A Graph-Based Knowledge Representation Language for Concept Description

Alexandre Delteil[1] and **Catherine Faron**[2]

Abstract. In this paper, we propose an expressive concept description language, $\mathcal{GDL}$, for real world applications combining features of both Conceptual Graphs (CGs) and Description Logics (DLs). Regarding concept descriptions in CGs, namely existential, positive and conjunctive graphs, $\mathcal{GDL}$ is the closure of this language under the Boolean operations. Now regarding DLs, $\mathcal{GDL}$ is an extension of $\mathcal{ALC}$ with graph structures in concept descriptions.
$\mathcal{GDL}$ extends $\mathcal{ALC}$ with the intersection $\sqcap$, composition $\circ$, converse of roles $\cdot^-$ and role identity $id(.)$: we show how these constructs can be expressed with EG and GR constructs.
We provide a sound and complete tableaux algorithm to prove the satisfiability of $\mathcal{GDL}$ in NEXPTIME.

1 INTRODUCTION

Conceptual graphs (CG) is a knowledge representation model descending from existential graphs [8] and semantic networks [6]. Description Logics (DLs) are a family of knowledge representation formalisms focusing on concept description and reasoning about these concepts, that provide effective reasoning procedures for fragments of FOL. Previous works have attempted to state a correspondance between these two formalisms [4] [1]. In this paper, we propose a concept description language ($\mathcal{GDL}$) which combines features of both Conceptual Graphs (CGs) and Description Logics (DLs). Regarding concept descriptions in CGs, namely existential, positive and conjunctive graphs, $\mathcal{GDL}$ is the closure of this language under the Boolean operations. Now regarding DLs, $\mathcal{GDL}$ is an extension of $\mathcal{ALC}$ with graph structures in concept descriptions.
This extension is achieved by generalizing the *existential restriction* and *universal restriction* constructs of $\mathcal{ALC}$ into respectively an *Existential Graph* (EG) construct and a *Graph Rule* (GR) construct. The EG construct enables to introduce graph structures in concept descriptions and the GR construct can be viewed as the dual of the EG construct: the negation of a graph rule concept can be expressed with existential graph concepts, the negation of an existential graph concept with a graph rule concept.
$\mathcal{GDL}$ is thus an expressive concept description language for real world applications: our work is driven by the modeling needs of real world applications like in bioinformatics, chemical engeneering or cartography, which are domains characterized by structured data.
In section 2 and 3, we present concept descriptions in DLs and in CGs. In Section 4, we describe our concept description language $\mathcal{GDL}$. In Section 5, we provide a tableaux algorithm to prove the satisfiability of $\mathcal{GDL}$. In Section 6, we prove our algorithm is sound and complete and terminates in NEXPTIME.

[1] INRIA, BP 93, 06902 Sophia Antipolis, France, adelteil@sophia.inria.fr
[2] I3S, BP 145, 06903 Sophia Antipolis, France, faron@essi.fr

2 CONCEPT DESCRIPTION IN DLs

A DL includes a terminological language and an assertional language. The assertional language is dedicated to the statement of facts and the terminological language to the description of concepts and roles. The fundamental reasoning task is the computation of subsumption relations between concepts.
A DL is inductively defined from a set P_c of primitives concepts, a set P_r of primitive roles, the constant concepts $\top$ and $\bot$, and two abstract syntax rules:

C, D	$\rightarrow$	$\top \mid \bot$	most general | absurd
		P	primitive concept
		$C \sqcup D$	concept disjunction
		$C \sqcap D$	concept conjunction
		$\neg C$	negation
		$\forall r.C$	universal restriction on roles
		$\exists r.C$	existential restriction on roles
r	$\rightarrow$	q	primitive role
		q^{-1}	role converse
		$r_1 \circ r_2$	role composition

The constructors used in these syntax rules determine the expressive power of the DL being defined.
For example, the following concept description describes all the highways or interstates whose at least one crossing road has a redlight:

$$(Highway \sqcup Interstate) \sqcap \exists crosses.(Road \sqcap \exists on^{-1}.RedLight).$$

The formal meaning of a concept description is classicaly given through an extensional semantics by an interpretation $\mathcal{I}$ which is a pair $(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$. The domain $\Delta^{\mathcal{I}}$ is an arbitrary non-empty set of individuals and $\cdot^{\mathcal{I}}$ is an interpation function mapping each concept with a subset of $\Delta^{\mathcal{I}}$ and each role with a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. The denotation of a concept of the DL defined above is given in Table 1.

An interpretation $\mathcal{I}$ is a model for a concept C if $C^{\mathcal{I}}$ is non-empty. C is *satisfiable* iff there exists an interpretation $\mathcal{I}$ which is a model of C. Based on this semantics, C is subsumed by D, noted $C \sqsubseteq D$, iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every interpretation $\mathcal{I}$.

