Default Conceptual Graph Rules: Preliminary Results for an Agronomy Application

Jean-François Baget[1,2], Madalina Croitoru[2], Jérôme Fortin[2], and Rallou Thomopoulos[3,2]

[1] INRIA Sophia Antipolis, 2004 Route des Lucioles 06902 Sophia Antipolis, France
baget@lirmm.fr
[2] LIRMM (CNRS & Université Montpellier II), F–34392 Montpellier Cedex 5, France
{croitoru,fortin}@lirmm.fr
[3] INRA, UMR1208, F–34060 Montpellier Cedex 1, France
rallou.thomopoulos@supagro.inra.fr

Abstract. In this paper, we extend Simple Conceptual Graphs with Reiter's default rules. The motivation for this extension came from the type of reasonings involved in an agronomy application, namely the simulation of food processing. Our contribution is many fold: first, the expressivity of this new language corresponds to our modeling purposes. Second, we provide an effective characterization of sound and complete reasonings in this language. Third, we identify a decidable subclass of Reiter's default logics. Last we identify our language as a superset of $\mathcal{SREC}^-$, and provide the lacking semantics for the latter language.

1 Introduction and Motivation

The modeling need that motivated this paper came from an agronomy application: the simulation of food processing (more specifically the pasta drying process). In this application, successive unit operations involved in the drying process have different impacts on product qualities. These impacts can be positive or negative, non monotonically depending on the considered quality and the concerned unit operation. 46 kinds of qualities have been identified for pasta products, moreover these qualities can themselves be subdivided into taxonomies of components (e.g. sub-families of vitamins) that behave differently, hence the need to account for particular cases concerning specific subfamilies [1]. The choice of Conceptual Graphs (CGs) as a modeling language stems from the intuitiveness of their graphical representation as well as the possibility to use their structure for optimization purposes.

Generally, languages of the CG family have a semantics that can be expressed in first-order logic (FOL). The non-monotonic features of the knowledge we want to represent for this application calls for an extension of these languages. The extension we consider here is based upon Reiter's default logics. This formalism has been designed to extend FOL with knowledge like "something is true unless we believe something else". On top of the traditional constructs of FOL-based CG languages (support, facts, rules, constraints), a knowledge base of our new

S. Rudolph, F. Dau, and S.O. Kuznetsov (Eds.): ICCS 2009, LNAI 5662, pp. 86–99, 2009.

language also consists of default CG rules inspired by Reiter's defaults. These default CG rules fully generalize CG rules and not only the type hierarchy as done in [2]. In this paper we formally present this language and illustrate it with motivating examples from our application.

We define the classical notions of conceptual graphs, rules and constraints in SECT. 2. SECT. 3 is devoted to introducing default reasoning. We recall in SECT. 3.2 Reiter's default formalism, and we introduce the syntax and semantics of default CG rules in SECT. 3.3. Finally, theoretical results are presented in SECT. 4: in SECT. 4.1 we introduce the derivation tree for default conceptual graphs and present a subclass of default CG rules for which this tree is finite. In SECT. 4.2 we use this tree for sound and complete reasonings. Finally, in SECT. 4.3 we relate our new language with the $\mathcal{SREC}^-$ of [3]. The paper concludes with future directions of work.

2 Conceptual Graphs, Rules and Constraints

In this section, we recall essential results about conceptual graphs (CGs). The different languages presented here are described in more detail in [3]. They all form a subset of first-order logics (FOL) since all objects introduced (support, graphs, rules or constraints) have a FOL semantics obtained via the transformation Φ ($\Phi(X)$ is thus the *logical interpretation* of the object X). In all these languages, we will consider a *knowledge base* (KB) containing different objects (*i.e.* support, graphs, rules and/or constraints).

Definition 1 (Semantics of a knowledge base). *The logical interpretation $\Phi(\mathcal{K})$ of a KB $\mathcal{K}$ is the conjunction of the logical interpretations $\Phi(X)$ of the objects X it contains. A KB $\mathcal{K}$ is said* satisfiable *if the FOL formula $\Phi(\mathcal{K})$ is satisfiable. If Q is a simple CG, we say that Q can be* deduced *from $\mathcal{K}$, and note $\mathcal{K} \models Q$, if $\Phi(Q)$ is a semantic consequence of $\Phi(\mathcal{K})$.*

We are interested here in the $\mathcal{X}$-SATISFIABILITY and $\mathcal{X}$-DEDUCTION problems, where $\mathcal{X}$ is a language of the CG family defined by the kinds of objects allowed in a KB. A $\mathcal{SG}$ KB contains only a support and a (set of) simple CG(s). A $\mathcal{SR}$ KB is the union of a $\mathcal{SG}$ KB with a set of rules, and a $\mathcal{SGC}^-$ KB the union of a $\mathcal{SG}$ KB with a set of negative constraint. A $\mathcal{SRC}^-$ KB is the union of a $\mathcal{SR}$ KB and of a $\mathcal{SGC}^-$ KB. The three following subsections successively present the syntax of the different objects that can compose a KB, their logical interpretations, and recall essential results allowing to compute $\mathcal{X}$-SATISFIABILITY and $\mathcal{X}$-DEDUCTION in these different languages.

2.1 Simple Conceptual Graphs: The $\mathcal{SG}$ Language

Syntax: Support and Simple CGs (SGs) A KB of the $\mathcal{SG}$ language is composed solely of a *support* (encoding a type hierarchy) and of a set of *simple CGs* (that represent entities and relationships between them).

