{VAR}^n$, for each $i \in \mathbb{N}$: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{B}})$.

Base: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_0}) = 0 \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{B}})$.

Hypothesis: Suppose $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_{n-1}}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{B}})$.

Step: $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n}) = v > 0$

$\Longrightarrow$ there exists a variant $r(\boldsymbol{x}) \leftarrow_f \phi\ \&\ q_1(\boldsymbol{x}_1)\ \&\ \ldots\ \&\ q_k(\boldsymbol{x}_k)$ of a clause in $\mathcal{P}_{\mathrm{F}}$ s.t. $v = f \times min\{\mu(\alpha(\boldsymbol{x}_1); q_1{}^{\mathcal{A}_{n-1}}), \ldots, \mu(\alpha(\boldsymbol{x}_k); q_k{}^{\mathcal{A}_{n-1}})\}$ and $\alpha \in [\![\phi]\!]^{\mathcal{A}_{n-1}}$, by Definition 7

$\Longrightarrow\ \mu(\alpha(\boldsymbol{x}_1); q_1{}^{\mathcal{B}}) \geq \mu(\alpha(\boldsymbol{x}_1); q_1{}^{\mathcal{A}_{n-1}})$, $\ldots, \mu(\alpha(\boldsymbol{x}_k); q_k{}^{\mathcal{B}}) \geq \mu(\alpha(\boldsymbol{x}_k); q_k{}^{\mathcal{A}_{n-1}})$ and $\alpha \in [\![\phi]\!]^{\mathcal{B}}$, by the hypothesis

$\Longrightarrow\ \mu(\alpha(\boldsymbol{x}); r^{\mathcal{B}}) \geq v$, since $\mathcal{B}$ is a model of $\mathcal{P}_{\mathrm{F}}$

$\Longrightarrow\ \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{B}})$.

For $v = 0$ follows immediately $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_n}) \leq \mu(\alpha(\boldsymbol{x}); r^{\mathcal{B}})$.

Claim (iii) follows by arithmetic induction. $\square$

Proposition 1 allows us to link the declarative description of the desired output from $\mathcal{P}_{\mathrm{F}}$ and a goal, i.e., a $\mathcal{P}_{\mathrm{F}}$ -answer, to the minimal model semantics of $\mathcal{P}_{\mathrm{F}}$. This is done by connecting the concept of logical consequence with the concept of minimal model.

Proposition 1. *Let $\mathcal{P}_{\mathrm{F}}$ be a quantitative definite clause specification in $\mathcal{R}(\mathcal{L})$, φ be an $\mathcal{L}$ -constraint and G be a goal. Then $\varphi\ {}_v\!\!\rightarrow G$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$ iff every minimal model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$ is a model of $\{\varphi\ {}_v\!\!\rightarrow G\}$.*

Proof. if: For each minimal model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$: $\mathcal{A}$ is a model of $\{\varphi\ {}_v\!\!\rightarrow G\}$

$\Longrightarrow$ for every base equivalent model $\mathcal{B}$ of $\mathcal{P}_{\mathrm{F}}$: $\mathcal{B}$ is a model of $\{\varphi\ {}_v\!\!\rightarrow G\}$, since $\mathcal{A} \subseteq \mathcal{B}$ by Theorem 1, (iii)

$\Longrightarrow$ $\varphi\ {}_v\!\!\rightarrow G$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$.

only if: $\varphi\ {}_v\!\!\rightarrow G$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$

$\Longrightarrow$ every model of $\mathcal{P}_{\mathrm{F}}$ is a model of $\{\varphi\ {}_v\!\!\rightarrow G\}$, by Definition 6

$\Longrightarrow$ $\mathcal{A}$ is a model of $\{\varphi\ {}_v\!\!\rightarrow G\}$. $\square$

The following toy example will illustrate the basic concepts of the declarative semantics of quantitative definite clause specifications.

Example 1. Consider a simple program $\mathcal{P}_{\mathrm{F}}$ consisting of clauses 1, 2 and 3. Let for the sake of the example be $[\![X = \phi \ \& \ X = \psi]\!]^{\mathcal{I}} = \emptyset$ for each $\mathcal{L}$ -interpretation $\mathcal{I}$.

1 $p(X) \leftarrow_{.7} X = \phi.$
2 $p(X) \leftarrow_{.5} X = \phi.$
3 $p(X) \leftarrow_{.9} X = \psi.$

A $\mathcal{P}_{\mathrm{F}}$ -chain for predicate p and an object $\alpha(X)$ allowed by the $\mathcal{L}$ -constraint $X = \phi$ is constructed as follows.

$$\begin{aligned}
&\mu(\langle\alpha(X)\rangle\, ; p^{\mathcal{A}_0}) = 0,\\
&\mu(\langle\alpha(X)\rangle\, ; p^{\mathcal{A}_1}) = max\{.7 \times min\ \emptyset, .5 \times min\ \emptyset\} = .7,\\
&\mu(\langle\alpha(X)\rangle\, ; p^{\mathcal{A}_2}) = max\{.7 \times min\ \emptyset, .5 \times min\ \emptyset\} = .7,\\
&\vdots
\end{aligned}$$

The membership value of this object in the denotation of p under the minimal model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$ is attained in step 1 and calculated as follows.

$$\mu(\langle\alpha(X)\rangle\, ; p^{\bigcup_{i\geq 0}\mathcal{A}_i}) = sup\{0, .7, .7, \ldots\} = .7\ .$$

Clearly, $\mathcal{A}$ is a model of clauses 1 and 2. A similar calculation can be done for clause 3.

