RWTH
LTCS–Report

Aachen University of Technology
Research group for
Theoretical Computer Science

On the Relation between Conceptual Graphs and Description Logics

Franz Baader and Ralf Molitor and Stefan Tobies

LTCS-Report 98-11

Technical report to the article "Tractable and Decidable Fragments of Conceptual Graphs" presented at ICCS'99

RWTH Aachen
LuFg Theoretische Informatik
http://www-lti.informatik.rwth-aachen.de

Ahornstr. 55
52074 Aachen
Germany

On the Relation between Conceptual Graphs and Description Logics

Franz Baader and Ralf Molitor and Stefan Tobies
LuFg Theoretische Informatik, RWTH Aachen
email: {baader,molitor,tobies}@informatik.rwth-aachen.de

1 Introduction

Conceptual graphs (CGs) are an expressive formalism for representing knowledge about an application domain in a graphical way. Since CGs can express all of first-order predicate logic (FO), they can also be seen as a graphical notation for FO formulae.

In knowledge representation, one is usually not only interested in *representing* knowledge, one also wants to *reason* about the represented knowledge. For CGs, one is, for example, interested in validity of a given graph, and in the question whether one graph subsumes another one. Because of the expressiveness of the CG formalism, these reasoning problems are undecidable for general CGs. In the literature [Sow84, Wer95, KS97] one can find complete calculi for validity of CGs, but implementations of these calculi have the same problems as theorem provers for FO: they may not terminate for formulae that are not valid, and they are very inefficient. To overcome this problem, one can either employ incomplete reasoners, or try to find decidable (or even tractable) fragments of the formalism. This paper investigates the second alternative.

The most prominent decidable fragment of CGs is the class of simple conceptual graphs (SGs), which corresponds to the conjunctive, positive, and existential fragment of FO (i.e., existentially quantified conjunctions of atoms). Even for this simple fragment, however, subsumption is still an NP-complete problem [CM92]. SGs that are trees provide for a tractable fragment of SGs, i.e., a class of simple conceptual graphs for which subsumption can be decided in polynomial time [MC93]. In this report, we will identify a tractable fragment of SGs that is larger than the class of trees.

Instead of trying to prove new decidability or tractability results for CGs from scratch, our idea was to transfer decidability results from description logics [DLNN97, DLNS96] to CGs. The goal was to obtain a "natural" sub-class of the class of all CGs in the sense that, on the one hand, this sub-class is defined directly by syntactic restrictions on the graphs, and not by conditions on the first-order formulae obtained by translating CGs into FO, and, on the other hand, is in some sense equivalent to a more or less expressive description logic.

Although description logics (DLs) and CGs are employed in very similar applications (e.g., for representing the semantics of natural language sentences), it turned out that these two formalisms are quite different for several reasons: (1) conceptual graphs are interpreted as closed FO formulae, whereas DL concept descriptions are interpreted by formulae with one free variable; (2) DLs do not allow for relations of arity > 2; (3) SGs are interpreted by existential sentences, whereas almost all DLs considered in the literature allow for universal quantification; (4) because DLs use a variable-free syntax, certain identifications of variables expressed by cycles in SGs and by co-reference links in CGs cannot be expressed in DLs. As a consequence of these differences, we could not identify a natural fragment of CGs corresponding to an expressive DL whose decidability was already shown in the literature. We could, however, obtain a new tractability result for a DL corresponding to SGs that are trees. This correspondence result strictly extends the one in [CF98]. In addition, we have extended the tractability result from SGs that are trees to SGs that can be transformed into trees using a certain "cycle-cutting" operation.

The report is structured as follows. We first introduce the description logic for which we will identify a subclass of equivalent SGs. In Section 3, we recall basic definitions and results on SGs. Thereafter, we introduce a syntactical variant of SGs which allows for directly encoding the support into the graphs (Section 4.1). In order to formalize the equivalence between DLs and SGs, we have to consider SGs with one distinguished node called root (Section 4.2). In Section 5, we finally identify a class of SGs corresponding to a DL that is a strict extension of the DL considered in [CF98].

2 Description Logics

We first introduce syntax and semantics of description logics (DLs) with existential restrictions as well as the inference problem of subsumption. In DLs, knowledge from an application domain is represented by so-called *concept descriptions*. Concept and role descriptions are inductively defined with the help of a set of *constructors*, starting with a set $\mathcal{N}_I$ of *constants*, a set $\mathcal{N}_C$ of *primitive concepts* (unary predicates), and a set $\mathcal{N}_R$ of *primitive roles* (binary predicates). The semantics of a concept description is also inductively defined whereby primitive concepts are interpreted as subsets of a domain Δ and role names are interpreted as binary relations on $\Delta \times \Delta$.

Definition 1 *Let $\mathcal{N}_C$ be a set of* primitive concepts*, $\mathcal{N}_R$ a set of* primitive roles*, and $\mathcal{N}_I$ a set of constants.* Concept descriptions *and* role terms *are inductively defined as follows.*

- *Each primitive concept P is a concept description.*
- *For each $a \in \mathcal{N}_I$, ONE-OF(a) is a concept description. In the sequel, we will use $\{a\}$ as an abbreviation for ONE-OF(a).*
- *Let C, D be concept descriptions and r a role term. Then*

Construct name	Syntax	Semantics	
top-concept	$\top$	$x = x$	
primitive concept $P \in \mathcal{N}_C$	P	$P(x)$	
conjunction	$C \sqcap D$	$\Psi_C(x) \wedge \Psi_D(x)$	$\mathcal{EL}$
existential restriction	$\exists r.C$	$\exists y.\Psi_r(x,y) \wedge \Psi_C(y)$	
constant $a \in \mathcal{N}_I$	$\{a\}$	$x = a$	$\mathcal{O}^1$
primitive role $r \in \mathcal{N}_R$	r	$r(x,y)$	
inverse role for $r \in \mathcal{N}_R$	r^-	$r(y,x)$	$\mathcal{I}$
role conjunction	$r_1 \sqcap r_2$	$\Psi_{r_1}(x,y) \wedge \Psi_{r_2}(x,y)$	$\mathcal{R}$

Table 1: Syntax and semantics of $\mathcal{ELIRO}^1$-concept descriptions.

- $C \sqcap D$ *(conjunction),*
- $\forall r.C$ *(value restriction),*
- $\exists r.C$ *(existential restriction)*

are concept descriptions as well.

- *For each primitive role r, r as well as the* inverse role r^- *is a role term.*
- *Let r_1, r_2 be role terms. Then the conjunction $r_1 \sqcap r_2$ is also a role term.*

An interpretation $\mathcal{I} = (\Delta, \cdot^I)$ exists of a set of individuals Δ and a function $\cdot^I$ that maps each constant $a \in \mathcal{N}_I$ to an element $a^I \in \Delta$, each primitive concept $P \in \mathcal{N}_C$ to a subset $P^I \subseteq \Delta$, and each primitive role $r \in \mathcal{N}_R$ to a subset $r^I \subseteq \Delta \times \Delta$. The extension of $\cdot^I$ to arbitrary concept descriptions and role terms is inductively defined as shown in Table 1. The constructor $\top$ describes the entire domain.

The set of concept descriptions defined by the set of constructors introduced in Definition 1 is denoted by $\mathcal{ELIRO}^1$, whereby $\mathcal{EL}$ denotes a language only allowing for existential restrictions and conjunction. The suffix $\mathcal{IRO}^1$ indicates an extension by inverse roles, conjunction of roles, and constants. Note that $\mathcal{O}$ in general indicates that the language allows for concept descriptions of the form ONE-OF$(a_1, \ldots, a_n)$. Since we are only interested in representing referents of simple graphs, we only need unary ONE-OF-descriptions.

Example 2 Throughout the paper, we will use the following signature for examples of concept descriptions as well as simple graphs:

$\mathcal{N}_C$:={Human, Male, Female, Student, CScourse}
$\mathcal{N}_R$:={attends, has-child, likes}
$\mathcal{N}_I$:={Mary, Peter, KR101}

The concept description

$$D = \mathsf{Female} \sqcap \exists \mathsf{likes}.\mathsf{Male} \sqcap \exists \mathsf{has\text{-}child}.(\mathsf{Student} \sqcap \exists \mathsf{attends}.\mathsf{CScourse})$$

Construct name	Description	FO formula
top-concept	$\top$	$x = x$
primitive concept $P \in \mathcal{N}_C$	P	$P(x)$
conjunction	$C \sqcap D$	$\Psi_C(x) \wedge \Psi_D(x)$
existential restriction	$\exists r.C$	$\exists y.\Psi_r(x,y) \wedge \Psi_C(y)$
constant $a \in \mathcal{N}_I$	$\{a\}$	$x = a$
primitive role $r \in \mathcal{N}_R$	r	$r(x,y)$
inverse role for $r \in \mathcal{N}_R$	r^-	$r(y,x)$
role conjunction	$r_1 \sqcap r_2$	$\Psi_{r_1}(x,y) \wedge \Psi_{r_2}(x,y)$

Table 2: Translating concept descriptions and role terms into FO formulae.

describes all women who like a man and have a child that is a student attending a computer science course (CScourse).

In order to obtain a structured representation of the knowledge about the application domain one is interested in the subsumption hierarchy formed by the concept descriptions.

Definition 3 (Subsumption) *Let C, D be concept descriptions.*
D subsumes *C (for short $C \sqsubseteq D$) iff $C^I \subseteq D^I$ for all interpretations $\mathcal{I}$.*
C is equivalent *to D (for short $C \equiv D$) iff $C \sqsubseteq D$ and $D \sqsubseteq C$, i.e., $C^I = D^I$ for all interpretations $\mathcal{I}$.*

In order to formalize the equivalence between $\mathcal{ELIRO}^1$ and simple graphs, we translate concept descriptions into first order formulae [Bor96]. Each concept description C yields a FO formula $\Psi_C(x)$ with one free variable x and each role term r is translated into a formula $\Psi_r(x,y)$ with two free variables x, y. An inductive definition of Ψ_C is shown in Table 2.

The semantics of C can now be described by $C^I = \{\xi \in \Delta \mid \mathcal{I} \models \Psi_C(\xi)\}$ whereby $\mathcal{I} = (\Delta, \cdot^I)$ is an interpretation of the signature $\Sigma = \mathcal{N}_C \cup \mathcal{N}_R \cup \mathcal{N}_I$. Further, subsumption between concept descriptions can be characterized w.r.t. the corresponding FO formulae by $C \sqsubseteq D$ iff $\Psi_C(x_0) \models \Psi_D(x_0)$, i.e., $\forall x_0.\Psi_C(x_0) \rightarrow \Psi_D(x_0)$ is valid.