Definition 2 (Support). *A* support *is a tuple* $\mathcal{S} = (T_C, T_R^1, \cdots, T_R^k, M)$ *whose elements are partially ordered, pairwise disjoint sets, respectively of* concepts types, relation types *of arity* $1, \cdots, k$, *and of* markers. *All partial orders are noted* $\leq$. *Markers are partitioned into an infinite set* M_G *of* (named) generic *markers (if* m *is generic, then* $\forall m' \in M, m' \leq m$*) and a set* M_I *of* individual *markers (that are pairwise non-comparable).*

Definition 3 (Simple conceptual graph). *A* simple conceptual graph *(or SG) defined on a support* $\mathcal{S} = (T_C, T_R^1, \cdots, T_R^k, M)$ *is a tuple* $G = (C, R, \gamma, \epsilon)$ *where* C *and* R *are disjoint finite sets, respectively of* concepts *and* relations. *The mapping* $\gamma : R \to C^+$ *associates to each relation a tuple of concepts* $\gamma(r) = (c_1, \cdots, c_p)$ *called the* arguments *of the relation. We note* $\gamma_i(r) = c_i$ *its* i^{th} *argument. The mapping* $\epsilon : C \cup R \to (2^{T_C} \times M) \cup_{1 \leq i \leq k} T_R^i$ labels *each concept and relation. If* c *is a concept of* C, *then* $\epsilon(c) = (t, m) \in 2^{T_c} \times M$ *(*t *is called the* type *of* c, m *is called its* marker*). If* $m \in M_G$ *then* c *is called* generic, *otherwise it is called* individual. *If* r *is a relation of* R, *then its type* $\epsilon(r) \in \cup_i T_R^i$ *must have the correct arity, i.e.* $|\gamma(r)| = j \Leftrightarrow \epsilon(r) \in T_R^j$.

A simple CG representation of information about "a pasta product that contains peroxidase and is undergoing a late end-of-cycle temperature drying" is presented in FIG. 1.

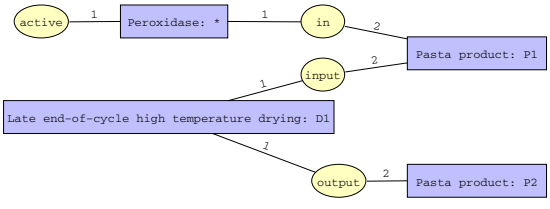

Fig. 1. The SG representing "a pasta product that contains peroxidase and is undergoing a late end-of-cycle temperature drying"

Note that (as required by our modeling, and as defined, for example, in [4]), the type of a concept can be a set of concept types of T_C (called a *conjunctive type*). A concept c can thus be an instance of many distinct types (*e.g.*, $\epsilon(c) = \{\texttt{Protein},\texttt{Enzyme}\}$).

FOL Semantics. Supports and SGs can be translated into first order logic (FOL) to obtain a precise semantics for our syntactic objects. We consider concept types (resp. relation types of arity i) as *predicate names* of arity 1 (resp. of arity i), generic markers as *variables* and individual markers as *constants*. If t and t' are two predicate names of arity i, and $t \leq t'$, then their logical interpretation is the FOL formula $\phi(t, t') = \forall x_1 \cdots \forall x_i (t(x_1, \cdots, x_i) \to t'(x_1, \cdots, x_i))$. The logical interpretation of a support $\mathcal{S}$ is the formula $\Phi(\mathcal{S})$ obtained from the conjunction of the formulas $\phi(t, t')$, for all t, t' such that t' covers [1] t in $\mathcal{S}$.

[1] We say that t' covers t if $t \leq t'$ and there is no other t'' (apart from t and t') such that $t \leq t'' \leq t'$.

Let $G = (C, R, \gamma, \epsilon)$ be a SG. If c is a concept, we note $\phi(c)$ the conjunction of atoms $t(m)$, where $t \in type(c)$ and m is the marker of c. If r is a relation, we note $\phi(r) = t(m_1, \cdots, m_i)$ where $t = \epsilon(r)$ and, $\forall 1 \leq j \leq i$, m_j is the marker of $\gamma_j(r)$. We note $\phi(G) = \bigwedge_{c \in C} \phi(c) \wedge \bigwedge_{r \in R} \phi(r)$. Then the logical interpretation $\Phi(G)$ of G is the existential closure of $\phi(G)$.

Theorem 1. *Every $\mathcal{SG}$ KB $\mathcal{K} = (\mathcal{S}, G)$ is satisfiable.*

Reasonings. Though $\mathcal{SG}$-SATISFIABILITY is a trivial problem, $\mathcal{SG}$-DEDUCTION is an important problem that has been studied both inside and outside the CG community. Classically, $\mathcal{SG}$-DEDUCTION can be computed using a kind of graph homomorphism known as *projection*. It maps concepts having the same marker of the query Q to concepts of the SG G in the KB, while preserving the existence of relations and possibly decreasing labels, as allowed by the order relation defined in the support $\mathcal{S}$. We note $G \preceq_{\mathcal{S}} Q$ when there exists such a mapping. For more details on projection/homomorphism, as defined for simple CGs with conjunctive types, the reader can refer to [5]. Projection is a sound operation w.r.t. our FOL semantics, but to be complete, the SG G must be put into its *normal form nf*(G) (a semantically equivalent SG whose concepts have all different markers). Then:

Theorem 2 (Soundness and completeness). *Let $\mathcal{K} = (\mathcal{S}, G)$ be a $\mathcal{SG}$ KB, and Q be a SG. Then $\mathcal{K} \models Q \Leftrightarrow nf(G) \preceq_{\mathcal{S}} Q$.*

As HOMOMORPHISM, $\mathcal{SG}$-DEDUCTION is thus a NP-complete problem. By imposing some restrictions to the SG Q (*e.g.*, when Q admits a bound hypertree decomposition, see [5,6]), the problem becomes polynomial.

2.2 Adding Rules: The $\mathcal{SG}$ Language

Syntax. A $\mathcal{SR}$ KB is obtained by adding CG rules of form (hypothesis, conclusion) to a $\mathcal{SG}$ KB.