3.2 Operational Semantics of Quantitative Definite Clause Specifications

The proof procedure for quantitative CLP is a search of a tree, corresponding to the search of an SLD-and/or tree in conventional logic programming and CLP. Such a tree is defined with respect to the inference rules $\xrightarrow{r}$ and $\xrightarrow{c}$ of [11] and a specific calculation of node values. The structure of such a tree exactly reflects the construction of a minimal model and thus may be defined as a min/max tree. In the following we will assume implicit constraint languages $\mathcal{L}$ and $\mathcal{R}(\mathcal{L})$ and a given quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ in $\mathcal{R}(\mathcal{L})$. Furthermore, V will denote the finite set of variables in the query and the V-solutions of a constraint ϕ in an interpretation $\mathcal{I}$ are defined as $[\![\phi]\!]_V^{\mathcal{I}} := \{\alpha|_V | \ \alpha \in [\![\phi]\!]^{\mathcal{I}}\}$ and $\alpha|_V$ is the restriction of α to V.

The first inference rule is given by a binary relation $\xrightarrow{r}$, called goal reduction, on the set of goals.

$A \ \& \ G \xrightarrow{r} F \ \& \ G$ if $A \leftarrow F$ is a variant of a clause in $\mathcal{P}$
s.t. $(\mathsf{V} \cup \mathsf{V}(G)) \cap \mathsf{V}(F) \subseteq \mathsf{V}(A)$.

A second rule takes care of constraint solving for the $\mathcal{L}$-constraints appearing in subsequent goals. The rule takes the conjunction of the $\mathcal{L}$-constraints from the reduced goal and the applied clause and gives, via the black box of a suitable $\mathcal{L}$- constraint solver, a satisfiable $\mathcal{L}$-constraint in solved form if the conjunction of $\mathcal{L}$-constraints is satisfiable. The constraint solving rule can then be defined as a total function $\xrightarrow{c}$ on the set of goals.

$\phi \ \& \ \phi' \ \& \ G \xrightarrow{c} \phi'' \ \& \ G$ if $[\![\phi \ \& \ \phi']\!]^{\mathcal{I}}_{\mathsf{V} \cup \mathsf{V}(G)} = [\![\phi'']\!]^{\mathcal{I}}_{\mathsf{V} \cup \mathsf{V}(G)}$
for each $\mathcal{L}$-interpretation $\mathcal{I}$ and for all $\mathcal{L}$ -constraints ϕ, ϕ' and ϕ''.

Definition 8 (min/max tree). *A min/max tree determined by a query G_1 and a quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$ has to satisfy the following conditions:*

1. *Each max-node is labeled by a goal. The value of each nonterminal max-node is the maximum of the values of its descendants.*
2. *Each min-node is labeled by a clause from $\mathcal{P}_{\mathrm{F}}$ and a goal. The value of each nonterminal min-node is $f \times m$, where f is the factor of the clause and m is the minimum of the values of its descendants.*
3. *The descendants of every max-node are all min-nodes s.t. for every clause C with $\xrightarrow{r}$ -resolvent G' obtained by C from goal G in a max-node, there is a min-node descendant labeled by C and G'.*
4. *The descendants of every min-node are all max-nodes s.t. for every $\mathcal{R}(\mathcal{L})$ - atom $r(\boldsymbol{x})$ in goal $G \,\&\, \phi \,\&\, \phi'$ in a min-node with $\xrightarrow{c}$ -resolvent $G \,\&\, \phi''$, there is a max-node descendant labeled by $r(\boldsymbol{x}) \ \& \ \phi''$.*
5. *The root node is a max-node labeled by G_1.*
6. *A success node is a terminal max-node labeled by a satisfiable $\mathcal{L}$ -constraint. The value of a success node is 1.*

7. *A failure node is a terminal max-node which is not a success node. The value of a failure node is 0.*

Definition 9 (proof tree). *A proof tree for goal G_1 from $\mathcal{P}_F$ is a subtree of a min/max supertree determined by G_1 and $\mathcal{P}_F$ and is defined as follows:*

1. *The root node of the proof tree is the root node of the supertree.*
2. *A max-node of the proof tree is a max-node of the supertree and takes* one *of the descendants of the supertree max-node as its descendant.*
3. *A min-node of the proof tree is a min-node of the supertree and takes* all *of the descendants of the supertree max-node as its descendants.*
4. *All terminal nodes in the proof tree are success nodes ϕ, ϕ', ... s.t. ϕ & ϕ' & $\ldots \xrightarrow{c} \varphi$ and φ is a satisfiable $\mathcal{L}$ -constraint, called answer constraint.*
5. *Values are assigned to proof tree nodes in the same way as to min/max tree nodes.*

To prove soundness and completeness of this generalized SLD-resolution proof procedure, some further concepts have to be introduced.