Example 4 (Example 2 continued) The semantics of the concept description D introduced in Example 2 is given by the following FO formula:

$$\begin{aligned}\Psi_D(x_0) \quad = \quad & \mathsf{Female}(x_0) \wedge \exists x.(\mathsf{likes}(x_0, x) \wedge \mathsf{Male}(x)) \wedge \\ & \exists y.(\mathsf{has\text{-}child}(x_0, y) \wedge \mathsf{Student}(y) \wedge \exists z.(\mathsf{attends}(y, z) \wedge \mathsf{CScourse}(z))).\end{aligned}$$

The concept description

$$\begin{aligned}C := \quad & \mathsf{Female} \sqcap \exists \mathsf{has\text{-}child}^-.\{\mathsf{Peter}\} \sqcap \exists(\mathsf{likes} \sqcap \mathsf{has\text{-}child}). \\ & (\mathsf{Male} \sqcap \mathsf{Student} \sqcap \exists \mathsf{attends}.(\mathsf{CScourse} \sqcap \{\mathsf{KR101}\}) \sqcap \exists \mathsf{likes}.\{\mathsf{Peter}\}).\end{aligned}$$

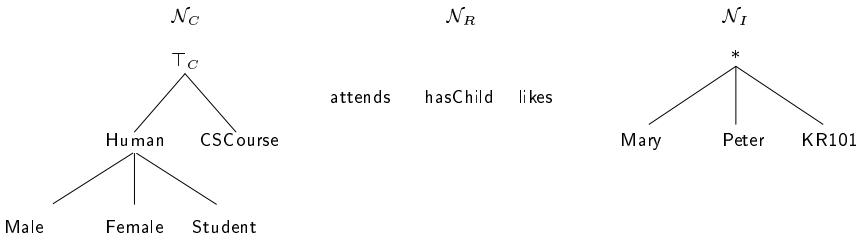

Figure 1: An example of a support.

describes all daughters of Peter who have a dear son, where this guy attends the computer science course number KR101 and likes Peter.

It is not hard to see that C yields a specialization of the situation described by D given in Example 2, i.e., $C \sqsubseteq D$.

3 Simple Graphs

Conceptual graphs have first been introduced by Sowa in [Sow84]. In this paper, we consider the class of simple graphs (see [Sow84] 3.1.2, page 73, and [CM92, CMS98]). For the readers convenience, we first recall the basic definitions.

Definition 5 (Support) *A* support *is a tuple* $\mathcal{S} = \langle \mathcal{T}_C, \mathcal{T}_R, \mathcal{T}_I \rangle$ *where*

- $\mathcal{T}_C = (\mathcal{N}_C, <_C)$ *is the* concept type hierarchy *of* $\mathcal{S}$. $\mathcal{N}_C$ *is a set of names of concept types which contains a distinguished type* $\top$*, and* $<_C$ *is a partial ordering on* $\mathcal{N}_C$ *which has* $\top$ *as its greatest element.*

- $\mathcal{T}_R = \langle (N_R^1, N_R^2, \ldots), <_R \rangle$ *is the* relation type hierarchy *of* $\mathcal{S}$. N_R^i *contains the relation symbols of arity* i*. We require* $N_R^i \cap N_R^j = \emptyset$ *if* $i \neq j$*. We define* $\mathcal{N}_R = \bigcup_{i \in \mathbb{N}} N_R^i$. $<_R$ *is a partial ordering on* $\mathcal{N}_R$ *for which relation types of different arity must be incomparable.*

- $\mathcal{T}_I = \langle \mathcal{N}_I \cup \{*\}, <_I \rangle$ *where* $\mathcal{N}_I$ *is the set of* individual markers *and* $*$ *is a distinguished marker called the* generic marker*. The partial ordering* $<_I$ *on* $\mathcal{N}_I \cup \{*\}$ *is defined by requiring* $*$ *to be the greatest element w.r.t.* $<_I$ *and all other elements to be pairwise incomparable.*

As an example consider the signature $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ introduced in Example 2. An intuitive ordering of the concept types is depicted in Figure 1. The binary relations are unordered and the constants are ordered as described in Definition 5.

Definition 6 (Simple graphs) *Let* $\mathcal{S}$ *be a support. A* simple graph *is a tuple* $g = (C, R, E, \ell)$*, where*

- *(C, R, E) is an undirected bipartite graph with vertex sets C and R and edge relation $E \subseteq C \times R$. The set C and R are called the sets of* concept nodes *and* relation nodes *of g respectively.*
- *ℓ labels (C, R, E) in the following way:*

$$\ell : \begin{cases} R \to \mathcal{N}_R \\ C \to \mathcal{N}_C \times \mathcal{N}_I \\ E \to 2^{\mathbf{N}} \setminus \{\emptyset\} \end{cases}$$

ℓ must satisfy: $\forall r \in R : \ell(r) \in N_R^i \Rightarrow \{\ell(c, r) \mid (c, r) \in E\}$ is a disjoint partition of $\{1, \ldots, i\}$.

For a concept node c if $\ell(c) = (type(c), ref(c))$, $type(c)$ and $ref(c)$ are called the type *and* referent *of c respectively.*

We define $SG(\mathcal{S})$ to be the set of all simple graphs over $\mathcal{S}$.

In description logics, we are only concerned with unary and binary relations. Since we are interested in a comparison between simple graphs and description logics, we reduce our attention to unary and binary predicates, i.e., in the support $\mathcal{S}$, we only allow for concept types and binary relations. In this special case, the notion of a simple graph can be simplified as follows. We discard from the set of relation nodes R, the set of edges E, and labeling R and E. Instead, for each relation node in g we introduce labeled edges directed from the first neighbour to the second. Thus, we obtain a set $E \subseteq C \times \mathcal{N}_R \times C$ of labeled edges and reduce the labeling function to $\ell : C \to \mathcal{N}_C \times \mathcal{N}_I$.

A second restrictions results from the following observation. An existential restriction $\exists r.C$ describes individuals having an r-successor that fulfills the restrictions given by C. Nested existential restrictions describe connected structures. Thus, we only have to consider connected SGs.

To sum up, we assume in the sequel that the support is of the form $\mathcal{S} = \langle \mathcal{T}_C, \mathcal{T}_R, \mathcal{I} \rangle$ whereby $\mathcal{T}_R = \langle N_R^2, <_R \rangle$, i.e., we only allow for binary relations. Additionally, we restrict our attention to connected simple graphs over $\mathcal{S}$. This kind of SGs will be denoted by $g = (V, E, \ell)$.

Simple graphs have a straightforward graphical representation by just drawing the graph and attaching the labels to the different parts of the graph. Thereby, an edge $v_1 r v_2$ is represented by an arrow from v_1 to v_2 labeled with r. Concept nodes are represented by rectangular boxes containing the labeling.

The fundamental reasoning service on simple graphs is computing structural *subsumption* relations. The subsumption relation is induced by homomorphisms between SGs. Simple graphs are provided with a first order semantics denoted by Φ [Sow84, CMS98]. The characterization of subsumption has been proven to be sound and complete w.r.t. Φ for SGs in normal form [CMS98, CM92, Sow84]. A SG $g = (C, R, E, \ell)$ is said to be in *normal form* if each individual $a \in \mathcal{N}_I$ appears at most once as the referent of a concept node.

A formal definition of Φ can be found in [Sow84] and [CMS98]. For a given support $\mathcal{S} = \langle \mathcal{T}_C, \mathcal{T}_R, \mathcal{T}_I \rangle$, $\Phi(\mathcal{S})$ is a FO formula corresponding to the interpretation of the partial orderings of $\mathcal{T}_C$ and $\mathcal{T}_R$, i.e., $t_1 <_C t_2$ yields the formula

$\forall x.(t_1(x) \rightarrow t_2(x))$ and $t_1 <_R t_2$ yields the formula $\forall x \forall y.(t_1(x,y) \rightarrow t_2(x,y))$. Then, $\Phi(\mathcal{S})$ is defined as the conjunction of all formulae induced by $\mathcal{T}_C$ and $\mathcal{T}_R$. The semantics $\Phi(g)$ of a SG $g = (V, E, \ell)$ is defined as follows. Each concept node $v \in V$ with label $\ell(v) = (P, a)$ corresponds to an atomic formula $P(x)$ whereby x is a variable if $a = *$ and $x = a$, if $a \in \mathcal{N}_I$. Two distinct generic nodes receive distinct variables. Each edge $v_1 r v_2 \in E$ yields an atomic formula of the form $r(x_1, x_2)$ where x_i is the term corresponding to v_i, $i = 1, 2$. Finally, $\Phi(g)$ is defined as the existentially closed conjunction of all atoms induced by concept nodes and edges in g. Throughout the paper, for a given SG $g = (V, E, \ell)$ we use $id(v)$ to refer to the term corresponding to $v \in V$ in $\Phi(g)$.

We consider the subsumption relation on SG based on the *projection operation* [Sow84]. In [CMS98], projection is defined as a graph homomorphism between simple graphs. Since this characterization will be used in the comparison within the next section, we recall the formal definition of a homomorphism between two SGs g and h w.r.t. a support $\mathcal{S}$.

Definition 7 (Homomorphism on Simple Graphs w.r.t. $\mathcal{S}$)
Let $g = (V_G, E_G, \ell_G)$ and $h = (V_H, E_H, \ell_H)$ be two simple graphs over support $\mathcal{S}$. A mapping $\varphi : V_H \longrightarrow V_G$ is a homomorphism *from h to g w.r.t. $\mathcal{S}$ iff the following conditions are satisfied:*

- *for all $v \in V_H$ let $\ell_H(v) = (P, a)$ and $\ell_G(\varphi(v)) = (P', a')$; then $P' \leq_C P$ and $a' \leq_I a$, and*
- *for all $vrw \in E_H$, there exists r' such that $r' \leq_R r$ and $\varphi(v) r \varphi(w) \in E_G$.*

Subsumption between SGs over support $\mathcal{S}$ is defined by *h subsumes g* (for short $g \sqsubseteq h$) iff there exists a homomorphism from h to g.

Subsumption is sound and complete w.r.t. the FO semantics, i.e., given two SGs g, h, then if $g \sqsubseteq h$ then $\Phi(\mathcal{S}) \wedge \Phi(g) \rightarrow \Phi(h)$ is valid [Sow84]; conversely, if g is in normal form and $\Phi(\mathcal{S}) \wedge \Phi(g) \rightarrow \Phi(h)$ is valid, then h subsumes g [CM92]. Note that one can not always obtain a simple graph in normal form equivalent to g if g is not in normal form. For example, consider a SG g over the support $\mathcal{S}$ (as given in Figure 1) with two nodes v_1, v_2 labeled with (Male, Peter) and (Student, Peter), respectively. Then there exists no SG in normal form equivalent to g, because either the information that Peter is a man or that Peter is a student cannot be expressed in a SG over $\mathcal{S}$ in normal form.

Example 8 Consider the SGs depicted in Figure 2. They are defined over the support $\mathcal{S}$ given in Figure 1. The SG g depicted on the left hand side describes all daughters of Peter who have a child that is a student and that likes his mother. The SG h describes a parent whose child is liked by another human. The SG h subsumes g, because mapping w_i to v_i for $0 \leq i \leq 2$ yields a homomorphism w.r.t. $\mathcal{S}$ from h to g.

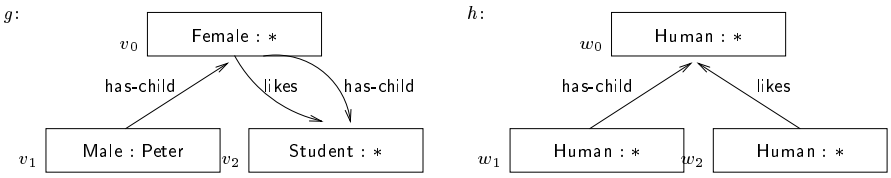

Figure 2: Subsumption of simple graphs w.r.t. $\mathcal{S}$.