Definition 4 (CG rule). *A CG rule over a support $\mathcal{S}$ is a tuple $R = (H, C)$ where H and C are two SGs. $H = hyp(R)$ is called the hypothesis of the rule and $C = conc(R)$ its conclusion.*

FOL Semantics. The transformation Φ defined in SECT. 2.1 and can be extended to take CG rules into account. If $R = (H, C)$ is a CG rule, we note $\phi_H(C) = \exists x_1 \cdots \exists x_p \phi(C)$ where $x_1, \cdots, x_p$ are all variables of $\phi(C)$ that do not also appear in $\phi(H)$. Then we note $\phi(R) = \phi(H) \rightarrow \phi_H(C)$ and the logical interpretation $\Phi(R)$ of the rule R is the universal closure of $\phi(R)$. The interpretation $\Phi(\mathcal{R})$ of a set of CG rules $\mathcal{R}$ is the conjunction of the interpretations $\Phi(R)$, for all rules $R \in \mathcal{R}$.

Theorem 3. *Every $\mathcal{SR}$ KB $\mathcal{K} = (\mathcal{S}, G, \mathcal{R})$ is satisfiable.*

Reasonings. Rules increase the complexity of our reasonings: $\mathcal{SR}$-DEDUCTION is *semi-decidable* (if $\mathcal{K} \models Q$, then a sound and complete algorithm will stop, but no sound and complete algorithm is ensured to stop otherwise). [7] provides a sound and complete *forward chaining algorithm*. It relies upon the application of a CG rule $R = (H, C)$ to a SG G. R is said applicable if there is a projection, say π from H to G. In that case, the application of R to G following π produces a SG $\alpha(G, R, \pi)$ obtained by juxtaposing G and C, then for each concept of c whose marker also appears in a concept c' of H, by fusioning c with $\pi(c')$. Note that other generic markers of C have to be renamed (a safe substitution), and that $\alpha(G, R, \pi)$ must be put into its normal form.

If $\mathcal{R}$ is a set of rules, we note $\alpha_{\mathcal{S}}(G, \mathcal{R})$ the SG obtained by applying all rules in $\mathcal{R}$ to G following all the projections of their hypothesis. Then we define inductively $\alpha^i_{\mathcal{S}}$ by $\alpha^0_{\mathcal{S}}(G, \mathcal{R}) = nf(G)$ and $\forall 1 \leq i, \alpha^i_{\mathcal{S}}(G, \mathcal{R}) = \alpha_{\mathcal{S}}(\alpha^{i-1}_{\mathcal{S}}(G, \mathcal{R}), \mathcal{R})$.

Theorem 4 (Soundness and completeness). *Let $\mathcal{K} = (\mathcal{S}, G, \mathcal{R})$ be a $\mathcal{SR}$ KB, and Q be a SG. Then $\mathcal{K} \models Q \Leftrightarrow \exists i, \alpha^i_{\mathcal{S}}(G, \mathcal{R}) \preceq_{\mathcal{S}} Q$.*

If $\mathcal{K} = (\mathcal{S}, G, \mathcal{R})$, we note $\mathcal{K}^* = lim_{i \to \infty} \alpha^i_{\mathcal{S}}(G, \mathcal{R})$. Note that, in general, $\mathcal{K}^*$ is an *infinite* SG. To ensure that forward chaining stops, even when $\mathcal{K} \not\models Q$, [3] relies upon the notion of *finite expansion sets* of rules, ensuring that $\mathcal{K}^*$ is finite.

Definition 5 (Finite expansion set (f.e.s.)). *Let $\mathcal{S}$ be a support, and $\mathcal{R}$ be a set of rules. We say that $(\mathcal{S}, \mathcal{R})$ is a* finite expansion set (or f.e.s.) *iff for every $\mathcal{SR}$ KB $\mathcal{K} = (\mathcal{S}, G, \mathcal{R})$, $\mathcal{K}^*$ is finite.*

If $(\mathcal{S}, \mathcal{R})$ is a f.e.s., forward chaining is ensured to stop (when $\alpha^i_{\mathcal{S}}(G, \mathcal{R}) \equiv \alpha^{i+1}_{\mathcal{S}}(G, \mathcal{R}) \equiv \mathcal{K}^*$). Finding large subsets of rules that have the finite expansion property is thus an important task. [3] provides two examples of f.e.s.: *disconnected rules (d.r.)*, that share no generic marker in the hypothesis and the conclusion, and *range restricted rules (r.r.)*, where all generic markers of the conclusion are already in the hypothesis. In both cases, $\mathcal{SR}$-DEDUCTION is NP-complete. [8] introduced the notion of *rules dependencies* (R_2 depends upon R_1 when an application of R_1 can trigger an new application of R_2). When the graph encoding these dependencies has no circuit, then the set of rules is a f.e.s. More importantly, when all strongly connected components of this graph are f.e.s., then we also obtain a f.e.s.

2.3 Adding Negative Constraints: The Languages $\mathcal{SGC}^-$ and $\mathcal{SRC}^-$

Theorems 1 and 3 point out that all SGs and SG rules are satisfiable. However considering that every KB is satisfiable is not always realistic in practice. For example, in our application, we do not want an enzyme to be active and inhibited at the same time. Though various mechanisms have been proposed to introduce the notion of insatisfiability to conceptual graphs, we focus here on *negative constraints*.

Syntax. By enriching a KB of the $\mathcal{SG}$ (respectively $\mathcal{SR}$) language with *negative constraints*, we obtain a KB of the $\mathcal{SGC}^-$ (resp. $\mathcal{SRC}^-$) language. A negative constraint encodes that some knowledge must not be found in a graph.

Definition 6 (Negative constraint). *A* negative constraint*, defined over a support $\mathcal{S}$, is noted $N = \neg G$, where G is a SG over $\mathcal{S}$.*

FOL Semantics. The notation $\neg H$ stems from the semantic of negative constraints since the interpretation of $N = \neg G$ is defined by $\Phi(N) = \neg\Phi(G)$. If $\mathcal{N}$ is a set of negative constraints, then $\Phi(\mathcal{N})$ is the conjunction of all $\Phi(N)$, for $N \in \mathcal{N}$.