First, we have to take care of **renaming closure** of the generalized constraint language $\mathcal{R}(\mathcal{L})$. A constraint language is said to be closed under renaming iff every constraint has a ρ-variant for every renaming ρ. Clearly, $\mathcal{R}(\mathcal{L})$ is closed under renaming if the underlying constraint language $\mathcal{L}$ is closed under renaming. Furthermore, for each $\mathcal{R}(\mathcal{L})$ closed under renaming, for each $\mathcal{R}(\mathcal{L})$ -interpretation $\mathcal{A}$: $\mathcal{A}$ is a model of an $\mathcal{R}(\mathcal{L})$ -constraint iff $\mathcal{A}$ is a model of each of its variants.

Next, we have to redefine a **complexity measure** for goal reduction for the quantitative case. This measure is crucial in proving termination of goal reduction and works by keying steps of the minimal model construction to steps of the goal reduction process.

- The complexity of a variable assignment α for an atom $r(\boldsymbol{x})$ in the minimal model $\mathcal{A}$ s.t. $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) > 0$ is defined as $comp(\alpha, r(\boldsymbol{x}), \mathcal{A}) := min\{i| \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) = \mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}_i})\}$.
- The complexity of α for goal $G = r_1(\boldsymbol{x}_1)$ & ... & $r_k(\boldsymbol{x}_k)$ & ϕ in $\mathcal{A}$ s.t. $\alpha \in [\![\phi]\!]^{\mathcal{A}}$ and $\mu(\alpha(\boldsymbol{x}_i); r_i{}^{\mathcal{A}}) > 0$ for all $i : 1 \leq i \leq k$ is defined as $comp(\alpha, G, \mathcal{A}) := \{comp(\alpha, r_i(\boldsymbol{x}_i), \mathcal{A})| \ 1 \leq i \leq k\}$ where $\{\ldots\}$ is a multiset.
- The V-complexity of α for goal $G = r_1(\boldsymbol{x}_1)$ & ... & $r_k(\boldsymbol{x}_k)$ & ϕ in $\mathcal{A}$ s.t. $\alpha \in [\![\phi]\!]^{\mathcal{A}}_V$ and $\mu(\alpha(\boldsymbol{x}_i); r_i{}^{\mathcal{A}}) > 0$ for all $i : 1 \leq i \leq k$ is defined as $comp_V(\alpha, G, \mathcal{A}) := min\{comp(\beta, G, \mathcal{A})| \ \beta \in [\![\phi]\!]^{\mathcal{A}}, \ \mu(\beta(\boldsymbol{x}_i); r_i{}^{\mathcal{A}}) > 0$ for all $i : 1 \leq i \leq k$ and $\alpha = \beta|_V\}$ where the minimum is taken with respect to a total ordering on multisets s.t. $M \leq M'$ iff $\forall x \in M \setminus M', \exists x' \in M' \setminus M$ s.t. $x < x'$.

Clearly, the constraint solving part of the deduction scheme does not affect the denotation or complexity of subsequent goals.

The following proofs show that the quantitative proof procedure is sound and complete with respect to the above stated semantic concepts. Again, there is a close similarity to the corresponding statements for the non-quantitative case of [11].

Theorem 2 (soundness). *For each quantitative definite clause specification $\mathcal{P}_{\mathrm{F}}$, for each goal G, for each $\mathcal{L}$-constraint φ: If there is a proof tree for G from $\mathcal{P}_{\mathrm{F}}$ with answer constraint φ and root value v, then $\varphi \;_v\!\rightarrow G$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$.*

Proof. The result is proved by induction on the depth d of the proof tree, where one unit of depth is from max-node to max-node.

Base: We know that proof trees of depth $d = 0$ have to take the form of a single max-node labeled by a satisfiable $\mathcal{L}$-constraint ψ with root value 1. Then $\psi \;_1\!\rightarrow \psi$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$.

Hypothesis: Suppose the result holds for proof trees of depth $d < n$.