3 CONCEPT DESCRIPTION IN CGs

3.1 Basic Notions on Conceptual Graphs

A simple conceptual graph is a bipartite (not necessarily connected) graph composed of concept nodes and relation nodes describing relations between these concepts. Each concept node c of a graph G

Construct name	Syntax	Semantics
top / bottom	$\top$ / $\bot$	$\Delta^{\mathcal{I}}$ / $\emptyset$
disjunction	$\mathcal{C} \sqcup \mathcal{D}$	$\mathcal{C}^{\mathcal{I}} \cup \mathcal{D}^{\mathcal{I}}$
conjunction	$\mathcal{C} \sqcap \mathcal{D}$	$\mathcal{C}^{\mathcal{I}} \cap \mathcal{D}^{\mathcal{I}}$
negation	$\neg\mathcal{C}$	$\Delta^{\mathcal{I}} \setminus \mathcal{C}^{\mathcal{I}}$
universal restriction	$\forall r.C$	$\{a \in \Delta^{\mathcal{I}} / \forall b \in \Delta^{\mathcal{I}}$, if $(a,b) \in r^{\mathcal{I}}$ then $b \in \mathcal{C}^{\mathcal{I}}\}$
existential restriction	$\exists r.C$	$\{a \in \Delta^{\mathcal{I}} / \exists b \in \Delta^{\mathcal{I}}$, $(a,b) \in r^{\mathcal{I}}$ and $b \in \mathcal{C}^{\mathcal{I}}\}$
role converse	r^{-1}	$\{(a,b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} / (b,a) \in r^{\mathcal{I}}\}$
role composition	$r_1 \circ r_2$	$\{(a,b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} / \exists c \in \Delta^{\mathcal{I}}$, $(a,c) \in r_1^{\mathcal{I}}$ and $(c,b) \in r_2^{\mathcal{I}}\}$

Table 1. Syntax and Semantics of DL Concept Descriptions

is labeled by a couple $(type(c), ref(c))$, where $ref(c)$ is either the generic marker $*$ corresponding to the existential quantification or an individual marker corresponding to an identifier; M is the set of all the individual markers. Each relation node r of a graph G is labeled by a relation type $type(r)$; each relation type is associated with a signature expressing constraints on the types of the concepts that may be linked to its arcs in a graph. Figure 1 presents an example of a simple conceptual graph.

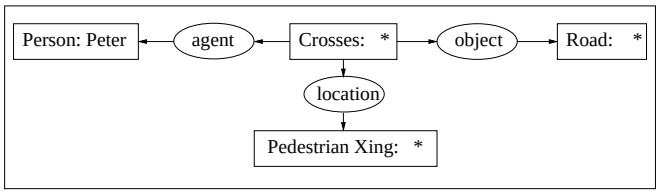

Figure 1. An example of conceptual graph

Concept types (respectively relation types of same arity) build up a set T_C (resp. T_R) partially ordered by a generalization/specialization relation $\leq_c$ (resp. $\leq_r$). (T_C, T_R, M) defines the *support* upon which conceptual graphs are constructed. A support thus represents the ontological knowledge.

For a further detailed presentation CGs, the reader is invited to refer to [10][3].

3.2 Logical Semantics

The semantics of CGs relies on the translation of both conceptual graphs and their support into first order logic (FOL) formulas. A generalization relation between two concept types t_1 and t_2 of a support is translated into the following FOL formula: $\forall x P_1(x) \Rightarrow P_2(x)$ where P_1 and P_2 are two unary predicates. A generalization relation between two n-ary relation types t_1 and t_2 of a support is translated into the following FOL formula:
$\forall x_1 \ldots \forall x_n \; P_1(x_1, \ldots, x_n) \Rightarrow P_2(x_1, \ldots, x_n)$ where P_1 and P_2 are two n-ary predicates.

A conceptual graph G is translated into FOL thanks to the Φ operator defined in [10]: $\Phi(G)$ is a conjunction of unary predicates translating the concept nodes of G and n-ary predicates translating the n-ary relation nodes of G; an existential quantification is introduced for each generic concept. The semantics of the conceptual graph presented in Figure 1 is given by the following FOL formula:
$\Phi(G) = \exists x, y, z \, Person(Peter) \wedge Crosses(x) \wedge Road(y) \wedge PedestrianXing(z) \wedge agent(x, Peter) \wedge object(x, y) \wedge location(x, z)$.

3.3 Reasoning with Conceptual Graphs

Conceptual graphs are provided with a subsumption relation $\leq_G$ corresponding to the logical implication:
$G_1 \leq G_2$ iff $\Phi(G_1) \Rightarrow \Phi(G_2)$.
A key operation called *projection* enables to compute subsumption relations between graphs: $G_1 \leq G_2$ iff there exists a projection π from G_2 to G_1. π is a graph morphism such that the label of a node n_1 of G_1 is a specialization of the label of a node n_2 of G_2 with $n_1 = \pi(n_2)$. Reasoning with conceptual graphs is based on the projection which is sound and complete with respect to logical deduction. Finding a projection between two graphs is a NP-complete problem [3].