It is then possible with negative constraints to express cases of insatisfiability. For example, a KB containing the SG G of FIG. 1 as well as the negative constraint represented by the same FIG. is unsatisfiable.

Theorem 5 (Insatisfiability). *Let $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{N})$ be a $\mathcal{SRC}^-$ KB (it is a $\mathcal{SGC}^-$ when $\mathcal{R} = \emptyset$). Then $\mathcal{K}$ is unsatisfiable iff there exists $N = \neg C$ such that $(\mathcal{S}, G, \mathcal{R}) \models C$.*

$\mathcal{SGC}^-$-SATISFIABILITY is thus co-NP complete, and $\mathcal{SRC}^-$-SATISFIABILITY is truly undecidable (though $\mathcal{SRC}^-$-UNSATISFIABILITY is semi-decidable). The polynomial subclasses of SECT. 2.1 apply for $\mathcal{SGC}^-$-SATISFIABILITY while the decidable subclasses of SECT. 2.2 apply for $\mathcal{SRC}^-$-SATISFIABILITY.

Reasonings. Since negative constraints encode negative information and the query encodes positive formulae, negative constraints play no more role in reasonings when the KB is satisfiable.

Theorem 6 (Deduction). *Let $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{N})$ be a $\mathcal{SRC}^-$ KB, and Q be a SG. Then $\mathcal{K} \models Q$ iff $\mathcal{K}$ is unsatisfiable or $(\mathcal{S}, G, \mathcal{R}) \models Q$.*

$\mathcal{SGC}^-$-DEDUCTION is thus a NP-complete problem and $\mathcal{SRC}^-$-DEDUCTION is semi-decidable. As previously discussed, particular subclasses of SECT. 2.1 and SECT. 2.2 still apply.

3 Adding Defaults to Conceptual Graphs

3.1 The Need for Default Reasonings

In FIG 2 an agronomy application example is depicted: "if a pasta product undergoes a quick drying, then it is subject to cracking unless the drying is accompanied by vapor-injection". To deal with such non monotonic knowledge in the following we propose to introduce default reasoning in the CG model, in order to express rules that will be applied in the default case, i.e. unless they are a source of insatisfiability.

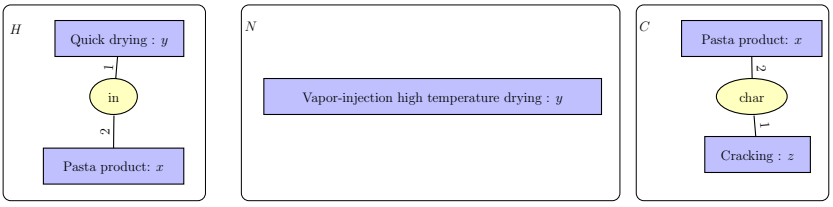

Fig. 2. An example of a default CG rule

3.2 Reiter's Default Logics

In this section we recall some basic definitions of Reiter's default logics [9,10]

Definition 7 (Reiter's default logic). *A Reiter's* default theory *is a pair* (Δ, W) *where* W *is a set of FOL formulae and* Δ *is a set of* defaults *of form* $\delta = \frac{\alpha(\overrightarrow{x}):\beta_1(\overrightarrow{x}),\cdots,\beta_n(\overrightarrow{x})}{\gamma(\overrightarrow{x})}$, $n \geq 0$, *where* $\overrightarrow{x} = (x_1, \cdots, x_k)$ *is a set of variables,* $\alpha(\overrightarrow{x})$, $\beta_i(\overrightarrow{x})$ *and* $\gamma(\overrightarrow{x})$ *are FOL formulae for which each free variable is in* $\overrightarrow{x}$.

The intuitive meaning of a default δ is "For all individuals $(x_1, \cdots, x_k)$, if $\alpha(\overrightarrow{x})$ is believed and each of $\beta_1(\overrightarrow{x}), \cdots, \beta_n(\overrightarrow{x})$ can be consistently believed, then one is allowed to believe $\gamma(\overrightarrow{x})$". $\alpha(\overrightarrow{x})$ is called the *prerequisite*, $\beta_i(\overrightarrow{x})$ are called the *justifications* and $\gamma(\overrightarrow{x})$ is called the *consequent*. A default is said to be *closed* if $\alpha(\overrightarrow{x})$, $\beta_i(\overrightarrow{x})$ and $\gamma(\overrightarrow{x})$ are all closed FOL formulae. A default theory (Δ, W) is said to be *closed* if all its defaults are closed. In this case we can omit the $\overrightarrow{x}$ notation.

Intuitively, an *extension* of a default theory (Δ, W) is a set of formulae that can be obtained from (Δ, W) while being consistently believed. More formally, an extension E of (Δ, W) is a minimal deductively closed set of formulae containing W such that for any $\frac{\alpha:\beta}{\gamma} \in \Delta$, if $a \in E$ and $\neg\beta \notin E$, then $\gamma \in E$.

The following theorem provides an equivalent characterization of extensions that we use here as a formal definition.

Theorem 7 (Extension). *Let* (Δ, W) *be a closed default theory and* E *be a set of closed FOL formulae. We inductively define* $E_0 = W$ *and for all* $i \geq 0$, $E_{i+1} = Th(E_i) \cup \{\gamma \mid \frac{\alpha:\beta_1\cdots,\beta_n}{\gamma} \in \Delta, \alpha \in E_i$ *and* $\neg\beta_1, \cdots, \neg\beta_n \notin E\}$[2].

Then E *is an extension of* (Δ, W) *iff* $E = \cup_{i=0}^{\infty} E_i$.

Note that extensions are only defined here for closed theories. In practice open defaults are transformed into the sets of their ground instances over the Herbrand universe.

Note also that this characterization is not effective for computational purposes since both E_i and $E = \cup_{i=0}^{\infty} E_i$ are required for computing E_{i+1}.