Step: Let $G_0 = r(\boldsymbol{x})\ \&\ \phi$ be a goal labeling a proof tree of depth $d = n$ with answer constraint ψ and root value h,
let $G'_0 = q_1(\boldsymbol{x}_1)\ \&\ \dots\ \&\ q_k(\boldsymbol{x}_k)\ \&\ \phi\ \&\ \phi'$ be a goal labeling the min-node obtained from G_0 via $\xrightarrow{r}$ using the variant $C' = r(\boldsymbol{x}) \leftarrow_f \phi'\ \&\ q_1(\boldsymbol{x}_1)\ \&\ \dots\ \&\ q_k(\boldsymbol{x}_k)$ of a clause C in $\mathcal{P}_{\mathrm{F}}$,
and let $G_1 = q_1(\boldsymbol{x}_1)\ \&\ \phi'', \dots, G_k = q_k(\boldsymbol{x}_k)\ \&\ \phi''$ be goals labeling max-nodes obtained from G'_0 via $\xrightarrow{c}$.

Then each goal $G_1, \dots, G_k$ labels a proof tree of depth $d < n$ with respective answer constraint $\psi_1, \dots, \psi_k$ and root value $g_1, \dots, g_k$ s.t. $h = f \times min\{g_1, \dots, g_k\}$ and for each model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$: $[\![\psi]\!]^{\mathcal{A}} = [\![\psi_1\ \&\ \dots\ \&\ \psi_k]\!]^{\mathcal{A}}$, by definition min/max tree

$\Longrightarrow$ $\psi_1 \;_{g_1}\!\rightarrow G_1, \dots, \psi_k \;_{g_k}\!\rightarrow G_k$ are logical consequences of $\mathcal{P}_{\mathrm{F}}$, by the hypothesis

$\Longrightarrow$ for each model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$, for each $\alpha \in \mathsf{ASS}$: $[\![\psi]\!]^{\mathcal{A}} \subseteq [\![\phi'']\!]^{\mathcal{A}}$ and if $\alpha \in [\![\psi]\!]^{\mathcal{A}}$, then $\mu(\alpha(\boldsymbol{x}_1); {q_1}^{\mathcal{A}}) \geq g_1, \dots, \mu(\alpha(\boldsymbol{x}_k); {q_k}^{\mathcal{A}}) \geq g_k$, by definition of logical consequence

$\Longrightarrow$ for each model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$, for each $\alpha \in \mathsf{ASS}$: $[\![\psi]\!]^{\mathcal{A}} \subseteq [\![\phi']\!]^{\mathcal{A}}$ and if $\alpha \in [\![\psi]\!]^{\mathcal{A}}$, then $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \geq f \times min\{\mu(\alpha(\boldsymbol{x}_1); {q_1}^{\mathcal{A}}), \dots, \mu(\alpha(\boldsymbol{x}_k); {q_k}^{\mathcal{A}})\}$, since each model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$ is a model of C' iff $\mathcal{A}$ is a model of C

$\Longrightarrow$ for each model $\mathcal{A}$ of $\mathcal{P}_{\mathrm{F}}$, for each $\alpha \in \mathsf{ASS}$: $[\![\psi]\!]^{\mathcal{A}} \subseteq [\![\phi]\!]^{\mathcal{A}}$ and if $\alpha \in [\![\psi]\!]^{\mathcal{A}}$, then $\mu(\alpha(\boldsymbol{x}); r^{\mathcal{A}}) \geq h$

$\Longrightarrow$ $\psi \;_h\!\rightarrow r(\boldsymbol{x})\ \&\ \phi$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$.

The result follows by arithmetic induction. □

Theorem 3 (completeness). *Let $\mathcal{P}_{\mathrm{F}}$ be a quantitative definite clause specification in $\mathcal{R}(\mathcal{L})$, $\mathcal{L}$ be closed under renaming, $\mathcal{A}$ be a minimal model of $\mathcal{P}_{\mathrm{F}}$, G be a goal of the form $r(\boldsymbol{x})\ \&\ \phi$, $\alpha \in [\![\phi]\!]^{\mathcal{A}}_V$ and $\mu(\beta(\boldsymbol{x}); r^{\mathcal{A}}) = v$ s.t. $v > 0$ and $\alpha = \beta|_V$. Then there exists a proof tree for G from $\mathcal{P}_{\mathrm{F}}$ with answer constraint φ and root value v and $\alpha \in [\![\varphi]\!]^{\mathcal{A}}_V$.*

Proof. The result is proved by induction on $c = comp_V(\alpha, G, \mathcal{A})$.

Base: We know that goals with complexity $c = \emptyset$ have to take the form of a satisfiable $\mathcal{L}$-constraint χ. Then there exists a proof tree for χ from $\mathcal{P}_{\mathrm{F}}$ consisting of a single max-node labeled with χ and root value 1.

Hypothesis: Suppose the result holds for goals with complexity $c < N$.

Step: Let $G_0 = q(\boldsymbol{x})\ \&\ \psi$, $\alpha' \in [\![\psi]\!]_V^{\mathcal{A}}$, $\alpha'' \in [\![\psi]\!]^{\mathcal{A}}$, $\alpha' = \alpha''|_V$, $comp_V(\alpha', G_0, \mathcal{A}) = comp(\alpha'', G_0, \mathcal{A}) = N$, $comp(\alpha'', q(\boldsymbol{x}), \mathcal{A}) := i$, $\mu(\alpha''(\boldsymbol{x}); q^{\mathcal{A}}) = h$ and $h > 0$.