3.4 Concept Description

Like DLs, CGs supports concept descriptions through a type definition mechanism [10]. It consists in associating a formal description to a type, this description being a conceptual graph.
Formally, a concept type definition $t_c(x) \equiv \lambda x G$ asserts an equivalence between a concept type t_c and a monadic abstraction $\lambda x G$ which consists of a conceptual graph G whose concepts are either atomic or defined and whose one concept among its generic ones is designated by the formal parameter x. Reasoning with type definition is detailed in [5]. Figure 2 presents the definition of the concept type *crossroad*.
Similarly, n-ary relation types t_r can be defined by n-ary abstractions $\lambda x_1, \ldots, x_n G$.
The Φ interpretation function of CGs is extended to type definition: $\Phi(t_c)$ is the FOL formula obtained from $\Phi(G)$ by removing the existential closure of the variable x. $\Phi(t_c)$ thus has one single free variable.

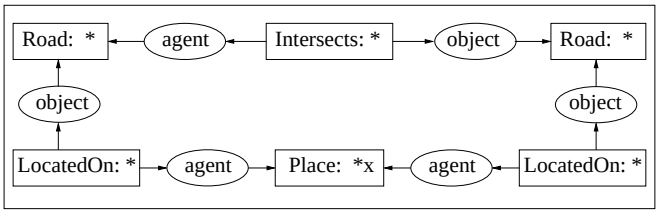

Figure 2. An example of concept type defi nition

When compared to concept descriptions in DLs, this type definition mechanism based on positive and conjunctive existential graphs lacks of the expressiveness the negation and disjunction constructs provide to most DLs. On the other hand, the *graph* structure of CGs provides a higher expressiveness than the *tree* structure of most DLs. We thus propose to merge both features into a graph based language for concept description. Previous works have attempted to state a correspondance between DLs and CGs [4] [1]. Here we take advantage of both formalisms and build upon CGs with DL constructs.

4 GDL: A GRAPH DESCRIPTION LANGUAGE

In this section, we propose a concept description language based on graphs and DL constructs. We call it $\mathcal{GDL}$ for *Graph Description*

Logic. Regarding concept descriptions in CGs, namely existential, positive and conjunctive graphs, $\mathcal{GDL}$ can be viewed as the closure of this language under the Boolean operations. Now regarding most DLs, they are provided with Boolean constructs and $\mathcal{GDL}$ can be viewed as an extension of the $\mathcal{ALC}$ DL to allow graph structures in concept descriptions. This extension is achieved by generalizing the *existential restriction* and *universal restriction* constructs of $\mathcal{ALC}$.

Formally, $\mathcal{GDL}$ is inductively defined from a set P_c of primitive concepts, a set P_r of primitive roles and the concepts $\top$ and $\bot$, by the following abstract syntax rule:

C, D	$\rightarrow$	$\top \mid \bot$	most general | absurd
		P	primitive concept
		$\neg C$	negation
		$C \sqcap D$	concept conjunction
		$C \sqcup D$	concept disjunction
		$\lambda x G$	existential graph
		$\lambda x R$	graph rule

In addition to the *conjunction*, *disjunction* and *negation* constructs, the *existential graph* construct enables to introduce graph structures in concept definitions.

Definition 1 *An existential graph $\lambda x_0 G$ is a concept description consisting of a connected graph G whose concept nodes are all generic, either atomic or defined by a concept description of $\mathcal{GDL}$. One of its concept nodes is designated by the formal parameter x_0.*

The *graph rule* construct is the dual of the *existential graph* construct: the negation of a graph rule concept can be expressed with existential graph concepts and the negation of an existential graph concept with a graph rule concept.

Definition 2 *A graph rule $\lambda x_0 R \equiv \lambda x_0 (G \Rightarrow C_0 \dots C_n)$ is a concept description which consists of a pair of abstractions $(\lambda x_0 x_1 \dots x_n G, \lambda x_0 x_1 \dots x_n [C_0 : x_0] \dots [C_n : x_n])$ where:*

- *$\lambda x_0 x_1 \dots x_n G$ is called the hypothesis of the graph rule $\lambda x_0 R$. It is a connected graph whose $n+1$ concept nodes are designated by the formal parameters $x_0, x_1, \dots, x_n$. The concepts of G are all generic, either atomic or defined by a concept of $\mathcal{GDL}$.*
- *$\lambda x_0 x_1 \dots x_n [C_0 : x_0] \dots [C_n : x_n]$ is the conclusion of the graph rule. Each C_i in the conclusion is a concept description of $\mathcal{GDL}$.*
- *$x_0, x_1, \dots, x_n$ are called connection points; they correspond to n coreference links between G and $[C_i : *]_{i=1..n}$, indicating that the generic marker of each C_i represents the same entity as one concept of G.*

The hypothesis and the conclusion of a graph rule have the same number of concepts ($n+1$). If a concept in the hypothesis of the rule - designated by a formal parameter x_i - has no corresponding concept C_i in the conclusion of the rule, the most general concept $\top$ is added for C_i.