Some closed default theories can have no extension. It is for example the case of the default theory $(\Delta, W) = (\{\frac{\top:\beta}{\neg\beta}\}, \emptyset)$. However, normal default theories are ensured to have extensions.

[2] We note $Th(E_i)$ the deductive closure of E_i.

Definition 8 (Normal defaults). *A default is said* normal *if its consequent is semantically equivalent to the conjunction of its justifications. Defaults of form* $\delta = \frac{\alpha(\overrightarrow{x}):\beta(\overrightarrow{x})}{\beta(\overrightarrow{x})}$ *are normal.*

The meaning of a normal default is *if* α *is true and it is consistent to deduce* β*, then deduce* β*.*

Theorem 8. *Every closed normal default theory has an extension.*

Let us see a classical example of a default theory. Suppose that we want to model the knowledge that, in general, *birds fly, penguins are birds*, and *penguins do not fly*. Finally we add a penguin called *Tweety* in our knowledge base. This knowledge can be model by the following default theory :

$$(\Delta, W) = \left(\left\{\frac{p(x) : \neg f(x)}{\neg f(x)}, \frac{p(x) : b(x)}{b(x)}, \frac{b(x) : f(x)}{f(x)}\right\}, \{p(Tweety)\}\right)$$

where $b(x)$ means that the individual x is a bird, $f(x)$ means that x flies, and $p(x)$ means x is a penguin. Note that the knowledge *penguins are birds* have no known exception, and so a rule $\forall x, p(x) \rightarrow b(x)$ can be added to W instead of the default rule $\frac{p(x):b(x)}{b(x)}$ in D. This default theory can lead to 2 different extensions, which are $E_1 = Th(\{p(Tweety), b(Tweety), \neg f(Tweety)\})$ and $E_2 = Th(\{p(Tweety), b(Tweety), f(Tweety)\})$.

Some problems that must be addressed in Reiter's default logics are the following:

- EXTENSION: Given a default theory (Δ, W), does it have an extension?
- SKEPTICAL DEDUCTION: Given a default theory (Δ, W) and a formula Q, does Q belong to all extensions of (Δ, W)? In this case we note $(\Delta, W) \models_S Q$.
- CREDULOUS DEDUCTION: Given a default theory (Δ, W) and a formula Q, does Q belong to an extension of (Δ, W)? In this case we note $(\Delta, W) \models_C Q$?

In the previous example (Δ, W) admits two extensions. Both $f(Tweety)$ and $\neg f(Tweety)$ can be credulously deduced, but neither can be skeptically deduced.

Note that even when restricting these problems to closed normal default theories, the expressive power of FOL makes them undecidable.

3.3 Introducing Default CG Rules: The $\mathcal{SRDC}^-$ Language

Syntax. A KB of the $\mathcal{SRDC}^-$ language is obtained from a $\mathcal{SRC}^-$ KB enriched with default CG rules inspired by Reiter's defaults.

Definition 9 (Default CG rule). *A* default CG rule *over a support* $\mathcal{S}$ *is a tuple* $D = (H, N_1, \cdots, N_n, C)$*, with* $n \geq 0$*,* H *and* C *are SG's, and the* N_i *are negative constraints over* S*. As in Reiter's defaults, we call* H *the prerequisite,* N_i *the justifications, and* C *the consequent.*

Intuitively, such default means that "if H is believed, the negative constraints (justifications) are each satisfied, and *it is consistent to believe* C, then it is allowed to believe C".

Default Semantics. The interpretation of $\mathcal{SRDC}^-$ KB $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{N}, \mathcal{D})$ is a default theory $\Upsilon(\mathcal{K}) = (\Upsilon(\mathcal{D}), \Phi((\mathcal{S}, G, \mathcal{R}, \mathcal{N})))$ where Φ is the FOL interpretation of the KB as defined in SECT. 2, and $\Upsilon(\mathcal{D}) = \{\Upsilon(D), D \in \mathcal{D}\}$. The mapping Υ translates each default CG rule D into a default in Reiter's sense $\Upsilon(D)$ called the *default interpretation* of D.

Let $D = (H, N_1, \cdots, N_n, C)$ be a default CG rule, where $N_i = \neg G_i$. Its default interpretation $\Upsilon(D)$ is built as follows:

- Let $\overrightarrow{h}$ be the variables occurring in $\phi(H)$, $\overrightarrow{f}$ the variables occurring both in $\phi(C)$ and in $\phi(H)$, and $\overrightarrow{c}$ the variables occurring in $\phi(C)$ and not in $\phi(H)$.
- For $\zeta \in \{\phi(C), \phi(G_1), \cdots, \phi(G_n)\}$, the formula $sk(\zeta)$ is obtained by replacing for $c_i \in \overrightarrow{c}$, each occurrence of c_i by the functional term $f_i^D(\overrightarrow{f})$ in ζ.
- For $\xi \in \{sk(G_1), \cdots, sk(G_n)\}$, $sk^*(\xi)$ is obtained by existentially quantifying all variables of ξ that are not in $\overrightarrow{h}$. Finally:

$$\Upsilon(D) = \frac{\phi(H) : sk(C), \neg sk^*(G_1), \cdots, \neg sk^*(G_n)}{sk(C)}$$

Let us illustrate this by the transformation of the default CG rule $D = (H, N, C)$ of FIG. 2. In the next equation, $QD(x)$ means that product x undergoes a *quick drying*, $P(x)$ signifies that x is a *pasta product*, $C(x)$ signifies the *Cracking* property of pasta and $VIHTD(y)$ specifies a *vapor-injection high temperature drying* y. While at the representation level the formula below has the same meaning as FIG. 2, the authors consider that FIG. 2 conveys its meaning in a more intuitive manner.

$$\Upsilon(D) = \frac{QD(y) \wedge in(y,x) \wedge P(x) : P(x) \wedge char(f_1^D(y,x), x) \wedge C(f_1^D(y,x)), VIHTD(y)}{P(x) \wedge char(f_1^D(y,x), x) \wedge C(f_1^D(y,x))}$$