First we observe, that $\mu(\alpha''(\boldsymbol{x}); q^{\mathcal{A}_i}) = h$, since $comp(\alpha'', q(\boldsymbol{x}), \mathcal{A}) := i$

$\Longrightarrow$ there exists a variant $q(\boldsymbol{x}) \leftarrow_f \psi'\ \&\ q_1(\boldsymbol{x}_1)\ \&\ \ldots\ \&\ q_k(\boldsymbol{x}_k)$ s.t. $h = f \times min\{\mu(\alpha(\boldsymbol{x}_1); {q_1}^{\mathcal{A}_{i-1}}), \ldots, \mu(\alpha(\boldsymbol{x}_k); {q_k}^{\mathcal{A}_{i-1}})\}$ and $\alpha'' \in [\![\psi']\!]^{\mathcal{A}_{i-1}}$ and $(\mathsf{V} \cup \mathsf{V}(\psi)) \cap \mathsf{V}(\psi'\ \&\ q_1(\boldsymbol{x}_1)\ \&\ \ldots\ \&\ q_k(\boldsymbol{x}_k)) \subseteq \mathsf{V}(q(\boldsymbol{x}))$, by definition 7 and renaming closure of $\mathcal{R}(\mathcal{L})$, finite V and infinitely many variables in VAR

$\Longrightarrow$ $G_0 \xrightarrow{r,c} G_0'$ s.t. $G_0' = q_1(\boldsymbol{x}_1)\ \&\ \ldots\ \&\ q_k(\boldsymbol{x}_k)\ \&\ \psi''$ and $[\![\psi'']\!]_V^{\mathcal{A}} = [\![\psi\ \&\ \psi']\!]_V^{\mathcal{A}}$, by definition of the inference rules.

Next, $\alpha' \in [\![\psi'']\!]_V^{\mathcal{A}}$, since $\alpha'' \in [\![\psi]\!]^{\mathcal{A}}$, $\alpha'' \in [\![\psi']\!]^{\mathcal{A}_{i-1}} \subseteq [\![\psi']\!]^{\mathcal{A}}$, $\alpha'' \in [\![\psi\ \&\ \psi']\!]^{\mathcal{A}}$, $[\![\psi\ \&\ \psi']\!]_V^{\mathcal{A}} = [\![\psi'']\!]_V^{\mathcal{A}}$ and $\alpha' = \alpha''|_V$.

Finally, $comp_V(\alpha', G_0', \mathcal{A}) < N$, since $comp_V(\alpha', G_0', \mathcal{A}) \leq comp(\alpha'', G_0', \mathcal{A}) < \{i\} = \{comp(\alpha'', q(\boldsymbol{x}), \mathcal{A})\} = comp(\alpha'', G_0, \mathcal{A}) = comp_V(\alpha', G_0, \mathcal{A}) = N$.

Now we can obtain goals $G_1 = q_1(\boldsymbol{x}_1)\ \&\ \psi'', \ldots, G_k = q_k(\boldsymbol{x}_k)\ \&\ \psi''$ from G_0' s.t. $\alpha' \in [\![\psi'']\!]_V^{\mathcal{A}}$, $\mu(\alpha''(\boldsymbol{x}_1); {q_1}^{\mathcal{A}}) = g_1 > 0, \ldots, \mu(\alpha''(\boldsymbol{x}_k); {q_k}^{\mathcal{A}}) = g_k > 0$, $\alpha' = \alpha''|_V$ and $comp_V(\alpha', G_1, \mathcal{A}) < N, \ldots, comp_V(\alpha', G_k, \mathcal{A}) < N$.

$\Longrightarrow$ for each goal $G_1, \ldots, G_k$, there exists a proof tree from $\mathcal{P}_{\mathrm{F}}$ with respective answer constraint $\chi_1, \ldots, \chi_k$ and respective root value $g_1' = g_1, \ldots, g_k' = g_k$ and $\alpha' \in [\![\chi_1\ \&\ \ldots\ \&\ \chi_k]\!]_V^{\mathcal{A}} = [\![\chi]\!]_V^{\mathcal{A}}$, by the hypothesis

$\Longrightarrow$ there exists a proof tree for G_0 from $\mathcal{P}_{\mathrm{F}}$ with answer constraint χ and root value $h' = f \times min\{g_1', \ldots, g_k'\} = f \times min\{g_1, \ldots, g_k\} = h$ and $\alpha' \in [\![\chi]\!]_V^{\mathcal{A}}$.

The result follows by arithmetic induction. □

Returning to our toy example, the proof procedure for quantitative definite clause specifications can be illustrated as follows.