The duality between the *existential graph* construct and the *graph rule* constructs is formally stated in the following two lemmas.

Lemma 1 *If $C \equiv \lambda x_0 G$ then $\neg C \equiv \lambda x_0 R$ where $\lambda x_0 R$ is the couple $(\lambda x_0 x_1 \dots x_n G, \lambda x_0 x_1 \dots x_n [\bot : x_0] \dots [\bot : x_n])$. The hypothesis of the rule is the graph G of the initial existential graph; the conclusion of the rule is the conjunction of absurd concepts.*

Lemma 2 *If $C \equiv \lambda x_0 R$ where $\lambda x_0 R$ is a couple $(\lambda x_0 x_1 \dots x_n G, \lambda x_0 x_1 \dots x_n [C_0 : x_0] \dots [C_n : x_n])$, then $\neg C \equiv \bigsqcup_k \lambda x_0 G_k$ where each G_k is the graph obtained from G by replacing one of its nodes $[D_k : x_k]$ by the node $[\neg C_k \sqcap D_k : x_k]$.*

These two lemma are the key to put a $\mathcal{GDL}$ concept description in negation normal form (NNF). A concept description is put in NNF by pushing down the negations to primitive concepts, applying lemma 1 in case of negating an existential graph and lemma 2 in case of negating a graph rule.

The formal meaning of $\mathcal{GDL}$ concept descriptions is given as an extensional semantics by an interpretation $\mathcal{I}$ as described in Table 2.

Construct	Syntax	Semantics
top / bottom	$\top$ / $\bot$	$\Delta^{\mathcal{I}}$ / $\emptyset$
disjunction	$\mathcal{C} \sqcup \mathcal{D}$	$\mathcal{C}^{\mathcal{I}} \cup \mathcal{D}^{\mathcal{I}}$
conjunction	$\mathcal{C} \sqcap \mathcal{D}$	$\mathcal{C}^{\mathcal{I}} \cap \mathcal{D}^{\mathcal{I}}$
negation	$\neg \mathcal{C}$	$\Delta^{\mathcal{I}} \setminus \mathcal{C}^{\mathcal{I}}$
existential graph	$\lambda x_0 G$	$\{b_0 \in \Delta^{\mathcal{I}} / \exists b_1 \in \Delta^{\mathcal{I}}, \dots, \exists b_n \in \Delta^{\mathcal{I}}$ not necessarily distinct, with $n+1$ the number of nodes of G in normal form, such that: - if $[\top : x_i] - r - [\top : x_j]$ is an arc of G then $(b_i, b_j) \in r^{\mathcal{I}}$, - if $[C : x_i]$ is a node in G then $b_i \in C^{\mathcal{I}}\}$
graph rule	$\lambda x_0 R$	$\{b_0 \in \Delta^{\mathcal{I}} / \forall b_1 \in \Delta^{\mathcal{I}}, \dots, \forall b_n \in \Delta^{\mathcal{I}}$ not necessarily distinct, with $R \equiv (G \Rightarrow C_0 \dots C_n)$, such that $b_k \in C_k^{\mathcal{I}}$ if the following conditions hold: - if $[\top : x_i] - r - [\top : x_j]$ is an arc of G then $(b_i, b_j) \in r^{\mathcal{I}}$, - if $[D : x_i]$ is a node in G then $b_i \in D^{\mathcal{I}}\}$

Table 2. Syntax and Semantics of $\mathcal{GDL}$

$\mathcal{GDL}$ extends $\mathcal{ALC}$ with the intersection $\sqcap$, composition $\circ$, converse of roles $\cdot^-$ and role identity $id(.)$: these constructs can be expressed with graph rules and existential graphs constructs as shown in Table 3. On the other hand, number restrictions, role union, instances can not be expressed in $\mathcal{GDL}$.
An additional expressiveness like the possibility to add relations in the conclusion of a graph rule would have led to loose the finite model property: for instance, it would have been possible to express the transitivity of a relation.