4 Expanded Simple Graphs and Rooted Simple Graphs

In order to formalize the correspondence between simple graphs and concept descriptions, we introduce syntactical variants of SGs.

1. We first eliminate the support $\mathcal{S}$ and define the class of so-called *expanded simple graphs*. More precisely, we allow for node labels of the form $(\{P_1, \ldots, P_n\}, x)$, i.e., the type of a concept node may now be a set of concept types. Such a type is interpreted as the conjunction of its elements and it directly corresponds to conjunctions of primitive concepts in a concept description. The referent is still an individual marker $a \in \mathcal{N}_I$ or the generic marker $*$.

2. Then, we introduce so-called *rooted simple graphs*. A rooted SG is an expanded SG with exactly one distinguished node called the *root*.

The semantics of expanded and rooted SGs is obtained from appropriate extensions of Φ. Since we can encode a support $\mathcal{S}$ into expanded SGs, there is a one-to-one correspondence between subsumption of expanded SGs and subsumption of SGs w.r.t. $\mathcal{S}$. Subsumption of rooted SGs yields a refinement of the subsumption relation on expanded SGs, because we obtain an additional condition on roots.

4.1 Expanded Simple Graphs

As already mentioned, we reconsider the notion of a SG in order to cope with conjunctions of primitive concepts. But actually this extension is interesting by itself since it allows for representing objects that are connected to several different concept types. In SGs over a support $\mathcal{S}$ one can only assign one type to a concept node and has to discard from additional types unless they occur as more general types in $\mathcal{S}$. In expanded SGs, one can assign a set of concept types to a concept node independently from a possible specialization relation between them.

Definition 9 (Expanded Simple Graphs) *A SG $\mathcal{G} = (V, E, \ell)$ is called* expanded simple graph *over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ if $E \subseteq V \times \mathcal{N}_R \times V$ and for all $v \in V$*

it is $\ell(v) = (typ(v), ref(v))$ *with* $typ(v) \subseteq \mathcal{N}_C$, *and* $ref(v) \in \mathcal{N}_I \cup \{*\}$. *The distinguished concept type* $\top_C$ *corresponds to the empty set.*

Note that the signature $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ of the class of expanded SGs is not ordered by a relation, i.e., expanded SGs are not defined over a support $\mathcal{S}$. However, we will see below that expanded SGs are as expressive as SGs over a support $\mathcal{S}$.

The FO semantics Φ of SGs extends to expanded SGs in a natural way. In order to cope with types of the form $typ(v) = \{P_1, \ldots, P_n\}$, $n \geq 1$, $P_i \in \mathcal{N}_C$, we introduce the conjunction $\bigwedge_{1 \leq i \leq n} P_i(id(v))$ instead of an atomic formula $P(id(v))$. As before, an edge $v_1 r v_2 \in E$ is translated into an atomic formula $r(x_1, x_2)$. Finally, the FO semantics $\Phi_e(\mathcal{G})$ of an expanded SG $\mathcal{G} = (V, E, \ell)$ is defined as the existentially closed conjunction of all atomic formulae corresponding to the concept nodes and edges of $\mathcal{G}$.

Remark 10 *At this point we should remark something about the normal form of SGs and expanded SGs. In [CMS98], the authors introduce the normal form of a SG g as the SG obtained from g by identifying all concept nodes having the same individual referent. In order to obtain an equivalent SG, the types of the identified nodes must be equal, or at least, they must possess a greatest lower bound in* $(\mathcal{N}_C, <_C)$.

On the other hand, we can obtain an equivalent expanded simple graph in normal form from any expanded SG $\mathcal{G}$ *by modifying the above transformation as follows: identify all concept nodes having the same individual marker and define the type of the resulting node as the union of the types of the identified nodes. The resulting graph* $\mathcal{G}'$ *is again an expanded SG. Further, the generic concept nodes in* $\mathcal{G}$ *are kept unchanged in* $\mathcal{G}'$. *Consequently, there exists a bijection* π *between the variables in* $\Phi_e(\mathcal{G})$ *and* $\Phi_e(\mathcal{G}')$, *respectively, and the set of atomic formulae occuring in* $\Phi_e(\mathcal{G})$ *is equal to the set corresponding to* $\Phi_e(\mathcal{G}')$ *w.r.t.* π. *So,* $\mathcal{G}$ *is equivalent to* $\mathcal{G}'$, *i.e.,* $\Phi_e(\mathcal{G}) \equiv \Phi_e(\mathcal{G}')$[1].

Just as for SGs, subsumption between expanded SGs is based on graph homomorphisms. Thereby, we have to adjust the conditions on node labels.

Definition 11 (Homomorphisms between Expanded Simple Graphs) *Let* $\mathcal{G} = (V_G, E_G, \ell_G)$ *and* $\mathcal{H} = (V_H, E_H, \ell_H)$ *be two expanded SGs over* $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$. *A mapping* $\varphi : V_H \longrightarrow V_G$ *is a* homomorphism *from* $\mathcal{H}$ *to* $\mathcal{G}$ *iff the following conditions are satisfied:*

- *for all* $v \in V_H$ *let* $\ell(v) = (\{P_1, \ldots, P_n\}, a)$ *and* $\ell_G(\varphi(v)) = (\{Q_1, \ldots, Q_m\}, a')$; *then*
  - $\{P_1, \ldots, P_n\} \subseteq \{Q_1, \ldots, Q_m\}$ *and*
  - $a' = a$ *or* $(a' \in \mathcal{N}_I$ *and* $a = *)$.
- *for all* $vrw \in E_H$ *it is* $\varphi(v) r \varphi(w) \in E_G$.

[1] We call two FO formulae Φ_1, Φ_2 without free variables *equivalent* if $\Phi_1 \leftrightarrow \Phi_2$ is valid.

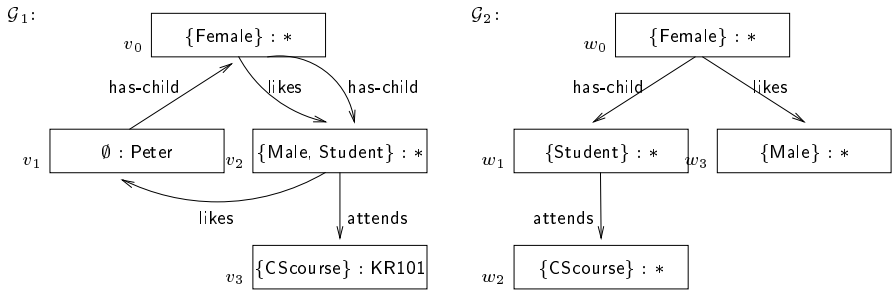

Figure 3: Subsumption of expanded simple graphs.

Note that the condition on the referents in node labels is equivalent to the condition $a' \leq_I a$ in Definition 7.

Example 12 Consider the expanded SGs depicted in Figure 3. They are defined over the signature $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ given in Example 2. The expanded SG $\mathcal{G}_1$ describes all women that are a daughter of Peter, and have a dear son that likes Peter and is a student attending the CScourse number KR101.

The expanded SG $\mathcal{G}_2$ depicted on the right hand side subsumes $\mathcal{G}_1$ because mapping w_0 onto v_0, w_1 and w_3 onto v_2, and w_2 onto v_3 yields a homomorphism from $\mathcal{G}_2$ to $\mathcal{G}_1$.

In the sequel, we will show that expanded simple graphs over a signature $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ are not more expressive than simple graphs over a support $\mathcal{S}$. More precisely, we will show that there exist translations α and β such that the following holds:

1. Let $\mathcal{G} = (V_G, E_G, \ell_G)$ be an expanded SG in normal form and $\mathcal{H} = (V_H, E_H, \ell_H)$ an expanded SG over the signature $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$. There exist SGs $\alpha(\mathcal{G}) = (V_G, E'_G, \ell'_G)$ and $\alpha(\mathcal{H}) = (V_H, E'_H, \ell'_H)$ defined over a support $\mathcal{S}$ in such a way that
   $\varphi : V_H \longrightarrow V_G$ is a homomorphism from $\mathcal{H}$ to $\mathcal{G}$
   iff $\Phi_e(\mathcal{G}) \models \Phi_e(\mathcal{H})^2$
   iff $\Phi(\mathcal{S}) \wedge \Phi(\alpha(\mathcal{G})) \models \Phi(\alpha(\mathcal{H}))$
   iff φ is a homomorphism from $\alpha(\mathcal{H})$ to $\alpha(\mathcal{G})$ w.r.t. $\mathcal{S}$.

2. Let $g = (V_G, E_G, \ell_G)$ be a SG over support $\mathcal{S}$ in normal form and $h = (V_H, E_H, \ell_H)$ be a SG over support $\mathcal{S}$. Let $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ denote the signature of $\mathcal{S}$. There exist expanded SGs $\beta(g) = (V_G, E'_G, \ell'_G)$ and $\beta(h) = (V_H, E'_H, \ell'_H)$ over the $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ such that
   $\varphi : V_H \longrightarrow V_G$ is a homomorphism from h to g w.r.t. $\mathcal{S}$
   iff $\Phi(\mathcal{S}) \wedge \Phi(g) \models \Phi(h)$
   iff $\Phi_e(\beta(g)) \models \Phi_e(\beta(h))$
   iff φ is a homomorphism from $\beta(h)$ to $\beta(g)$.

Due to 1. we get that expanded SGs are not more expressive than SGs w.r.t. $\mathcal{S}$. By 2. we get that the support can be encoded into expanded SGs, i.e., w.l.o.g. we can abstain from the support $\mathcal{S}$.

Translating Simple Graphs into Expanded Simple Graphs

The idea behind the translation β of SGs over a support $\mathcal{S}$ into expanded SGs over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ is as follows. The support $\mathcal{S}$ provides us with a partial ordering of concept types and binary relations. Intuitively, $\mathcal{S}$ represents specialization hierarchies on $\mathcal{N}_C$ and $\mathcal{N}_R$, respectively, e.g., if $P <_C Q$, then P is a specialization of Q. Thus, whenever a concept node v is labeled with type P and $P <_C Q$, then v also represents an instance of Q. Since the subsumption relation yields a specialization hierarchy on $\mathrm{SG}(\mathcal{S})$, $<_C$ and $<_R$ have to be taken into account within the definition of homomorphisms between SGs.

Within the class of expanded SGs, we discard from the support $\mathcal{S}$, i.e., the information that is implicit encoded in the partial orderings is explicitly represented in the labels of concept nodes and the set of edges. Formally, let $g = (V, E, \ell)$ be a simple graph over $\mathcal{S}$. We define $\beta(g) := (V, E', \ell')$ by

- $E' := \{v_1 s v_2 \mid \text{ exists } v_1 r v_2 \in E \text{ and } r \leq_R s\}$,
- $\ell'(v) := (typ'(v), ref'(v))$ with
  - $typ'(v) := \{Q \mid typ(v) = P \text{ and } P \leq_C Q\}$,
  - $ref'(v) := ref(v)$.