The problems defined in Reiter's default logics are easily recast in $\mathcal{SRDC}^-$:

- $\mathcal{SRDC}^-$-EXTENSION: Given a $\mathcal{SRDC}^-$ KB $\mathcal{K}$, does $\Upsilon(\mathcal{K})$ have an extension?
- $\mathcal{SRDC}^-$-SKEPTICAL DEDUCTION: Given a $\mathcal{SRDC}^-$ KB $\mathcal{K}$ and a SG Q, does $\Upsilon(\mathcal{K}) \models_S \Phi(Q)$? In this case we note $\mathcal{K} \models_S Q$.
- $\mathcal{SRDC}^-$-CREDULOUS DEDUCTION: Given a $\mathcal{SRDC}^-$ KB $\mathcal{K}$ and a SG Q, does $\Upsilon(\mathcal{K}) \models_C \Phi(Q)$? In this case we note $\mathcal{K} \models_C Q$.

Modeling Choices. Two features of our chosen semantics might seem surprising to the reader. First, the presence of $sk(C)$ as an added justification. This is due to the fact that we need to be able to represent normal defaults in our language (if a default rule $D = (H, C)$ has no negative constraint then $\Upsilon(D)$ is a normal default). Second, we have introduced functional terms in the interpretation of a default. This is due to the fact that the default interpretation is composed of many formulae and functional terms are the only way to link up the variables of these formulae.

4 Reasoning in $\mathcal{SRDC}^-$

4.1 The Defaults Derivation Tree (d.d.t.)

The defaults derivation tree (d.d.t.) of a $\mathcal{SRDC}^-$ KB $\mathcal{K} = (\mathcal{S},G,\mathcal{R},\mathcal{N},\mathcal{D})$ is a rooted, labeled and possibly infinite tree $ddt(\mathcal{K})$ used as a tool to compute extensions. To define this tree, we need new objects generalizing negative constraints, that we call attached constraints.

Attached Constraints. Let G be a SG. A constraint attached to G is a pair (A, μ) where A is a SG and μ is a partial mapping from the concepts of A to the concepts of G. We say that G *violates* (A, μ) iff there exists a projection π from A into G such that π extends μ. Otherwise G *satisfies* (A, μ).

Note that attached constraints generalize negative constraints (the latter occurs in the case of $\mu = \emptyset$). A $\mathcal{SR}$ KB $\mathcal{K} = (\mathcal{S}, G, \mathcal{R})$ violates a constraint (A, μ) attached to G iff there exists $i \geq 0$ and a projection π from A to $\alpha^i_{\mathcal{S}}(G, \mathcal{R})$ such that π extends μ. It satisfies (A, μ) otherwise. $\mathcal{K}$ violates a set $\mathcal{A}$ of constraints attached to G iff it violates one $(A, \mu) \in \mathcal{A}$. It satisfies $\mathcal{A}$ otherwise. The complexity of computing satisfiability with attached constraints remains the same as for negative constraints.

Note that if (A, μ) is a constraint attached to G and G' is a SG containing G (such as a SG obtained by applying rules on G), then we can consider (A, μ) as a constraint attached to G'. In the same way, many algorithms rely on finding a smaller equivalent SG G' by fusioning concepts of the SG G. Then, for every (A, μ) attached to G, we attach a constraint (A, μ') to G' such that if there is a concept c in A such that $\mu(c)$ has been fusioned into c' in G', then $\mu'(c) = c'$, and $\mu'(c) = c$ otherwise.

Vertices of the d.d.t. The d.d.t. intuitively represents a kind of derivation tree. Each node v is labeled by $\lambda(v) = (G_v, \mathcal{A}_v)$. G_v represents a state of knowledge derived from the initial KB and $\mathcal{A}_v$ represents the suppositions that we made to derive G_v. For example, consider the application of the default CG rule represented in FIG. 2 on a pasta product A which undergoes a quick drying Q. To conclude that A is subject to cracking, we need to suppose (and remember in $\mathcal{A}_v$ for further derivation) that Q is not a *vapor-injection high temperature drying*. To remember this ensures that no further derivation can conclude that Q was a *vapor-injection high temperature drying*.

A vertex v of $ddt(\mathcal{K})$ is labeled by $\lambda(v) = (G_v, \mathcal{A}_v)$ where G_v is a SG and $\mathcal{A}_v$ is a set of constraints attached to G_v. The root r of $ddt(K)$ is labeled by $\lambda(r) = (G, \emptyset)$. A vertex v of $ddt(K)$ is *satisfiable* iff $(\mathcal{S}, G_v, \mathcal{R}, \mathcal{N})$ is satisfiable and $(\mathcal{S}, G_v, \mathcal{R})$ satisfies $\mathcal{A}_v$.

If v is satisfiable, then for each $D = (H, N_1, \cdots, N_n, C)$, for each projection π into some $G' = \alpha^i_{\mathcal{S}}(G_v, \mathcal{R})$, if π is not "blocked" v admits a child $v' = \delta(v, \pi)$.

Let us consider the SG $G'' = \alpha(G', (H, C), \pi)$. For each justification N_k, we build the constraint (N_k, μ_k) attached to G'' where μ_k is defined as follows: if c is a concept of N_k whose generic marker appears in a node c' of H then $\mu_k(c) = \pi(c')$. Otherwise, if this marker appears in a node c' of C then $\mu_k(c)$ is a

concept obtained from a copy of c' in G''. We note $\mathcal{A}'_v = \mathcal{A}_v \cup \{A_k\}_{1 \leq k \leq n}$. Finally π is *blocked* iff there exists $j \geq i$ such that $\alpha(\alpha^j_{\mathcal{S}}(G_v, \mathcal{R}), (H, C), \pi)$ violates $\mathcal{A}'_v$. If π is not blocked, then $\lambda(v') = (G'', \mathcal{A}'_v)$.