5 THE TABLEAUX EXPANSION RULES

In this section, we present a tableaux algorithm to prove the satisfiability of $\mathcal{GDL}$. Our proof is based on the construction of a *constraint system* (see for instance [2]). A constraint system (c.s) is a labeled graph $(\mathcal{N}, \mathcal{E}, \mathcal{L})$, where $\mathcal{N}$ is a set of nodes, $\mathcal{E}$ is a set of edges and $\mathcal{L}$ is a function mapping a node to a set of concepts and an edge to a set of roles. A c.s. contains a *clash* if for some node $x \in \mathcal{N}$, $\bot \in \mathcal{L}(x)$ or $\{A, \neg A\} \subseteq \mathcal{L}(x)$. A c.s. is *complete* when all applicable rules have been applied.
To prove the satisfiability of a concept C, we start with the structure $(\{x_0\}, \emptyset, \mathcal{L})$ where $\mathcal{L}(x_0) = C$. Then we build the c.s. by applying the expansion rules described in Figure 3, assuming C is in NNF. C is satisfiable iff the c.s. is complete and clash-free. We will prove it in the next section.
The expansion rule for the existential graph construct can be seen as

DL constructs	$\mathcal{GDL}$ expressions
$\exists RC$	$[\top : *x_0] \rightarrow (R) \rightarrow [C : *]$
$\forall RC$	$[\top : *x_0] \rightarrow (R) \rightarrow [\top : *x] \Rightarrow [C : *x]$
$\exists(R_1 \sqcap R_2)C$	$[\top : *x_0] \rightarrow (R_1) \rightarrow [C : *x]$ $\searrow (R_2) \nearrow$
$\forall(R_1 \sqcap R_2)C$	$[\top : *x_0] \rightarrow (R_1) \rightarrow [\top : *x] \Rightarrow [C : *x]$ $\searrow (R_2) \nearrow$
$\exists(R_1 \circ R_2)C$	$[\top : *x_0] \rightarrow (R_1) \rightarrow [\top : *] \rightarrow (R_2) \rightarrow [C : *x]$
$\forall(R_1 \circ R_2)C$	$[\top : *x_0] \rightarrow (R_1) \rightarrow [\top : *] \rightarrow (R_2) \rightarrow [\top : *x]$ $\Rightarrow [C : *x]$
$\exists(R^-)C$	$[\top : *x_0] \leftarrow (R) \leftarrow [C : *]$
$\forall(R^-)C$	$[\top : *x_0] \leftarrow (R) \leftarrow [\top : *x] \Rightarrow [C : *x]$
$\exists(R \sqcap id(C)).D$	$[C \sqcap D : *x_0] \bigcirc (R)$
$\forall(R \sqcap id(C)).D$	$[C : *x_0] \bigcirc (R) \Rightarrow [D : *x_0]$

Table 3. Expression of $\mathcal{ALC}(\sqcap, \circ, \cdot^-, id(.))$ constructs in $\mathcal{GDL}$

the generalization of the rule for the $\exists RC$ construct.
A naive expansion rule for the graph rule construct that would fire when the antecedent of the graph rule is matched would not be sound, as it would not detect that the following concept is unsatisfiable:
$[\top : x_0] \rightarrow (R) \rightarrow [C : *]$
$\sqcap \ [A : x_0] \rightarrow (R) \rightarrow [\top : *x] \Rightarrow [\neg C : *x]$
$\sqcap \ [\neg A : x_0] \rightarrow (R) \rightarrow [\top : *x] \Rightarrow [\neg C : *x]$
The sound rule, stated in Figure 3, fires as soon as just the relational part of the antecedent of the graph rule is matched.

$\sqcap$	if	$C_1 \sqcap C_2 \in \mathcal{L}(x)$ and $\{C_1, C_2\} \not\subseteq \mathcal{L}(x)$
	then	$\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C_1, C_2\}$
$\sqcup$	if	$C_1 \sqcup C_2 \in \mathcal{L}(x)$ and $C_1 \notin \mathcal{L}(x)$ and $C_2 \notin \mathcal{L}(x)$
	then	$\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C_1\}$ or $\mathcal{L}(x) \rightarrow \mathcal{L}(x) \cup \{C_2\}$
$\lambda x_0 G$	if	1. $\lambda x_0 G \in \mathcal{L}(b_0)$ 2. there does not exist $b_1 \in \mathcal{N}, \ldots, b_n \in \mathcal{N}$ not necessarily distinct, such that for all i,j: - if $[\top : x_i] \rightarrow r_{ij} \rightarrow [\top : x_j]$ is an arc in G then $r_{ij} \in \mathcal{L}(b_i, b_j)$ - if $[C_i : x_i]$ is a node in G then $C_i \in \mathcal{L}(b_i)$,
	then	1. create n new nodes $b_1, \ldots, b_n$ 2. $\mathcal{L}(b_i) = \{C_i\}$ if $[C_i : x_i]$ is a node in G 3. $\mathcal{L}(b_0) \rightarrow \mathcal{L}(b_0) \cup \{C_0\}$ if $C_0 \neq \top$ 3. $\mathcal{L}(b_i, b_j) = \{r_{ij}$ / $[\top : x_i] \rightarrow r_{ij} \rightarrow [\top : x_j]$ is an arc in G $\}$
$\lambda x_0 R$	if	1. $\lambda x_0 R \in \mathcal{L}(b_0)$ where R is $(\lambda x_0 x_1 \ldots x_n G$, $\lambda x_0 x_1 \ldots x_n [C_0 : x_0] \ldots [C_n : x_n])$ 2. there exists $b_1 \in \mathcal{N}, \ldots, b_n \in \mathcal{N}$ not necessarily distinct, such that if $[D_i : x_i] \rightarrow r_{ij} \rightarrow [D_j : x_j]$ is an arc in G then $r_{ij} \in \mathcal{L}(b_i, b_j)$
	then	for all i, $\mathcal{L}(b_i) \rightarrow \mathcal{L}(b_i) \cup \{D_i\} \cup \{C_i\}$ or $\mathcal{L}(b_0) \rightarrow \mathcal{L}(b_0) \cup \{\neg D_0\}$ or ... or $\mathcal{L}(b_n) \rightarrow \mathcal{L}(b_n) \cup \{\neg D_n\}$