$\leq_C$ and $\leq_R$ denote the reflexive closure of $<_C$ and $<_R$, respectively. Obviously, it is $P \leq_C Q$ iff $\{P' \mid P \leq_C P'\} \subseteq \{Q' \mid Q \leq_C Q'\}$. Further, it is $r \leq_R s$ iff $\{v_1 r' v_2 \mid r \leq_R r'\} \subseteq \{v_1 s' v_2 \mid s \leq_R s'\}$. Using these observations it is not hard to prove that φ is a homomorphism from h to g w.r.t. $\mathcal{S}$ iff φ is a homomorphism from $\beta(h)$ to $\beta(g)$.

In order to prove all equivalences proposed in item 2 on page 10 we need the following theorem.

Theorem 13 *Let $\mathcal{G} = (V_G, E_G, \ell_G)$ be an expanded SG in normal form and $\mathcal{H} = (V_H, E_H, \ell_H)$ an expanded SG. There exists a homomorphism $\varphi : V_H \longrightarrow V_G$ from $\mathcal{H}$ to $\mathcal{G}$ iff $\Phi_e(\mathcal{G}) \models \Phi_e(\mathcal{H})$.*

Proof: A proof of Theorem 13 can be obtained from the proof of Theorem 1 in [CMS98], pages $527f$, by just neglecting the formula $\Phi(\mathcal{S})$.

Translating Expanded Simple Graphs into Simple Graphs

In order to complete the proof of the equivalences proposed in the items 1 and 2 on page 10, we have to define an appropriate translation α from expanded SGs over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ to SGs over a support $\mathcal{S}$.

In the first step, we define the support $\mathcal{S} := (\mathcal{T}_C, \mathcal{T}_R, \mathcal{T}_I)$ by

- $\mathcal{T}_C := (\mathcal{N}'_C, <_C)$ where
  - $\mathcal{N}'_C := \{0,1\}^n = \{(b_1, \ldots, b_n) \mid b_i \in \{0,1\}, 1 \leq i \leq n\}$, $n = |\mathcal{N}_C|$, and
  - for $\mathbf{b}, \mathbf{b}' \in \mathcal{N}'_C$ it is $\mathbf{b} \leq_C \mathbf{b}'$ iff $b_i \geq b'_i$ for all $1 \leq i \leq n$,
- $\mathcal{T}_R = (\mathcal{N}_R, \emptyset)$, and
- $\mathcal{T}_I = (\mathcal{N}_I \cup \{*\}, <_I)$ where $<_I$ is defined as in Definition 5.

The idea behind the definition of $\mathcal{T}_C$ is as follows. Let $\mathcal{N}_C = \{P_1, \ldots, P_n\}$. Then $\mathcal{T}_C$ yields a binary encoding of the partial ordering $(\mathcal{P}(\mathcal{N}_C), \subseteq)$ of all subsets of $\mathcal{N}_C$. More precisely, there is a one-to-one correspondence between subsets $Q \subseteq \mathcal{N}_C$ and concept types $t \in \mathcal{N}'_C$:

- For $Q = \{P_{i_1}, \ldots, P_{i_m}\}$ it is $t_Q := (b_1, \ldots, b_n)$ with $b_i = 1$ iff $i \in \{i_1, \ldots, i_m\}$.
- For $t = (b_1, \ldots, b_n)$ it is $Q_t := \{P_j \mid b_j = 1\}$.

It is not hard to see that $Q \subseteq Q'$ iff $t_Q \leq_C t_{Q'}$.

We are now equipped to define the translation α of expanded SGs $\mathcal{G}$ over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ into SGs $\alpha(\mathcal{G})$ over $\mathcal{S}$. Let $\mathcal{G} = (V, E, \ell)$. Then $\alpha(\mathcal{G}) := (V, E, \ell')$ where $\ell'(v) := ((b_1, \ldots, b_n), ref(v))$ with $b_i = 1$ iff $P_i \in typ(v)$, $1 \leq i \leq n$.

Just as for the translation β from SGs to expanded SGs, it is easy to see that φ is a homomorphism from $\mathcal{H}$ to $\mathcal{G}$ iff φ is a homomorphism from $\alpha(\mathcal{H})$ to $\alpha(\mathcal{G})$ w.r.t. $\mathcal{S}$. Consequently we have proven the equivalences proposed in item 1 on page 10.

4.2 Rooted Simple Graphs

In order to formalize the correspondence between subsumption of SGs and concept descriptions, respectively, we have to extend the notion of an expanded SG to so-called *rooted simple graphs*.

Definition 14 (Rooted Simple Graphs) *A graph $\mathcal{G} = (V, E, v_0, \ell)$ is called* rooted simple graph *over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ if $(V, E\ell)$ is an expanded SG over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ and $v_0 \in V$ is a distinguished node in $\mathcal{G}$ called the* root *of $\mathcal{G}$.*

Given an interpretation $\mathcal{I} = (\Delta, \cdot^I)$ of $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$, the semantics of a rooted SG $\mathcal{G} = (V, E, v_0, \ell)$ is given by $\{\delta \in \Delta \mid \mathcal{I} \models \Phi_r(\mathcal{G})\}$, where Φ_r is an extension of Φ_e to rooted SGs. To be more precise, the FO formula $\Phi_r(\mathcal{G})(x_0)$ with one free variable x_0 is obtained from $\mathcal{G}$ as follows. We distinguish two cases, namely (1) $ref(v_0) \in \mathcal{N}_I$ and (2) $ref(v_0) = *$. In the first case, we extend $\Phi_e(\mathcal{G})$ by a conjunct $x_0 = ref(v_0)$, where x_0 is a new variable. In the second case, we assume w.l.o.g. x_0 to be the variable corresponding to v_0. Then, $\Phi_r(\mathcal{G})(x_0)$ is obtained from $\Phi_e(\mathcal{G})$ by eliminating the existential quantifier binding x_0.

Example 15 (Example 12 continued) Consider the expanded SG $\mathcal{G}_1$ depicted on the left hand side in Figure 3. Considering $\mathcal{G}_1$ as a rooted SG with root v_0, we obtain the following FO formula $\Phi_r(\mathcal{G}_1)(x_0)$ with one free variable x_0:

$$\begin{aligned}\Phi_r(\mathcal{G}_1)(x_0) = \exists x_1 \,.\, &\mathsf{Female}(x_0) \wedge \mathsf{has\text{-}child}(\mathsf{Peter}, x_0) \\ &\wedge \mathsf{has\text{-}child}(x_0, x_1) \wedge \mathsf{likes}(x_0, x_1) \\ &\wedge \mathsf{Male}(x_1) \wedge \mathsf{Student}(x_1) \\ &\wedge \mathit{attends}(x_1, \mathsf{KR101}) \wedge \mathsf{CScourse}(\mathsf{KR101}).\end{aligned}$$

Remark 16 *Just as for SGs, a rooted simple graph $\mathcal{G}$ is said to be in* normal form*, if each constant $a \in \mathcal{N}_I$ occurs at most once as a referent of a node in $\mathcal{G}$. We can obtain a rooted SG in normal form from any rooted SG $\mathcal{G} = (V, E, v_0, \ell)$ by identifying all nodes having the same individual marker $a \in \mathcal{N}_I$ as their referent. The type of the resulting node is the union of the types of the identified nodes, and its referent is a. If v_0 has to be identified with other nodes, then the resulting node is again named v_0.*

The resulting rooted SG $\mathcal{G}' = (V', E', v_0, \ell')$ is equivalent to $\mathcal{G}$. This equivalence can be shown analogous to the equivalence result for expanded SGs proposed in Remark 10.

Subsumption of rooted SGs can be seen as a special case of subsumption of expanded SGs. Given two rooted SGs $\mathcal{G}$ and $\mathcal{H}$, the root of $\mathcal{H}$ must be mapped onto the root of $\mathcal{G}$. In other words, the roots of both graphs represent the same object. In the corresponding FO formulae, this object is represented by the variable x_0 occuring as the unique free variable in $\Phi_r(\mathcal{G})$ and $\Phi_r(\mathcal{H})$. Formally, a *homomorphism* from a rooted SG $\mathcal{H}$ to a rooted SG $\mathcal{G}$ is a mapping $\varphi : V_H \longrightarrow V_G$ that (1) fulfills the conditions on a homomorphism between expanded SGs (Definition 11) and (2) maps the root of $\mathcal{H}$ onto the root of $\mathcal{G}$, i.e., $\varphi(v_{0H}) = v_{0G}$. As before, we say that the rooted SG $\mathcal{H}$ *subsumes* the rooted SG $\mathcal{G}$ iff there exists a homomorphism from $\mathcal{H}$ to $\mathcal{G}$.

Remark 17 *Subsumption of rooted SGs can be seen as a special case of subsumption between expanded SGs, because whenever there exists a homomorphism φ from $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ to $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$, then φ also yields a homomorphism from (V_H, E_H, ℓ_H) to (V_G, E_G, ℓ_G). Consequently, subsumption between expanded SGs can be reduced to subsumption between rooted SGs as follows: $\mathcal{H} = (V_H, E_H, \ell_H)$ subsumes $\mathcal{G} = (V_G, E_G, \ell_G)$ iff for an arbitrary $w \in V_H$, there exists $v \in V_G$ such that (V_H, E_H, w, ℓ_H) subsumes (V_G, E_G, v, ℓ_G).*

Conversely, not every homomorphism φ from (V_H, E_H, ℓ_H) to (V_G, E_G, ℓ_G) yields a homomorphism from the rooted SG (V_H, E_H, w_0, ℓ_H) to the rooted SG (V_G, E_G, v_0, ℓ_G). For example, consider the rooted SGs depicted in Figure 4. Obviously, mapping w_0 to v_1 and w_1 to v_0 yields a homomorphism from the expanded SG $\mathcal{H}$ to the expanded SG $\mathcal{G}$. But if we assume v_0 to be the root of $\mathcal{G}$ and w_0 to be the root of $\mathcal{H}$, then there exists no homomorphism from $\mathcal{H}$ to $\mathcal{G}$.

Subsumption between rooted SGs is sound and complete w.r.t. the FO formulae induced by Φ_r.

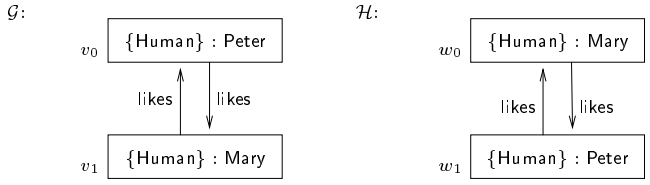

Figure 4: Homomorphisms between expanded and rooted simple graphs.

Theorem 18 *Let $\mathcal{G}$ be a rooted SG in normal form and $\mathcal{H}$ be a rooted SG. There exists a homomorphism from $\mathcal{H}$ to $\mathcal{G}$, iff $\Phi_r(\mathcal{G}) \models \Phi_r(\mathcal{H})$, i.e., $\forall x_0.\Phi_r(\mathcal{G})(x_0) \rightarrow \Phi_r(\mathcal{H})(x_0)$ is valid.*

Proof: We first show that $\Phi_r(\mathcal{G})(x_0) \models \Phi_r(\mathcal{H})(x_0) \Longrightarrow \mathcal{G} \sqsubseteq \mathcal{H}$. The idea behind the proof is as follows. We treat the free variable x_0 as a new constant c_0. Introducing c_0 as the referent of the roots in $\mathcal{G}$ and $\mathcal{H}$, respectively, we obtain two expanded simple graphs $\mathcal{G}_0$ and $\mathcal{H}_0$ such that $\Phi_e(\mathcal{G}_0) \models \Phi_e(\mathcal{H}_0)$. By Theorem 13, we obtain a homomorphism φ from $\mathcal{H}_0$ to $\mathcal{G}_0$. In the last step, we will show that φ also yields a homomorphism from $\mathcal{H}$ to $\mathcal{G}$.