Example 2. Starting from the simple program of Example 1, a min/max tree for query $p(X)\ \&\ X = \phi$ and $\mathcal{P}_{\mathrm{F}}$ is constructed as follows.

$$\begin{tikzcd}[column sep=small]
& \begin{array}{c} p(X) \,\&\, X = \phi \\ \mathbf{max\{.7, .5, 0\}} \end{array} \arrow[dl, dash, "r"'] \arrow[d, dash, "r"] \arrow[dr, dash, "r"] & \\
\begin{array}{c} 1, X = \phi \,\&\, X = \phi \\ \mathbf{.7} \times \mathbf{min\{1\}} \end{array} \arrow[d, dash, "c"] & \begin{array}{c} 2, X = \phi \,\&\, X = \phi \\ \mathbf{.5} \times \mathbf{min\{1\}} \end{array} \arrow[d, dash, "c"] & \begin{array}{c} 3, X = \psi \,\&\, X = \phi \\ \mathbf{.9} \times \mathbf{min\{0\}} \end{array} \arrow[d, dash, "c"] \\
\begin{array}{c} X = \phi \\ \mathbf{1} \end{array} & \begin{array}{c} X = \phi \\ \mathbf{1} \end{array} & \begin{array}{c} - \\ \mathbf{0} \end{array}
\end{tikzcd}$$

This tree contains two success branches and one failure branch (from left to right). The proof trees obtained from this min/max tree are as follows.

$$\begin{tikzcd}
\begin{array}{c} p(X) \,\&\, X = \phi \\ \mathbf{max\{.7\}} \end{array} \arrow[d, dash, "r"] \\
\begin{array}{c} 1, X = \phi \,\&\, X = \phi \\ \mathbf{.7} \times \mathbf{min\{1\}} \end{array} \arrow[d, dash, "c"] \\
\begin{array}{c} X = \phi \\ \mathbf{1} \end{array}
\end{tikzcd}$$

$$\begin{tikzcd}
\begin{array}{c} p(X) \,\&\, X = \phi \\ \mathbf{max\{.5\}} \end{array} \arrow[d, dash, "r"] \\
\begin{array}{c} 2, X = \phi \,\&\, X = \phi \\ \mathbf{.5} \times \mathbf{min\{1\}} \end{array} \arrow[d, dash, "c"] \\
\begin{array}{c} X = \phi \\ \mathbf{1} \end{array}
\end{tikzcd}$$

Clearly, $X = \phi \;{}_{.7}\!\!\rightarrow p(X) \;\&\; X = \phi$ is a logical consequence of $\mathcal{P}_{\mathrm{F}}$.

As proposed by [26], search strategies such as alpha-beta pruning (see [22]) can be used quite directly to define an interpreter for quantitative rule sets. The same techniques can be applied to a min/max proof procedure in quantitative CLP. In general, the amount of search needed to find the best proof for a goal, i.e., the maximal valued proof tree for a goal from a program, will be reduced remarkably by controlling the search by the alpha-beta algorithm.

4 Quantitative CLP and Weighted CLGs

To sum up, the quantitative CLP scheme presented above allows for a definition of the parsing problem (and similarly of the generation problem) for weighted

CLGs in the following way: Given a program $\mathcal{P}_F$ (coding some weighted CLG) and a query G (coding some input string), we ask if we can infer a $\mathcal{P}_\mathrm{F}$ -answer φ of G (coding an analysis) at a value v (coding the weight of the analysis) proving $\varphi \ {}_v\!\rightarrow G$ to be a logical consequence of $\mathcal{P}_\mathrm{F}$. The concept of weighted logical consequence thus can be seen as a model-theoretic counterpart to the operational concept of weighted inference.

We showed soundness and completeness results for a general proof procedure for quantitative constraint logic programs with respect to a simple declarative semantics based on concepts of fuzzy set algebra. These terms in turn allow for a deeper characterization of the concept of weighted logical consequence: A $\mathcal{P}_\mathrm{F}$ -answer to a query $G = r(\boldsymbol{x})\ \&\ \phi$ at value v is a satisfiable $\mathcal{L}$ -constraint φ such that for each model $\mathcal{A}$ of $\mathcal{P}_\mathrm{F}$ holds: If φ is satisfiable, then ϕ is satisfiable and all objects assigned to $\boldsymbol{x}$ by a solution of φ are in the denotation of $r(\boldsymbol{x})$ at a membership value of at least v.

Considering concrete instantiations and applications of this formal scheme, the remaining question is how to give the concept of weight an intuitive interpretation. In the following we will briefly discuss two possible interpretations of weighted CLGs each of which is determined by the specific aims of a specific application.

One interpretation of weights is as a correlate to the degree of grammaticality of an analysis. In [8, 9], Erbach attempts to calculate the degree of grammaticality of an analysis from the application probabilities of clauses used in the analysis and additional user-defined weights.[3] Regardless of the motivation for this specific determination of degrees of grammaticality, the choice to interpret weights in correspondence to degrees of grammaticality severely restricts the possible applications of such weighted CLGs.