Figure 3. Tableaux Expansion Rules for $\mathcal{GDL}$

6 SOUNDNESS AND COMPLETENESS

In this section, we prove soudness and completeness of our tableaux algorithm, and an NEXPTIME upper-bound for termination.
Let the *size* of a concept C, noted $s(C)$, be the number of symbols in its description. It is inductively computed as follows:
- $s(P) = 1$ for P a primitive concept,
- $s(A \sqcap B) = s(A) + s(B) + 1$ and $s(A \sqcap B) = s(A) + s(B) + 1$,
- $s(EG) = 1 + \sum_i s(C_i)$ where the C_i are the concepts of EG,
- $s(GR) = 1 + \sum_i s(C_i) + s(D_i)$ where the C_i are the concepts of the hypothesis of GR and D_i are the concepts of its conclusion.

Lemma 3 *Let n be the size of the input concept C for which a c.s. is being built ($\mathcal{L}(x_0) = C$). Each node x of the c.s. is labelled by at most $2n$ concepts.*

Proof. The number of subconcepts of C is bounded by n, and each concept labelling a node of the c.s. is either a subconcept of C or a subconcept of the negation of a subconcept of C (because of the $\lambda x_0 R$ rule).
A node labelled by the negation of an EG concept is in fact labelled by one GR concept (lemma 1).
A node labelled by the negation of a GR concept is in fact labelled by a disjunction of at most n EG concepts (lemma 2). We optimize the $\sqcup$ rule application by choosing one of these n EG concepts (instead of applying $n-1$ times the $\sqcup$ rule). A node labelled by the negation of a GR concept is thus in fact labelled by only one EG concept.
Each subconcept of the GR and EG concepts thus obtained is either a subconcept of C or the negation of a subconcept of C.
Negating a concept at a node x of the c.s. thus only adds one concept to x's labels, since all the subconcepts of this additional concept are either subconcepts of C or negations of subconcepts of C. Each node of the c.s. is thus labelled by at most $2n$ concepts.

Lemma 4 *Let n be the size of the input concept C for which a c.s. is being built ($\mathcal{L}(x_0) = C$). Each node x of the c.s. is directly linked to at most $2n$ other nodes.*

Proof. A node x of the c.s. is linked to both the nodes of the c.s. created by application of the $\lambda x_0 G$ rule at node x and the nodes created by application of the $\lambda x_0 G$ rule at another node x' of the c.s., this rule application having created nodes among which is x.
The application of the $\lambda x_0 G$ rule to x' has created at most n nodes; x may be linked to these n nodes.
Each of the $EG_{i,i=1..2n}$ concepts labelling x is a subconcept of C, or is obtained by negating a GR concept of the same size. These GR concepts are subconcepts of C, all different from each other (lemma 3). The sum of the sizes of the $EG_{i,i=1..2n}$ concepts is bounded by the size n of C. The applications of the $\lambda x_0 G$ rule to the $EG_{i,i=1..2n}$ at node x will then create at most n nodes in the c.s.; x may be linked to these n nodes.
x is thus linked to at most $2n$ nodes.

Lemma 5 *Let n be the size of the input concept C for which a c.s. is being built ($\mathcal{L}(x_0) = C$). For each node x of the c.s., there is an undirected path of length less or equal to n between x_0 and x.*

Proof. Node x has been created by applying the $\lambda x_0 G$ rule to a concept EG_1 at another node y_1 of the c.s.
A concept EG_i (resp. GR_i) is a label of a node y_i of the c.s. iff one of the following holds:
- The $\lambda x_0 G$ rule has been applied at a node y_{i+1} of the c.s. to a concept EG_{i+1}, with EG_i (resp. GR_i) being a subconcept of EG_{i+1}.