Let $\mathcal{G} = (V_G, E_G, v_0, \ell_G)$ be a rooted SG in normal form and $\mathcal{H} = (V_H, E_H, w_0, \ell_H)$ a rooted SG over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$. Further, let c_0 be a new constant, i.e., $c_0 \notin \mathcal{N}_I$. $\Phi_r(\mathcal{G})(x_0) \models \Phi_r(\mathcal{H})(x_0)$ implies $\Phi_r(\mathcal{G})(c_0) \models \Phi_r(\mathcal{H})(c_0)$.

We define $\mathcal{G}_0 := (V_G, E_G, \ell_{G0})$ by $\ell_{G0}(v) := \ell_G(v)$ for all $v \in V_G \setminus \{v_0\}$ and $\ell_{G0}(v_0) := (typ_G(v_0), c_0)$. Let $\mathcal{H}_0 := (V_H, E_H, \ell_{H0})$ be defined in the same way. $\mathcal{G}_0$ and $\mathcal{H}_0$ are expanded SGs over $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I \cup \{c_0\})$. We want to show that $\Phi_r(\mathcal{G})(c_0) \equiv \Phi_e(\mathcal{G}_0)$. We distinguish two cases: (1) $ref_G(v_0) = *$ and (2) $ref_G(v_0) = a \in \mathcal{N}_I$. In both cases there exists a bijection π between the generic nodes in $\mathcal{G}_0$ and all generic nodes except v_0 in $\mathcal{G}$. Obviously, π yields a bijection between the sets of existentially quantified variables $\{z_1, \ldots, z_n\}$ in $\Phi_e(\mathcal{G}_0)$ and $\{y_1, \ldots, y_n\}$ in $\Phi_r(\mathcal{G})(c_0)$, respectively. In the first case, it is $\Phi_e(\mathcal{G}_0)[z_1/\pi(z_1), \ldots, z_n/\pi(z_n)] = \Phi_r(\mathcal{G})(c_0)$. In the second case we get $\Phi_r(\mathcal{G})(x_0) = (x_0 = a) \wedge \Phi_e(V_G, E_G, \ell_G)$. It is $\Phi_r(\mathcal{G})(c_0) = (c_0 = a) \wedge \Phi_e(V_G, E_G, \ell_G) \equiv \Phi_e(V_G, E_G, \ell_G)[a/c_0] \equiv \Phi_e(\mathcal{G}_0)$.

We now have shown that $\Phi_e(\mathcal{G}_0) \equiv \Phi_r(\mathcal{G})(c_0)$ and $\Phi_e(\mathcal{H}_0) \equiv \Phi_r(\mathcal{H})(c_0)$. By the premise, it follows $\Phi_e(\mathcal{G}_0) \models \Phi_e(\mathcal{H}_0)$.

By Theorem 13, there exists a homomorphism φ from $\mathcal{H}_0$ to $\mathcal{G}_0$. In order to prove $\mathcal{G} \sqsubseteq \mathcal{H}$, we show that φ also yields a homomorphism from $\mathcal{H}$ to $\mathcal{G}$. Therefore, it remains to show that (1) $\varphi(w_0) = v_0$ and (2) $\ell_G(v_0)$ is more specific than $\ell_H(w_0)$, i.e., $typ_G(v_0) \subseteq typ_H(w_0)$ and $ref_G(v_0) = ref_H(w_0)$ or $(ref_H(w_0) \in \mathcal{N}_I$ and $ref_G(v_0) = *)$.

1. It is $ref_{H0}(w_0) = c_0$. Since φ is a homomorphism, it is $ref_{G0}(\varphi(w_0)) = c_0$. v_0 is the only node with referent c_0 in $\mathcal{G}_0$. So, $\varphi(w_0) = v_0$.

2. Since φ is a homomorphism, it is $typ_H(w_0) \subseteq typ_G(v_0)$. In order to prove $ref_G(v_0) = ref_H(w_0)$ or $(ref_H(w_0) \in \mathcal{N}_I$ and $ref_G(v_0) = *)$ we have to

distinguish two cases.

(a) $ref_H(w_0) = *$. Due to the second disjunct in the condition, nothing has to be shown.

(b) $ref_H(w_0) = a \in \mathcal{N}_I$. Assume $ref_G(v_0) \neq a$.

First case: $ref_G(v_0) = *$. Then $\Phi_r(\mathcal{G})(x_0)$ contains no conjunct of the form $(x_0 = b)$, $b \in \mathcal{N}_I$. We derive a contradiction to $\Phi_r(G)(x_0) \models \Phi_r(\mathcal{H})(x_0)$ as follows. Let $(\mathcal{I}, \sigma)$ be a model of $\Phi_r(\mathcal{G})(x_0)$, i.e., $\mathcal{I} = (\Delta, \cdot^I)$ is an interpretation of $(\mathcal{N}_C, \mathcal{N}_R, \mathcal{N}_I)$ and σ is an $\mathcal{I}$-allocation such that $(\mathcal{I}, \sigma) \models \Phi_r(\mathcal{G})(x_0)$. Let $\xi := \sigma(x_0)$. We extend $(\mathcal{I}, \sigma)$ to $\mathcal{I}', \sigma')$ by adding a clone $\xi' \notin \Delta$ of ξ, i.e.,

- $\Delta' := \Delta \cup \{\xi'\}$,
- $P^{I'} := P^I \cup \{\xi'\}$, if $\xi \in P^I$, and $P^{I'} := P^I$, otherwise,
- $r^{I'} := r^I \cup \{\xi', \delta) \mid (\xi, \delta) \in r^I\} \cup \{(\delta, \xi') \mid (\delta, \xi) \in r^I\}$,
- $b^{I'} := b^I$ for all $b \in \mathcal{N}_I$, and
- $\sigma'(x_0) := \xi'$.

Then it is $(\mathcal{I}', \sigma') \models \Phi_r(\mathcal{G})(x_0)$, but since $\sigma'(x_0) \neq a^{I'}$, it is $(\mathcal{I}', \sigma') \not\models \Phi_r(\mathcal{H})(x_0)$.

Second case: $ref_G(v_0) = b \neq a$, $b \in \mathcal{N}_I$. As in the first case, we derive a contradiction by extending a model $(\mathcal{I}, \sigma)$ to $(\mathcal{I}', \sigma')$. Let $a^I = \xi$ and $\xi' \notin \Delta$. We define $\mathcal{I}'$ as above except of

- $b'^{I'} := b'^I$ for all $b' \in \mathcal{N}_I \setminus \{a\}$ and $a^{I'} := \xi'$, and
- $\sigma'(x_0) := b^I$.

As before, it is $(\mathcal{I}', \sigma') \models \Phi_r(\mathcal{G})(x_0)$, but since $\sigma'(x_0) \neq a^{I'}$, it is $(\mathcal{I}', \sigma') \not\models \Phi_r(\mathcal{H})(x_0)$.

Consequently, the assumption $ref_G(v_0) \neq a$ leads to a contradiction. Hence, $ref_G(v_0) = a$.

Overall, it follows that φ is a homomorphism from $\mathcal{H}$ to $\mathcal{G}$.

In the second part of the proof we have to show that $\mathcal{G} \sqsubseteq \mathcal{H} \Longrightarrow \Phi_r(\mathcal{G})(x_0) \models \Phi_r(\mathcal{H})(x_0)$ [3].

Let $\{y_1, \ldots, y_m\}$ and $\{x_1, \ldots, x_n\}$ be the existentially quantified variables in $\Phi_r(\mathcal{H})(x_0)$ and $\Phi_r(\mathcal{G})(x_0)$, respectively. Let $\Phi_r(\mathcal{G})(x_0) = \exists x_1 \ldots \exists x_n . \rho_G(x_0, x_1, \ldots, x_n)$ and $\Phi_r(\mathcal{H})(x_0) = \exists y_1 \ldots \exists y_m . \rho_H(x_0, y_1, \ldots, y_m)$.

Let $(\mathcal{I}, \sigma)$ be a model of $\Phi_r(\mathcal{G})(x_0)$. We have to show $(\mathcal{I}, \sigma) \models \Phi_r(\mathcal{H})(x_0)$. $(\mathcal{I}, \sigma) \models \Phi_r(\mathcal{G})(x_0)$ implies that there exist $\xi_1, \ldots, \xi_n \in \Delta$ such that $(\mathcal{I}, \sigma_G) \models \rho_G(x_0, x_1, \ldots, x_n)$, whereby $\sigma_G(x_0) := \sigma(x_0)$ and $\sigma_G(x_i) := \xi_i$ for $1 \leq i \leq n$.

We define an allocation σ_H such that $(\mathcal{I}, \sigma_H) \models \rho_H(x_0, y_1, \ldots, y_m)$ and $\sigma_H \mid_{\{x_0\}} = \sigma$. This implies $(\mathcal{I}, \sigma) \models \Phi_r(\mathcal{H})(x_0)$ and hence $\Phi_r(\mathcal{G})(x_0) \models \Phi_r(\mathcal{H})(x_0)$.

σ_H is defined as follows:

[3] For two FO formulae Φ_1, Φ_2 both having x_0 as their only free variable, $\Phi_1(x_0) \models \Phi_2(x_0)$ means $\forall x_0 . \Phi_1(x_0) \rightarrow \Phi_2(x_0)$ is valid.

$$\begin{array}{lcl} \sigma_H(x_0) & := & \sigma(x_0) \text{ and} \\ \sigma_H(y_i) & := & \begin{cases} \xi_j, \text{ if } id(w) = y_i \text{ and } id(\varphi(w)) = x_j, \\ a^I, \text{ if } id(w) = y_i \text{ and } id(\varphi(w)) = a \in \mathcal{N}_I. \end{cases} \end{array}$$

σ_H is well-defined and by definition, $\sigma_H \mid_{\{x_0\}} = \sigma$. Further, let $P(t)$ and $r(t_1, t_2)$ be atomic formulae in ρ_H. We have to show (1) $(\mathcal{I}, \sigma_H) \models P(t)$ and (2) $(\mathcal{I}, \sigma_H) \models r(t_1, t_2)$.

1. Let $t = id(w)$. Since φ is a homomorphism, it is $typ_H(w) \subseteq typ_G(\varphi(w))$. Hence, $P(id(\varphi(w))$ occurs in ρ_G. By the premise $(\mathcal{I}, \sigma_G) \models \rho_G$ and by the definition of σ_H we get $(\mathcal{I}, \sigma_H) \models P(t)$.

2. Let $t_i = id(w_i)$, $i = 1, 2$. Since φ is a homomorphism, it is $(\varphi(w_1), \varphi(w_2)) \in E_G$. Hence, $r(id(\varphi(w_1)), id(\varphi(w_2)))$ occurs in ρ_G. By the premise $(\mathcal{I}, \sigma_G) \models \rho_G$ and by the definition of σ_H we get $(\mathcal{I}, \sigma_H) \models r(t_1, t_2)$.