Considering for example the problem of ambiguity resolution which is also addressed by Erbach, we think that the concepts of preference value and degree of grammaticality should be clearly differentiated. As discussed in [1], the problem of ambiguity resolution cannot be reduced to some few unrealistic examples. Instead, when describing a nontrivial part of natural language, grammars of the usual sort will produce massive artificial ambiguity where we can find grammatical readings even for the most abstruse analyses. Suppose for example a grammar which licenses, among many others, analyses such as *1) John believes* $[$*Peter saw Mary*$]_S$ and *2) John believes* $[New\ York_N\ taxi_N\ drivers_N]_{NP}$. Such a grammar would also license the analysis *3) John believes* $[Peter_N\ saw_N\ Mary_N]_{NP}$ (provided a noun entry for the noun reading of *saw*), which is clearly less preferred than *1)*. Analysis *3)* otherwise is not less grammatical, as we can find an acceptable reading (where the NP refers to the $Mary$ associated with some kind of *saw* called a *Peter saw*). Degrading the weight of the rule $NP \rightarrow N$

[3] Erbach sketches a calculation scheme which employs a restricted summation over clause weights instead of a minimization as is done in our quantitative CLP scheme. This calculation scheme could easily be captured by our quantitative CLP scheme by replacing *min* by a restricted sum in the relevant definitions of the declarative semantics and accordingly of the procedural semantics of our scheme.

N N (licensing multiple nominal modifications) would on the other hand also degrade the weight of *2)*, which prevents a disambiguation by an interpretation of weights in terms of degrees of grammaticality.

Considering the problem of graded grammaticality, it seems necessary to employ richer models for a determination of degrees of grammaticality. A first attempt to incorporate degrees of grammaticality investigated by psycholinguistic experiments into CLGs is presented in [13, 14].[4] Weighted CLGs interpreted in a serious framework of graded grammaticality then could provide a valuable framework for a clear procedural and declarative treatment of graded grammaticality in CLGs.

Another interpretation of weighted CLGs is possible from the viewpoint of probabilistic grammars. This approach has been shown to be fruitful, e.g., for the problem of ambiguity resolution. The simple but useful approximation adopted here is to assume the most plausible analysis of a string to be the most probable analysis of that string.

An attempt to transfer the techniques of probabilistic context-free grammars (see [3]) to CLGs was presented in [7]. In this approach the derivation process of CLGs is defined as a stochastic process by the following stochastic model: Each program clause gets assigned an application probability and the probabilities of all clauses defining one predicate have to sum to 1. The probability of a proof tree is calculated as the product of the probabilities of the rules used in it.[5] In order to make the probabilities of proof trees as defined by the stochastic model constitute a proper probability distribution, an additional normalization with respect to the overall probability of proof trees has to be made.[6]

What is interesting about probabilistic language models is their ability to estimate the probabilistic parameters of the model in accord to empirical probability distributions. Eisele attempts to estimate the probability of clauses proportional to the expected frequency of clauses in derivations. Unfortunately, this approach to parameter estimation is incorrect when applied to the probabilistic CLG model of Eisele. This means that the probability distribution over proof trees as defined by a probabilistic CLG model estimated by the expected clause frequency method is not in accord with the frequency of the proof trees in the training corpus. Similarly, when dealing with unparsed corpora, the EM-algorithm used for parameter estimation optimizes the wrong function when applied to this model. The reason for this incorrectness is that the set of trees generated from such a probabilistic CLG model is constrained in a way which violates basic assumptions made in the applied parameter estimation method. In other words, the probabilistic CLG model defined by Eisele could be said to

[4] Keller concentrates on experimental investigation of degrees of grammaticality and sketches a model of graded grammaticality based on ranked constraints. Such a model should easily be given a formal basis in terms of our quantitative CLP scheme.

[5] This calculation scheme also could easily be captured by our quantitative CLP scheme by replacing *min* by a product accordingly in the relevant definitions of the declarative and procedural semantics of our scheme.

[6] [3] discuss further conditions on consistency of probabilistic grammars which would have to be satisfied also by a probabilistic CLG model.

be incorrect, in the sense that it makes an independency assumption for clause applications which is violated by the languages generated from such probabilistic CLGs.

Since the proposed parameter estimation method is only provably correct for the context-free case, the probabilistic language model of Eisele faces a severe restriction. The only approach we know of to present a correct parameter estimation algorithm for probabilistic grammars involving context-dependencies is the model of stochastic attribute-value grammars of [2], a discussion of which is beyond our present scope.

5 Conclusion

We presented a simple and general scheme for quantitative CLP. Our quantitative extension straightforwardly transferred the nice properties of the CLP scheme of [11] into close analogs holding for a quantitative version of CLP. With respect to related approaches to quantitative extensions of conventional logic programming, our extension raises ideas from this area to the general framework of CLP.