- The $\lambda x_0 R$ rule has been applied at a node y_{i+1} of the c.s. to a concept GR_{i+1}, with EG_i being a subconcept of a concept in the conclusion of GR_{i+1}.
- The $\lambda x_0 R$ rule has been applied at a node y_{i+1} of the c.s. to a concept GR_{i+1}, with EG_i (resp. GR_i) being a subconcept of the negation of a concept C_{i+1} in the hypothesis of GR_{i+1}.

By repeating this reasoning, we obtain a list of nodes $y_1, \ldots, y_k$ and a list of concepts $C_1, \ldots C_k$, where C_k is a subconcept of C and each $C_{i,i=1..k}$ is either an EG concept or a GR concept which is a subconcept of C_{i+1} (or can be identified to a subconcept of same length). Let the *graph size* of an EG concept (resp. a GR concept) be the number of nodes in its graph (resp. its graph hypothesis). The sum of the graph sizes of $C_{i,i=1..k}$ is thus bounded by n. Since $C_{i,i=1..k}$ are connected graphs, there exists a path of length less or equal to n between x_0 and x.

Theorem 1 *The application of the expansion rules for deciding of the satisfiability of a $\mathcal{GDL}$ concept C terminates in NEXP*TIME.

Proof. Each node of the c.s. built for C is linked to at most $2n$ other nodes (lemma 4) and there exists a path of length n between x_0 and each of the other nodes of the c.s. (lemma 5). The number of nodes of the c.s. is thus bounded by $(2n)^n$.

There are at most $2n$ concepts labelling each of the (at most) $(2n)^n$ nodes of the c.s. (lemma 3). The number of applications of an expansion rule is thus bounded by $(2n)^n * 2n$.

The cost of all the applications of the $\sqcup$ (resp. $\sqcap$) rule is thus bounded by $(2n)^n * 2n$.

The size of each GR (resp. EG) concept being less or equal to n, the cost of all the applications of the $\lambda x_0 R$ (resp. $\lambda x_0 G$) rule is thus bounded by $(2n)^n * 2n * ((2n)^n)^n$.

The application of the expansion rules terminates in $O(n^{n^2}) = O(2^{n^2 log(n)})$.

Theorem 2 (Soundness) *Let C be a $\mathcal{GDL}$ concept description in NNF. If there exists a sequence of expansion rules starting from C that results in a complete c.s. S without clash, then C is satisfiable.*

Proof. As S is clash-free, we use it to build an interpretation $\mathcal{I}$ with $\Delta^{\mathcal{I}} = \mathcal{N}$ and $C^{\mathcal{I}} = \{x \in \mathcal{N} / C \in \mathcal{L}(x)\}$ and $r^{\mathcal{I}} = \{(x_1, x_2) \in \mathcal{N}^2 / r \in \mathcal{L}(< x_1, x_2 >)\}$. We prove that $\mathcal{I}$ is a model of C by induction on the structure of C. Here we only detail the case of the GR construct. Let B be a subconcept of C with $B \in \mathcal{L}(x)$:

If $B \equiv \lambda x_0 R \in \mathcal{L}(b_0)$ with R is $(\lambda x_0 x_1 \ldots x_n G, \lambda x_0 x_1 \ldots x_n [C_0 : x_0] \ldots [C_n : x_n])$, then for all $b_1 \in \Delta^{\mathcal{I}}, \ldots, b_n \in \Delta^{\mathcal{I}}$ such that $(b_i, b_j) \in r_{ij}^{\mathcal{I}}$ if $[D_i : x_i] \rightarrow r_{ij} \rightarrow [D_j : x_j]$ is an arc in G, then either $b_i \in C_i^{\mathcal{I}} \cap D_i^{\mathcal{I}}$ for all i or $b_j \in (\neg D_j)^{\mathcal{I}}$ for some j. Thus $b_0 \in B^{\mathcal{I}}$.

It proves that $x_0 \in C^{\mathcal{I}}$, with x_0 the individual representing the starting node. Thus C is satisfiable.

Theorem 3 (Completeness) *Let C be a $\mathcal{GDL}$ concept description in NNF. If C is satisfiable, then there is a sequence of expansion rules starting from C that results in a complete and clash-free c.s.*

Proof. C is satisfiable; let us call $\mathcal{I}$ a model of C. We use $\mathcal{I}$ to guide the expansion rules applications. We inductively build a function π mapping the nodes of the c.s. to the individuals of $\Delta^{\mathcal{I}}$, such that $\pi(x) \in (\mathcal{L}'(x))^{\mathcal{I}}$ (1) and $(\pi(x), \pi(y)) \in (\mathcal{L}'(x, y))^{\mathcal{I}}$ (2) where $\mathcal{L}'(a)$ is the conjunction of elements of $\mathcal{L}(a)$. We suppose π has been defined for the c.s. being constructed and we extend it depending on the next expansion rule to be applied. Here we only detail the case of the GR rule:

If $\lambda x_0 R \in \mathcal{L}(c_0)$ with $\lambda x_0 R \equiv (\lambda x_0 x_1 \ldots x_n G, \lambda x_0 x_1 \ldots x_n [C_0 : x_0] \ldots [C_n : x_n])$ and $\pi(c_0) = b_0$, then $\pi(c_0) \in (\lambda x_0 R)^{\mathcal{I}}$ and for all $b_1 \in \Delta^{\mathcal{I}}, \ldots, b_n \in \Delta^{\mathcal{I}}$ such that $(b_i, b_j) \in r_{ij}^{\mathcal{I}}$ if $[D_i : x_i] \rightarrow r_{ij} \rightarrow [D_j : x_j]$ is an arc in G, then there are two cases: either $b_i \in C_i^{\mathcal{I}} \cap D_i^{\mathcal{I}}$ for all i or $b_j \in (\neg D_j)^{\mathcal{I}}$ for some j. For all nodes $c_1, \ldots, c_n$ (and c_0) such that $r_{ij} \in \mathcal{L}(< c_i, c_j >)$, in the first case we add C_i and D_i to $\mathcal{L}(c_i)$ for all i or in the second case we add $\neg D_j$ to $\mathcal{L}(b_j)$ for j. $\pi(c_i) \in (\mathcal{L}'(c_i))^{\mathcal{I}}$ still holds for all i.

The c.s. can thus be completed with π satisfying (1) and (2). $\mathcal{I}$ is a model of the c.s. and the c.s. is thus satisfiable and clash-free.

7 CONCLUSION

We have presented the graph-based concept description language $\mathcal{GDL}$, which we have shown to be decidable and for which we have given a sound and complete tableaux algorithm. $\mathcal{GDL}$ enables to represent concept descriptions with complex graph patterns as well as negation and disjunction, which leads to an expressive language.

Other works have studied the possibility of including specific or general graph patterns in concept descriptions. In DLs literature, [7] presents an algorithm for satisfiability of the DL $\mathcal{ALC}(\sqcap, \circ, \sqcup, \cdot^-, id(\cdot))$, for which a NEXPTIME upper bound is established. Role composition and intersection provide a way to represent some particular graph-like structures, that the author calls *cactus-shaped models*. Its algorithm takes advantage of the fact that the patterns in his DL are of a special kind: they are inductively defined thanks to $\circ$, $\sqcup$ and $\sqcap$ constructs, and can thus be inductively decomposed. Special expansion rules are provided, one for each role construct and each quantifier. When compared to $\mathcal{ALC}(\sqcap, \circ, \sqcup, \cdot^-, id(\cdot))$, $\mathcal{GDL}$ graph patterns are of any possible complexity and are handled as a whole, by a unique expansion rule for each quantifier.

In CGs literature, graph rules have been introduced in [9]. Their structure is more general than the one of $\mathcal{GDL}$ GR, but they do not handle disjunction nor negation.

REFERENCES

[1] Baader, F., Molitor, R., Tobies, S. Tractable and Decidable Fragments of Conceptual Graphs. In Proc. of ICCS'99, Blacksburg, VA, USA, LNAI 1640, Springer-Verlag, p. 480-493, 1999.

[2] Buchheit, M., Donini, F., Shaerf, A. Decidable reasoning in terminological knowledge representation systems. In Journal of Artifi cial Intelligence Research, 82:353-367, 1996.

[3] Chein, M., Mugnier, M-L., Simonet, G. Nested Graphs: A Graph-based Knowledge Representation Model with FOL Semantics. In Proc. of KR'98, Trento, Italy, Morgan Kaufmann Publishers, p. 524-534., 1998.

[4] Coupey, P., Faron, C. Towards Correspondence between Conceptual Graphs and Description Logics. In Proc. of ICCS'98, Montpellier, France, LNCS 1453, Springer-Verlag, p. 165-178, 1998.

[5] Leclere, M. Reasoning with type defi nitions. In proc. of ICCS'97, Seattle, WA, LNAI 1257, Springer-Verlag, 1997.

[6] Lehman, F. (ed.) Semantic Networks in Artifi cial Intelligence. Pergamon Press, Oxford, UK, 1992.

[7] Massacci, F. Decision Procedures for Expressive Description Logics with Intersection, Composition, Converse of Roles and Role Identity. In proc. of IJCAI'01, Seattle, WA, Morgan Kaufmann Publishers, 2001.

[8] Pierce, C.S. Collected Papers of Charles Sanders Pierce. In Hartshorne C. and Weiss P. eds, Harvard University Press, Cambridge, MA, 1932.

[9] Salvat, E. Theorem Proving Using Graph Operations in the Conceptual Graph Formalism. In proc. of ECAI'98, Brighton, UK, 1998.

[10] Sowa, J.F.: Conceptual Graphs, Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley, MA, 1984.