The items (1) and (2) imply $(\mathcal{I}, \sigma_H) \models \rho_H(x_0, y_1, \ldots, y_m)$ and hence, $(\mathcal{I}, \sigma) \models \Phi_r(\mathcal{H})(x_0)$.

This completes the proof of Theorem 18.

Deciding Subsumption of Rooted Simple Graphs

For SGs over a support $\mathcal{S}$, subsumption is known to be an NP-complete problem. The known algorithms deciding $g \sqsubseteq h$ w.r.t. $\mathcal{S}$ are based on the characterization of subsumption by homomorphisms, and thus require the subsumee g to be in normal form. In order to obtain a subsumption algorithm for rooted SGs, we must simply adjust the conditions tested for concept nodes and edges according to the modified conditions on homomorphisms between rooted SGs. Conversely, subsumption of SGs w.r.t. $\mathcal{S}$ can be reduced to subsumption of rooted SGs (see item 2 on page 10 and Remark 17). This shows that subsumption for rooted SGs is also an NP-complete problem.

In [MC93], a polynomial-time algorithm is introduced that can decide $g \sqsubseteq \mathbf{t}$ w.r.t. a support $\mathcal{S}$ provided that $\mathbf{t}$ is a tree and g is a SG in normal form. In this context, a SG $\mathbf{t}$ is called a *tree* iff $\mathbf{t}$ contains no cycles of length greater than 2. The notion of a tree can be adapted to rooted SGs $\mathcal{T}$ by viewing $\mathcal{T}$ as an *undirected graph*.

Note that in a rooted SG $\mathcal{T}$ that is a tree each concept node $v \in V \setminus \{v_0\}$ has a unique predecessor, though there may be more than one edge between them. For example, the node v_1 in the rooted SG $\mathcal{G}$ in Figure 4 has the root v_0 as its unique predecessor and the two edges between them yield a cycle of length 2 in $\mathcal{G}$.

A simple modification of the algorithm in [MC93] yields a polynomial time algorithm deciding $\mathcal{G} \sqsubseteq \mathcal{T}$ for a rooted SG $\mathcal{T}$ that is a tree and a rooted SG $\mathcal{G}$ in normal form. A formal description is given in Figure 5.

Complexity of the algorithm

The algorithm answers in time polynomial in the size of $\mathcal{G}$ and $\mathcal{T}$.

Input: A rooted SG $\mathcal{T}$ that is a tree and a rooted SG $\mathcal{G}$ in normal form.
Output: "yes", if there exists a homomorphism from $\mathcal{T}$ to $\mathcal{G}$, "no", otherwise.
Let $\mathcal{T} = (V_T, E_T, v_{0T}, \ell_T)$ and $\mathcal{G} = (V_G, E_G, v_{0G}, \ell_G)$. Further, let $\{v_1, \ldots, v_n\}$ be a post-order sequence of V_T, whereby v_1 is a leaf and $v_n = v_{0T}$. Define a labeling $\delta : V_G \longrightarrow \mathcal{P}(V_T)$ as follows.
Initialize δ by $\delta(w) := \emptyset$ for all $w \in V_G$.
For $1 \leq i \leq n$ do
  For all $w \in V_G$ do
    If $typ_T(v_i) \subseteq typ_G(w)$ and
      $ref_G(w) \leq_I ref_T(v_i)$ and
      for all $v_i r v \in E_T$ there is $w' \in V_G$ such that
        $v \in \delta(w')$ and $wrw' \in E_G$
    Then $\delta(w) := \delta(w) \cup \{v_i\}$
  od
od
If $v_{0T} \in \delta(v_{0G})$, then return "yes", else return "no".

Figure 5: Homomorphisms from trees to rooted simple graphs.

In order to define δ (see Figure 5), we have to consider all nodes of $\mathcal{G}$ for each node in $\mathcal{T}$, i.e., we have to test the condition in the inner loop $|V_T| \cdot |V_G|$ times. Testing the condition takes time polynomial in the size of $\mathcal{G}$ and $\mathcal{T}$.

Soundness and completeness of the algorithm

We have to show that there exists a homomorphism from $\mathcal{T}$ to $\mathcal{G}$ iff the algorithm depicted in Figure 5 applied on $\mathcal{T}$ and $\mathcal{G}$ answers "yes".
Completeness: Assume that there exists a homomorphism φ from $\mathcal{T}$ to $\mathcal{G}$. We have to show that the algorithm depicted in Figure 5 answers "yes".
Therefore, we prove by induction for all $m \in \{1, \ldots, n\}$ the following

Claim: Let φ be a homomorphism from $\mathcal{T} = (V_T, E_T, v_{0T}, \ell_T)$ to $\mathcal{G} = (V, E, v_{0G}, \ell)$. Further, let $\{v_1, \ldots, v_n\}$ be a post-order sequence of V_T. For all v_m, it is $v_m \in \delta(\varphi(v_m))$.

Basis step: $m = 1$
Let $w = \varphi(v_1) \in V_G$. Since φ is a homomorphism, it is $ref_G(w) \leq_I ref_T(v_1)$ and $typ_T(v_1) \subseteq typ_G(w)$. Furthermore, v_1 has no successors (because v_1 is a leaf in $\mathcal{T}$). Thus, all conditions in the if-statement of the algorithm are satisfied and hence, $\delta(w)$ is set to $\delta(w) \cup \{v_1\}$.

Induction step: $m - 1 \longrightarrow m$
By the induction hypothesis we have $v_i \in \delta(\varphi(v_i))$ for all $1 \leq i < m$. If $m > n$,

nothing has to be shown. Assume $m \leq n$. Let $w = \varphi(v_m)$. Since φ is a homomorphism, it is $ref_G(w) \leq_I ref_T(v_m)$ and $typ_T(v_m) \subseteq typ_G(w)$. Let $v_i \in V_T$ and $v_m r v_i \in E_T$. This implies $w r \varphi(v_i) \in E_G$. It is $i < m$, because $\{v_1, \ldots, v_n\}$ is a post-order sequence. By induction, it is $v_i \in \delta(\varphi(v_i))$. Since $v_m r v_i \in E_T$ has been chosen arbitrarily, all conditions in the if-statement of the algorithm are satisfied and hence, $\delta(w)$ is set to $\delta(w) \cup \{v_m\}$.

It is $v_n = v_{0T}$. The claim implies $v_{0T} \in \delta(\varphi(v_{0T})) = \delta(v_{0G})$. So, the algorithm answers "yes".

Soundness: Assume that the algorithm answers "yes". We have to show that there exists a homomorphism from $\mathcal{T} = (V_T, E_T, v_{0T}, \ell_T)$ to $\mathcal{G} = (V_G, E_G, v_{0G}, \ell_G)$.

Let $\{v_1, \ldots, v_n\}$ be the post-order sequence used by the algorithm to define δ. We prove the claim by defining $\varphi(v_i)$ top down, i.e., starting with the root $v_{0T} = v_n$ and ending with the leaf v_1. In each iteration step we maintain $v_i \in \delta(\varphi(v_i))$. Using this invariant, it is easy to see that φ is a homomorphism from $\mathcal{T}$ to $\mathcal{G}$.

We start with $\varphi(v_n) = \varphi(v_{0T}) := v_{0G}$. Since the algorithm answered "yes", it is $v_{0T} \in \delta(\varphi(v_{0G}))$. Now assume that $\varphi(v_n), \ldots, \varphi(v_{i+1})$, $n > i \geq 1$ are already defined. We have to define $\varphi(v_i)$. Since $\mathcal{T}$ is a tree, there exists a unique predecessor v_j of v_i. Since $\{v_1, \ldots, v_n\}$ is a post-order sequence, it is $i < j \leq n$. Thus, $\varphi(v_j)$ is already defined. Let $w = \varphi(v_j)$. By the invariant, we have $v_j \in \delta(\varphi(v_j))$. It follows that there exists $w' \in E_G$ such that $w r w' \in E_G$ and $v_i \in \delta(w')$. Define $\varphi(v_i) := w'$. By definition, we get $v_i \in \delta(\varphi(v_i))$.

This completes the proof of soundness and completeness of the algorithm.

In the next section, we will show that the polynomial-time algorithm can also be applied to decide subsumption of $\mathcal{ELIRO}^1$-concept descriptions.

5 The Relation between Simple Graphs and Description Logics

In this section, we extend the results on the correspondence between conceptual graphs and description logics presented in [CF98]. In [CF98], an equivalence result is only obtained for a small fragment of simple graphs and a weak description logic.

On the one hand, Faron and Coupey only consider trees, i.e., connected simple graphs consisting of concept nodes and binary relations that are trees. Further, they do not allow for individual markers in the label of concept nodes.

On the other hand, the authors only allow for concept descriptions built from primitive concepts, existential restrictions, primitive roles, inverse roles, and a restricted form of conjunction (see [CF98], Section 4.1 for details). Thus, the resulting description logic is a fragment of the description logic $\mathcal{ELI}$ allowing

for conjunction, existential restrictions, and inverse roles.

We are now concerned with the class of rooted SGs and the description logic $\mathcal{ELIRO}^1$. We will prove a one-to-one correspondence between $\mathcal{ELIRO}^1$-concept descriptions of a certain form and the class of rooted SGs that are trees. Thus, we extend the results of Faron and Coupey in the following sense: On the one hand, we allow for individual concept nodes[4] in the SGs. On the other hand, we cope with full conjunction in concept descriptions by considering sets of concept types in the labels of concept nodes. However, we only allow for $\mathcal{ELIRO}^1$-concept descriptions in which *each conjunction contains at most one constant.* This is due to the fact that referents of individual concept nodes in rooted SGs are single constants $a \in \mathcal{N}_I$.

Remark 19 *A conjunction of at least two ONE-OF-concept descriptions* $\{a_1\} \sqcap \ldots \sqcap \{a_n\}$ *occuring in a concept description* C *implies* $a_1^I = \ldots = a_n^I$ *for all models* $\mathcal{I}$ *of* C*. Though equality of individual markers cannot be expressed in (rooted) SGs, we can extend the results on subsumption of* $\mathcal{ELIRO}^1$*-concept descriptions of the restricted form given below to arbitrary* $\mathcal{ELIRO}^1$*-concept descriptions by extending the notion of rooted SGs to graphs which allow for set of constants as referents of concept nodes. In this work, however, we concentrate on the correspondence between SGs and DLs and therefore refrain from such an extension.*

5.1 Translating Concept Descriptions into Rooted Simple Graphs

The main idea underlying the translation is to represent a concept description C as a tree T_C. Intuitively, C is represented by a tree with root v_0 where all atomic concepts and constants occuring in the top-level conjunction of C yield the label of v_0, and each existential restriction $\exists r.C'$ in this conjunction yields an r-successor that is the root of the tree corresponding to C'.

Formally, $T_C = (V_C, E_C, v_0, \ell_C)$ is recursively defined as follows: Let $C = P_1 \sqcap \ldots \sqcap P_n \sqcap \exists r_1.C_1 \sqcap \ldots \sqcap \exists r_m.C_m$. where $P_i \in \mathcal{N}_I \cup \mathcal{N}_C$, $1 \leq i \leq n$, with at most one $P_j \in \mathcal{N}_I$.