Building Finite d.d.t. Given the expressive power of SG rules, $ddt(\mathcal{K})$ is an infinite tree: it can have an infinite depth and each vertex can have an infinite number of children. To be able to finitely build d.d.t., let us now extend the notion of finite expansion sets (SECT. 2.2).

Definition 10 (Finite expansion property). *If $D = (H, N_1, \cdots, N_n, C)$ is a default CG rule, we note $fol(D) = (H, C)$ its associated SG rule. If $\mathcal{D}$ is a set of default CG rules, we note $fol(\mathcal{D}) = \{fol(D)\}_{D \in \mathcal{D}}$. Then a $\mathcal{SRDC}^-$ KB $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{N}, \mathcal{D})$ is said to have the* finite expansion property *iff $\mathcal{R} \cup fol(\mathcal{D})$ is a finite expansion set.*

If $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{N}, \mathcal{D})$ has a finite expansion property, then for every vertex v of $ddt(\mathcal{K})$, with $\lambda(v) = (G_v, \mathcal{A}_v)$, G_v is a subgraph of the finite SG $(\mathcal{S}, G, \mathcal{R} \cup fol(\mathcal{D}))^*$. Then, since the graph G_v labeling each vertex v is bigger than the graph labeling is parent, the depth of $ddt\mathcal{K}$ is finite. And since $(\mathcal{S}, G_v, \mathcal{R})^*$ is finite, there is a finite number of projections of the defaults in it, so the number of children of v is finite, and its satisfiability can be computed in finite time. It follows that:

Theorem 9. *If a $\mathcal{SRDC}^-$ KB $\mathcal{K}$ has the finite expansion property then $ddt(\mathcal{K})$ can be computed in finite time.*

4.2 Sound and Complete Reasoning w.r.t. Υ

Let us now show that the d.d.t. can be used for sound and complete reasonings in $\mathcal{SRDC}^-$.

Theorem 10. *Let $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{N}, \mathcal{D})$ be a $\mathcal{SRDC}^-$ KB, and Q be a SG. Then either $(\mathcal{S}, G, \mathcal{R}, \mathcal{N})$ is unsatisfiable or the following assertions are equivalent:*

i There exists an extension E of $\Upsilon(\mathcal{K})$ such that $E \models \Phi(Q)$.
ii There exists a satisfiable leaf v of $ddt(\mathcal{K})$ with $\lambda(v) = (G_v, \mathcal{A}_v)$ such that $(\mathcal{S}, G_v, \mathcal{R})$ models$Q$.

Due to space requirements the proof of this theorem is omitted in this paper.

It follows that:

Theorem 11 (Soundness and completeness). *Let $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{N}, \mathcal{D})$ be a $\mathcal{SRDC}^-$ KB, and Q be a SG. Then $\mathcal{K} \models_S Q$ (resp. $\mathcal{K} \models_C Q$) iff either $ddt(\mathcal{K})$ has a unique unsatisfiable vertex, or, for all satisfiable leaves, (resp. there exists one satisfiable leaf) v in $ddt(\mathcal{K})$ with $\lambda v = (G_v, \mathcal{A}_v)$, $(\mathcal{S}, G_v, \mathcal{R}) \models Q$.*

This latter theorem provides us with an effective characterization of the deduction problems in $\mathcal{SRDC}^-$. Thanks to THM. 9, this characterization also provides a halting algorithm when $\mathcal{K}$ has the finite expansion property.

4.3 Relationship with $\mathcal{SREC}$

The Language $\mathcal{SREC}$. [3] presents a family of CG languages. The most expressive one in this language hierarchy is the language $\mathcal{SREC}$. In this language a KB $\mathcal{K}$ is composed of a support $\mathcal{S}$, a SG G, a set $\mathcal{R}$ of *inference rules* (that behave exactly as CG rules), a set $\mathcal{E}$ of *evolution rules* of form (H, C), and a set $\mathcal{C}$ of constraints. By restricting constraints to the negative constraints presented here we obtain the language $\mathcal{SREC}^-$.

Reasoning in $\mathcal{SREC}^-$. Reasonings in $\mathcal{SREC}^-$ rely upon building a "tree of possible worlds", akin to the d.d.t. presented in this paper. Since $\mathcal{SREC}^-$ does not dynamically generate constraints, a possible world is only labeled by a SG. Children of a possible world are generated as if we considered each evolution rule as a default rule without justification. Finally, an answer to a SG Q can be found in any possible world, not only in the leaves as done in $\mathcal{SRDC}^-$. However, default rules translating evolution rules are normal, and thus any possible world is an ancestor or an extension (THM. 8). Therefore, if an answer to Q can be found in a possible world v, the same answer can be found in all leaves/extensions having v as an ancestor.

Default Semantics for $\mathcal{SREC}^-$. By comparing the reasonings in $\mathcal{SREC}^-$ and $\mathcal{SRDC}^-$ we obtain an interesting equivalence result that provides the formally lacking semantics of $\mathcal{SREC}^-$. Let us consider the bijection τ from $\mathcal{SREC}^-$ KBs to $\mathcal{SRDC}^-$ KBs that transforms each evolution rule into a default CG rule without justification (*i.e.* a normal default CG rule).