We showed soundness and completeness results with respect to a declarative semantics based on concepts of fuzzy set algebra. This approach to a declarative semantics was motivated by the aim to give a clear and simple formal semantics for weighted CLGs.

Clearly, more expressive quantitative extensions of CLP are possible and will be addressed in future work. Regarding the interest in computational linguistics problems such as ambiguity resolution, however, a necessary prerequisite for a more sophisticated semantics for probabilistically interpreted quantitative CLP is the development of a probabilistic model for CLP which allows for correct parameter estimation from empirical data.

References

1. Steven Abney. Statistical methods and linguistics. In Judith Klavans and Philip Resnik, editors, *The Balancing Act*. The MIT Press, Cambridge, MA, 1996.
2. Steven Abney. Stochastic attribute-value grammars. *Computational Linguistics*, 23(4):597–618, 1997.
3. Taylor L. Booth and Richard A. Thompson. Applying probability measures to abstract languages. *IEEE Transactions on Computers*, C-22(5):442–450, 1973.
4. Jochen Dörre and Michael Dorna. CUF - a formalism for linguistic knowledge representation. In Jochen Dörre, editor, *Computational Aspects of Constraint-Based Linguistic Description I*, pages 3–22. DYANA-2 Deliverable R1.2.A, 1993.
5. Jochen Dörre and Andreas Eisele. A comprehensive unification-based grammar formalism. Technical report, DYANA Deliverable R3.1.B, 1991.
6. Didier Dubois, Jérôme Lang, and Henri Prade. Towards possibilistic logic programming. In *Proceedings of the 8th International Conference on Logic Programming (ICLP '91)*, Paris, 1991.

7. Andreas Eisele. Towards probabilistic extensions of constraint-based grammars. In Jochen Dörre, editor, *Computational Aspects of Constraint-Based Linguistic Description II*, pages 3–21. DYANA-2 Deliverable R1.2.B, 1994.
8. Gregor Erbach. Towards a theory of degrees of grammaticality. CLAUS Report 34, Computational Linguistics at the Univerity of the Saarland, 1993.
9. Gregor Erbach. *Bottom-Up Earley Deduction for Preference-Driven Natural Language Processing*. PhD thesis, Computational Linguistics, Universität des Saarlandes, Saarbrücken, 1998.
10. Thilo Götz. Compiling HPSG constraint grammars into logic programs. In *Workshop on Computational Logic and Natural Language Processing*, Edinburgh, 1995.
11. Markus Höhfeld and Gert Smolka. Definite relations over constraint languages. LILOG Report 53, IBM Deutschland, Stuttgart, 1988.
12. Joxan Jaffar and Jean-Louis Lassez. Constraint logic programming. Technical Report 74, Department of Computer Science, Monash University, 1986.
13. Frank Keller. Extraction from complex noun phrases. A case study in graded grammaticality. Master's thesis, Institut für maschinelle Sprachverarbeitung, Universität Stuttgart, 1996.
14. Frank Keller. How do humans deal with ungrammatical input? Experimental evidence and computational modelling. In Dafydd Gibbon, editor, *Natural Language Processing and Speech Technology. Results of the 3rd KONVENS Conference*, Berlin, 1996. Mouton de Gruyter.
15. Michael Kifer and V. S. Subrahmanian. Theory of generalized annotated logic programming and its applications. *Journal of Logic Programming*, 12:335–367, 1992.
16. Paul J. King. An expanded logical formalism for head-driven phrase structure grammar. Arbeitspapiere des Sonderforschungsbereichs 340 59, University of Tübingen, 1994.
17. Laks V. S. Lakshmanan and Fereidoon Sadri. Probabilistic deductive databases. In *Proceedings of the International Logic Programming Symposium (ILPS '94)*, Ithaca, NY, 1994.
18. Laks V. S. Lakshmanan and Fereidoon Sadri. On a theory of probabilistic deductive databases. unpublished manuscript, October 1997.
19. J. W. Lloyd. *Foundations of Logic Programming*. Springer, Berlin, 1987.
20. Raymond Ng and V. S. Subrahmanian. Probabilistic logic programming. *Information and Computation*, 101:150–201, 1992.
21. Raymond Ng and V. S. Subrahmanian. A semantical framework for supporting subjective and conditional probabilities in deductive databases. *Journal of Automated Reasoning*, 10:191–235, 1993.
22. Nils J. Nilsson. *Principles of Artificial Intelligence*. Springer, Berlin, 1982.
23. Carl Pollard and Ivan A. Sag. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago, 1994.
24. Gert Smolka. A feature logic with subsorts. LILOG Report 33, IBM Deutschland, Stuttgart, 1988.
25. V. S. Subrahmanian. On the semantics of quantitative logic programs. In *Proceedings of the 4th IEEE Symposium on Logic Programming*, pages 173–182, Washington D.C., 1987. Computer Society Press.
26. M. H. van Emden. Quantitative deduction and its fixpoint theory. *The Journal of Logic Programming*, 1:37–53, 1986.
27. L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.