If $\mathsf{depth}(C) = 0$ **then**

- $V_C := \{v_0\}$,
- $E_C := \emptyset$, and
- $\ell_C(v_0) := P_1, \ldots, P_n$.

If $\mathsf{depth}(C) > 0$ **then** for $1 \leq i \leq m$, let $\mathcal{G}_i = (V_i, E_i, v_{0i}, \ell_i)$ be the recursively defined trees corresponding to C_i where w.l.o.g. the V_i are pairwise disjoint. Define

[4] A concept node v is called *individual concept node* if $ref(v) \in \mathcal{N}_I$; otherwise, v is called *generic concept node.*

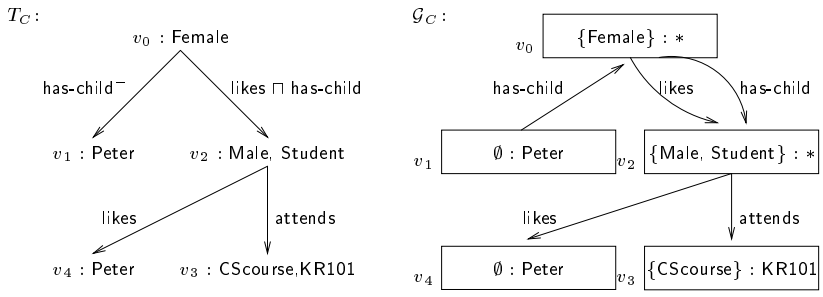

Figure 6: Translating $\mathcal{ELIRO}^1$-concept descriptions into rooted simple graphs.

- $V_C := \{v_0\} \cup \bigcup_{1 \leq i \leq m} V_i$,
- $E_C := \{v_0 r_i v_{0i} \mid 1 \leq i \leq m\} \cup \bigcup_{1 \leq i \leq m} E_i$,
- $\ell_C(v_0) := P_1, \ldots, P_n$ and for $v \in V_i$, define $\ell_C(v) := \ell_i(v)$.

Example 20 The concept description C below yields the tree T_C depicted on the left hand side in Figure 6:

$$C := \quad \mathsf{Female} \sqcap \exists \mathsf{has\text{-}child}^-.\{\mathsf{Peter}\} \sqcap \exists(\mathsf{likes} \sqcap \mathsf{has\text{-}child}).\\ (\mathsf{Male} \sqcap \mathsf{Student} \sqcap \exists \mathsf{attends}.(\mathsf{CScourse} \sqcap \{\mathsf{KR101}\}) \sqcap \exists \mathsf{likes}.\{\mathsf{Peter}\}).$$

Now, we can define the rooted SG $\mathcal{G}_C = (V, E, v_0, \ell)$ corresponding to C as follows. The nodes in T_C yield the set of concept nodes V of $\mathcal{G}_C$. The label $\ell(v)$ of a concept node $v \in V$ is determined by the label $\ell_T(v)$ of v in T_C, i.e., $typ(v)$ is the set of all atomic concepts occurring in $\ell_T(v)$ and, if there is a constant $a \in \ell_T(v)$, then $ref(v) := a$; otherwise $ref(v) := *$. Note that $ref(v)$ is well-defined because we have restricted $\mathcal{ELIRO}^1$-concept descriptions to those containing at most one constant in each conjunction. Finally, the set of edges of $\mathcal{G}_C$ is obtained from the edges in T_C: conjunctions of roles are decomposed ($v(r_1 \sqcap \ldots \sqcap r_n)w$ yields n edges $vr_1d, \ldots, vr_nw$) and inverse roles are redirected (vr^-w yields the edge wrv).

Example 21 (Example 20 continued) For the $\mathcal{ELIRO}^1$-concept description C from Example 20 we obtain the rooted SG $\mathcal{G}_C$ depicted on the right hand side in Figure 6, which is a tree with root v_0.

Using the recursive definition of T_C it can be shown that C is equivalent to its corresponding rooted SG $\mathcal{G}_C$.

Proposition 22 *Let C be an $\mathcal{ELIRO}^1$-concept description (of the restricted form) and $\mathcal{G}_C$ its corresponding rooted SG. Then C is equivalent to $\mathcal{G}_C$, i.e., $\forall x_0 : \psi(C)(x_0) \leftrightarrow \Phi_r(\mathcal{G}_C)(x_0)$ is a valid formula.*

Proof: In order to simplify the proof, we first introduce a slight modification of the FO formula $\Phi_r(\mathcal{G})(x_0)$ corresponding to a rooted SG $\mathcal{G} = (V, E, v_0, \ell)$.

Intuitively, we introduce a new variable x for each node $v \in V^1$ independent of the referent of v. If $ref^1(v) = a \in \mathcal{N}_I$, then we just introduce the literal $x \doteq a$.

Formally, we define $\Phi'_r(\mathcal{G})(x_0)$ as follows. We assign distinct variables to all nodes $v \in V$ referred to by $var(v)$. W.l.o.g. let $var(v_0) = x_0$. Then, each node v with $var(v) = x$ and label $\ell(v) = (\{P_1, \ldots, P_n\}, a)$ yields a formula $\bigwedge_{1 \leq i \leq n} P_i(x)$ if $a = *$; otherwise $\bigwedge_{1 \leq i \leq n} P_i(x) \wedge (x \doteq a)$. Each edge $vrw \in E$ yields an atomic formula $r(var(v), var(w))$. Finally, $\Phi_r(\mathcal{G})(x_0)$ is defined as the conjunction of all formulae corresponding to concept nodes and edges, respectively, whereby all variables except x_0 are existentially quantified. Hence, x_0 is the unique free variable in $\Phi'_r(\mathcal{G})(x_0)$.

Based on the equivalence $P_1(x) \wedge \ldots \wedge P_n(x) \wedge x \doteq a \equiv P_1(a) \wedge \ldots \wedge P_n(a)$, it is not hard to see that $\Phi_r(\mathcal{G})(x_0) \equiv \Phi'_r(\mathcal{G})(x_0)$. Thus, in order to prove Proposition 22, it is sufficient to show that $\Psi_C(x_0) \equiv \Phi'_r(\mathcal{G})(x_0)$.

In order to prove this claim, we first assume that w.l.o.g. all existentially quantified variables in $\Psi_C(x_0)$ are distinct. Now, we can show by induction on $\mathsf{depth}(C)$ that there exists a bijection π from the set of variables in $\Psi_C(x_0)$ to the set of variables in $\Phi'_r(\mathcal{G})(x_0)$ such that

- $\pi(x_0) = x_0$, and
- $\pi(\mathcal{C}(\Psi_C(x_0))) = \mathcal{C}(\Phi'_r(\mathcal{G}_C)(x_0))$, i.e., the set of atomic formulae occuring in $\Psi_C(x_0)$ is equal to the set of atomic formulae occuring in $\Phi'_r(\mathcal{G}_C)(x_0)$ except for renaming variables.

Due to the assumption on distinct variables in $\Psi_C(x_0)$ this implies

$$
\begin{array}{ll}
 & \Psi_C(x_0) \\
\equiv & \exists x_1 \ldots \exists x_t. \bigwedge_{P(x_i) \in \mathcal{C}(\Psi_C(x_0))} P(x_i) \\
 & \quad \wedge \bigwedge_{r(x_i, x_j) \in \mathcal{C}(\Psi_C(x_0))} r(x_i, x_j) \\
 & \quad \wedge \bigwedge_{x_i \doteq a \in \mathcal{C}(\Psi_C(x_0))} x_i \doteq a \\
\equiv & \exists \pi(x_1) \ldots \exists \pi(x_t). \bigwedge_{P(\pi(x_i)) \in \mathcal{C}(\Psi_C(x_0))} P(\pi(x_i)) \\
 & \quad \wedge \bigwedge_{r(\pi(x_i), \pi(x_j)) \in \mathcal{C}(\Psi_C(x_0))} r(\pi(x_i), \pi(x_j)) \\
 & \quad \wedge \bigwedge_{\pi(x_i) \doteq a \in \mathcal{C}(\Psi_C(x_0))} \pi(x_i) \doteq a \\
\equiv & \Phi'_r(\mathcal{G})(x_0)
\end{array}
$$

$\mathsf{depth}(C) = 0$**:** Then $C = P_1 \sqcap \ldots \sqcap P_n$. Obviously, $\pi = \{x_0 \mapsto x_0\}$ is a bijection and it is $\mathcal{C}(\Psi_C(x_0)) = \mathcal{C}(\Phi_r(\mathcal{G}_C)(x_0))$.

$\mathsf{depth}(C) > 0$**:** Then $C = P_1 \sqcap \ldots \sqcap P_n \sqcap \exists r_1.C_1 \sqcap \ldots \sqcap \exists r_m.C_m$. We have to distinguish two cases, namely (1) $P_i \in \mathcal{N}_C$ for all $1 \leq i \leq n$ and (2) $P_j \in \mathcal{N}_I$ for some $j \in \{1, \ldots, n\}$.

In the second case, the claim follows from (1) by just considering the conjunct $x_0 \doteq P_j$ instead of $P_j(x_0)$. Hence, we reduce our attention to the first case.

It is $\Psi_C(x_0) = P_1(x_0) \wedge \ldots \wedge P_n(x_0) \wedge \exists x_1.(\Psi_{r_1}(x_0, x_1) \wedge \Psi_{C_1}(x_1)) \wedge \exists x_m.(\Psi_{r_m}(x_0, x_m) \wedge \Psi_{C_m}(x_1))$.

By definition of T_C and$\mathcal{G}_C$, there exist $v_1, \ldots, v_m \in V$ such that

- the subtree $\mathcal{G}_C(v_i)$ with root v_i of $\mathcal{G}_C$ is the recursively defined tree corresponding to C_i, and
- the set of edges between v_0 and v_i corresponds to the role term r_i, i.e., $r_i = \bigsqcap_{v_0 s v_i \in E} s \sqcap \bigsqcap_{v_i s v_0 \in E} s^-$.

The induction hypothesis yields a bijection π_i from the set of variables in $\Psi_{C_i}(x_i)$ to the set of variables in $\Phi'_r(\mathcal{G}_C(v_i))(x_i)$ for each $1 \leq i \leq m$ such that $\pi_i(x_i) = x_i$ and $\pi(\mathcal{C}(\Psi_{C_i}(x_i))) = \mathcal{C}(\Phi'_r(\mathcal{G}_C(v_i))(x_i))$.

Note that we assume w.l.o.g. that x_i is the free variable in $\Phi_r(\mathcal{G}_C(v_i)(x_i)$ and also the variable corresponding to v_i in $\mathcal{G}_C$. Thereby, the resulting bijection is the identic mapping id, i.e., $id(x) = x$ for all variables x. In general, however, one may introduce different variables in the formulae corresponding to subconcepts C_i in C and corresponding subtrees $\mathcal{G}_{C_i}$ in $\mathcal{G}_C$.

We define $\pi := \{x_0 \mapsto x_0\} \cup \bigcup_{1 \leq i \leq m} \pi_i$.

It remains to show that the set of atomic formulae in which the variable x_0 occurs coincides for $\Psi_C(x_0)$ and $\Phi'_r(\mathcal{G}_C)(x_0)$ w.r.t. π.