Theorem 12. *Let $\mathcal{K} = (\mathcal{S}, G, \mathcal{R}, \mathcal{E}, \mathcal{N})$ be a $\mathcal{SREC}^-$ KB, and Q be a SG. Then $(\mathcal{S}, G, \mathcal{R}, \tau(\mathcal{E}), \mathcal{N}) \models_C Q$ iff $(\mathcal{S}, G, \mathcal{R}, \mathcal{N})$ is unsatisfiable or $\mathcal{K} \models Q$ ($\models$ being the deduction used in $\mathcal{SREC}^-$).*

We can finally provide a logical semantics $\Upsilon_{\mathcal{E}}$ to the $\mathcal{SREC}^-$ language, by defining:

$$\Upsilon_{\mathcal{E}}((\mathcal{S}, G, \mathcal{R}, \mathcal{E}, \mathcal{N})) = \begin{cases} (\bot, \emptyset) \text{ if } (\mathcal{S}, G, \mathcal{R}, \mathcal{N}) \text{ is unsatisfiable} \\ \Upsilon((\mathcal{S}, G, \mathcal{R}, \tau(\mathcal{E}), \mathcal{N})) \text{ otherwise.} \end{cases}$$

$\mathcal{SREC}^-$ is thus the subset of $\mathcal{SRDC}^-$ restricted to normal defaults, and:

Theorem 13. *Deduction in $\mathcal{SREC}^-$ is sound and complete with respect to credulous deduction according to the $\Upsilon_{\mathcal{E}}$ semantics.*

5 Conclusion and Perspectives

In this paper we have formally defined the syntax and semantics of a new language of the $\mathcal{SG}$ family, namely the $\mathcal{SRDC}^-$ language. This extension was necessary in the agronomy application we are involved in, and the semantics of this language are expressed in Reiter's default logics. Since this subset of default logics is built upon a particular subset of FOL, we were able to provide a

constructive characterization of Reiter's extensions (THM. 10). Using the finite expansion sets that form a decidable subclass of $\mathcal{SR}$, we defined a new decidable subclass of Reiter's default logics (THM. 9). Finally, we showed that the $\mathcal{SREC}^-$ language of [3] is a strict subclass of $\mathcal{SRDC}^-$, and provided a formerly lacking default logic semantics for that language.

Some problems are still to be addressed to be able to encode the knowledge required by our application and to compute deductions in an efficient way:

Functional Relations. For more precise reasonings we need to be able to represent numerical information in a knowledge base and to express functional constraints such as the following rule given by a domain expert: a high temperature for drying has to be above Naples average spring temperature. [11] extends the language $\mathcal{SR}$ to handle such knowledge. This language could provide the foundations for a functional extension of $\mathcal{SRDC}^-$.

Other Decidable Subclasses of CG Rules. The KB obtained from our preliminary modeling has the finite expansion property that ensures finite reasonings. It may be possible that with the introduction of new knowledge this property no longer holds. It would then be essential to investigate other kinds of decidable KBs. An interesting research direction could be to extend other kinds of decidable subclasses of $\mathcal{SR}$ to $\mathcal{SRDC}^-$. Such decidable subclasses could be finite unification sets (that ensure a finite backward chaining rewriting) or a bounded treewidth sets (a strict generalization of f.e.s. ensuring that $\mathcal{K}^*$ has a bounded treewidth)[12].

Reasoning with Preferences. Default logics can be extended to take defaults preferences into account. In this model, one can define an order (partial or total) on the set of defaults. Our default CG rule model provides a natural order on defaults: a default CG rule D_1 should be preferred to a default D_2 if the prerequisite of D_1 is a specialization of the prerequisite of D_2. This is exactly what is intuitively needed in our agronomy scenario. The consequent problems of computing extensions are then transformed into finding the most preferred extensions [10]. Even when defaults are totally ordered, the procedure that chooses the application of the most preferred unblocked default at each vertex of the d.d.t., is not ensured to lead to a preferred extension. Formally defining and finding preferred extensions is left for further work.

References

1. Abécassis, J.: Qualité du blé dur de la semoule et des pâtes alimentaires, 7–11 (1991)
2. Faron, C., Ganascia, J.: Representation of defaults and exceptions in conceptual graphs formalism. In: Delugach, H.S., Keeler, M.A., Searle, L., Lukose, D., Sowa, J.F. (eds.) ICCS 1997. LNCS, vol. 1257, pp. 153–167. Springer, Heidelberg (1997)
3. Baget, J.F., Mugnier, M.L.: Extensions of Simple Conceptual Graphs: the Complexity of Rules and Constraints. Jour. of Artificial Intelligence Research 16, 425–465 (2002)

4. Baget, J.F., Mugnier, M.L.: The sg family: Extensions of simple conceptual graphs. In: Proc of Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, pp. 205–212. Morgan Kaufmann, San Francisco (2001)
5. Baget, J.F.: Simple Conceptual Graphs Revisited: Hypergraphs and Conjunctive Types for Efficient Projection Algorithms. In: Ganter, B., de Moor, A., Lex, W. (eds.) ICCS 2003. LNCS, vol. 2746, pp. 229–242. Springer, Heidelberg (2003)
6. Gottlob, G., Leone, N., Scarcello, F.: A comparison of structural csp decomposition methods. Artificial Intelligence 124 (2000)
7. Salvat, E., Mugnier, M.L.: Sound and complete forward and backward chainingd of graph rules. In: Eklund, P., Mann, G.A., Ellis, G. (eds.) ICCS 1996. LNCS, vol. 1115. Springer, Heidelberg (1996)
8. Baget, J.F.: Improving the forward chaining algorithm for conceptual graphs rules. In: Proceedings of the Ninth International Conference (KR 2004), Whistler, Canada, pp. 407–414. AAAI, Menlo Park (2004)
9. Reiter, R.: A logic for default reasoning. Artificial Intelligence 13, 81–132 (1980)
10. Brewka, G., Eiter, T.: Prioritizing default logic: Abridged report. In: Festschrift on the occasion of Prof.Dr. W. Bibel's 60th birthday. Kluwer, Dordrecht (1999)
11. Baget, J.F.: A datatype extension for simple conceptual graphs and conceptual graphs rules. In: Proc of ICCS 2007: Conceptual Structures: Knowledge Architectures for Smart Applications. LNCS, vol. 4604, pp. 83–96. Springer, Heidelberg (2007)
12. Baget, J.F., Leclère, M., Mugnier, M.L., Salvat, E.: Extending decidable cases for rules with existential variables. IJCAI 2009 (submitted, 2009)