For each conjunct s in r_i, it is $s(x_0, x_i)$ an atomic formula in $\Psi_{r_i}(x_0, x_i)$, and, by definition of $\mathcal{G}_C$, $v_0 s v_i \in E$ and hence $s(x_0, x_i)$ is an atomic formula in $\Phi'_r(\mathcal{G}_C)(x_0)$. Analogously, $s(x_i, x_0)$ occurs in $\Psi_{r_i}(x_0, x_i)$ iff $s(x_0, x_i)$ occurs in $\Phi'_r(\mathcal{G}_C)(x_0)$ for all $1 \leq i \leq m$.

Further, it is $\ell(v_0) = (\{P_1, \ldots, P_n\}, *)$ and thus, $P_i(x_0)$ occurs in $\Phi'_r(\mathcal{G}_C)(x_0)$ for all $1 \leq i \leq n$.

So, $\pi(\mathcal{C}(\Psi_C(x_0))) = \mathcal{C}(\Phi'_r(\mathcal{G}_C)(x_0))$. □

It is not hard to see that we can translate a rooted SG $\mathcal{G} = (V, E, v_0, \ell)$ whose edges yield a tree with root v_0 into an equivalent concept description C_G by just considering $\mathcal{G}$ as the tree of C_G. Formally, C_G is recursively defined as follows.

$|V| = 1$: Then it is $V = \{v_0\}$ and $E = \emptyset$. Let $\ell(v_0) = (\{P_1, \ldots, P_n\}, a)$. C_G is defined by

$$C_G := \begin{cases} a \sqcap P_1 \sqcap \ldots \sqcap P_n & \text{, if } a \in \mathcal{N}_I, \\ P_1 \sqcap \ldots \sqcap P_n & \text{, if } a = *. \end{cases}$$

$|V| > 1$): Let $\ell(v_0) = (\{P_1, \ldots, P_n\}, a)$. Let $\{v_1, \ldots, v_n\}$ be the set of all successors of v_0. We define the set of all role terms labeling edges between v_0 and v_i by $R_i := \{r \in \mathcal{N}_R \mid v_0 r v_i \in E\} \cup \{r^- \mid r \in \mathcal{N}_R, v_i r v_0 \in E\}$.

Further, let $\mathcal{G}_i$ be the subtree with root v_i of $\mathcal{G}$ and C_i the recursively defined $\mathcal{ELIRO}^1$-concept description of $\mathcal{G}_i$. Then, C_G is defined by

$$C_G := \begin{cases} a \sqcap P_1 \sqcap \ldots \sqcap P_n \sqcap \bigsqcap_{1 \leq i \leq n} \exists(\bigsqcap_{r \in R_i} r).C_i & \text{, if } a \in \mathcal{N}_I, \\ P_1 \sqcap \ldots \sqcap P_n \sqcap \bigsqcap_{1 \leq i \leq n} \exists(\bigsqcap_{r \in R_i} r).C_i & \text{, if } a = *. \end{cases}$$

The above translations yield a one-to-one correspondence between $\mathcal{ELIRO}^1$-concept descriptions and rooted SGs that are trees. Because of this correspondence, we can reduce subsumption in $\mathcal{ELIRO}^1$ to subsumption between rooted SGs.

5.2 Deciding Subsumption in $\mathcal{ELIRO}^1$

Using the translation introduced in the previous section, we can reduce subsumption of $\mathcal{ELIRO}^1$-concept descriptions to subsumption between rooted simple graphs. Thereby the rooted SG corresponding to the subsumer is a tree and the one of the subsumee can be transformed into an equivalent rooted SG in normal form. Thus, we can apply the polynomial-time algorithm and hence obtain the following complexity result for subsumption of $\mathcal{ELIRO}^1$-concept descriptions.

Theorem 23 *Subsumption $C \sqsubseteq D$ of $\mathcal{ELIRO}^1$-concept descriptions can be decided in time polynomial in the size of C and D.*

Proof: As already mentioned, we restricted ourselves to $\mathcal{ELIRO}^1$-concept descriptions where each conjunction contains at most one constant. Therefore, at first sight, the following argument only holds for this fragment of $\mathcal{ELIRO}^1$. But as already mentioned in Remark 19, one can easily extend the result to arbitrary $\mathcal{ELIRO}^1$-concept descriptions by allowing for sets of constants as referents of concept nodes. We leave this extension as an exercise to the interested reader.

The proof of the tractability result for $\mathcal{ELIRO}^1$-concept descriptions of the restricted form yields the rewards of our labour since we only have to gather up the results of the previous sections.

Given two $\mathcal{ELIRO}^1$-concept descriptions C, D, we first determine the rooted SGs $\mathcal{G}_C$ and $\mathcal{G}_D$ corresponding to C and D in polynomial time. $\mathcal{G}_C$ can then be tranformed (in polynomial time) into an equivalent rooted SG $\mathcal{G}'_C$ in normal form. Now we can apply the polynomial-time algorithm depicted in Figure 5 on $\mathcal{G}_D$ and $\mathcal{G}'_C$. If it answers "yes", then "$C \sqsubseteq D$"; otherwise, "$C \not\sqsubseteq D$".

Obviously, we just described a polynomial-time algorithm. Furthermore, the following equivalences imply that this algorithm decides subsumption $C \sqsubseteq D$ of two $\mathcal{ELIRO}^1$-concept descriptions (of the restricted form).

	$C \sqsubseteq D$	(Section 2)
iff	$\Psi_C(x_0) \models \Psi_D(x_0)$	(Proposition 22, Remark 16)
iff	$\Phi(\mathcal{G}'_C)(x_0) \models \Phi(\mathcal{G}_D)(x_0)$	(Theorem 18)
iff	exists a homomorphism φ from $\mathcal{G}_D$ to $\mathcal{G}'_C$	
iff	$\mathcal{G}'_C \sqsubseteq \mathcal{G}_D$.	

□

5.3 Extending the Tractability Result

As a by-product of the above tractability result for $\mathcal{ELIRO}^1$, we will now extend the tractability result from (rooted) SGs that are trees to (rooted) SGs that can be transformed into equivalent trees by "cutting cycles" of length greater than 2. For a given rooted SG $\mathcal{G}$, we can eliminate an (undirected) cycle $w_0, \ldots, w_n$ where $w_0 = w_n$ in $\mathcal{G}$ by applying the *split-operation* on concept nodes as introduced for SGs in [CM92]. To be more precise, we (1) arbitrarily choose a node $w_i \in \{w_1, \ldots, w_n\}$, (2) introduce a new node v labeled like w_i, and (3) replace all edges between w_{i-1} and w_i by edges between w_{i-1} and w. We then say that the cycle is *cut in* w_i. Obviously, any cyclic SG $\mathcal{G}$ can be transformed into an acyclic SG $\mathcal{G}^*$ by applying this operation a polynomial number of times. In general, however, the resulting SG $\mathcal{G}^*$ need not be equivalent to $\mathcal{G}$.

Example 24 (Example 15 continued) Consider the rooted SG $\mathcal{G}_1$ in Figure 3. On the one hand, we can eliminate the cycle v_1, v_0, v_2, v_1 by introducing a new node v_4 with label $(\emptyset, \mathsf{Peter})$ and replacing the edge $v_2\mathsf{likes}v_1$ by $v_2\mathsf{likes}v_4$. The resulting tree coincides with the tree $\mathcal{G}_C$ in Figure 6, and it is equivalent to $\mathcal{G}_1$ because $\mathcal{G}_1$ is a normal form of $\mathcal{G}_C$. On the other hand, if we introduce a new node v labeled $(\{Female\}, *)$ and replace $v_1\mathsf{has\text{-}child}v_0$ by $v_1\mathsf{has\text{-}child}v$, then the resulting tree is not equivalent to $\mathcal{G}_1$ because the student's mother and Peter's child need no longer to be the same person.

The following proposition introduces a condition on rooted SGs that ensures that rooted SGs satisfying this condition can be transformed into equivalent trees by applying the split operation to individual concept nodes.

Proposition 25 *If each cycle of length greater than 2 in the rooted SG $\mathcal{G}$ contains at least one individual concept node v, then $\mathcal{G}$ can be transformed into an equivalent tree $\mathcal{G}^*$ in time polynomial in the size of $\mathcal{G}$.*

Proof: In order to prove the proposition, we first formalize the cutting operation described above for the special case of cycles containing at least one individual node. The resulting transformation rule is defined in Figure 7.

Now, let $\mathcal{G} = (V, E, v_0, \ell)$ be a rooted SG. Each iterated application of the transformation rule depicted in Figure 7 terminates, because each application $\mathcal{G} \longrightarrow \mathcal{G}'$ decreases the number of cycles containing at least one individual node. Since the number of these cycles can be bounded by the number of edges leading to or from individual concept nodes, each iterated application terminates.

The termination argument also yields a polynomial upper bound on the size of $\mathcal{G}^*$. A cycle in $\mathcal{G}$ containing at least one individual node is eliminated by introducing a new node. The number of edges leading to or from an individual node (and hence the number of new nodes) is polynomial in the size of $\mathcal{G}$. So, $\mathcal{G}^*$ is obtained in time polynomial in the size of $\mathcal{G}$.

Finally, the resulting tree $\mathcal{G}^*$ is equivalent to $\mathcal{G}$ because $\mathcal{G}$ is a normal form of $\mathcal{G}^*$. □

Consequently, $\mathcal{G} \sqsubseteq \mathcal{H}$ can be decided in polynomial time if $\mathcal{G}$ is a rooted SG in normal form and $\mathcal{H}$ satisfies the premise of Proposition 25. It is easy to see

Transformation Rule: $\mathcal{G} \longrightarrow \mathcal{G}'$.
Let $\mathcal{G} = (V, E, v_0, \ell)$ be a rooted SG and $(w_0, w_1, \ldots, w_n)$, $n > 1$, a cycle of length greater 2 in $\mathcal{G}$.
Let w_i, $i \in \{1, \ldots, n\}$ be an individual concept node on the cycle and let v be a new node, i.e., $v \notin V$. Then derive $\mathcal{G}' = (V', E', v_0, \ell')$ from $\mathcal{G}$ by

- $V' := V \cup \{v\}$,
- $E' := \big(E \setminus (\{w_{i-1} r w_i \mid r \in \mathcal{N}_R\} \cup \{w_i r w_{i-1} \mid r \in \mathcal{N}_R\})\big) \cup \{v r w_i \mid w_{i-1} r w_i \in E\} \cup \{w_i r v \mid w_i r w_{i-1} \in E\}$, and
- $\ell'(w) := \ell(w)$ for all $w \in V$ and $\ell'(v) := \ell(w_i)$.

Figure 7: Eliminating cycles in rooted simple graphs.

that this tractability result also applies to (non-rooted) SGs over a support $\mathcal{S}$. More precisely, given a SG g over $\mathcal{S}$ in normal form and a SG h over $\mathcal{S}$ such that each cycle of length greater 2 in h contains at least one individual concept node, then (1) h can be transformed into an equivalent SG over $\mathcal{S}$ by an iterated application of the transformation rule given in Figure 7 (in polynomial time), and (2) $g \sqsubseteq h$ w.r.t. $\mathcal{S}$ can be decided in polynomial time applying the algorithm introduced in [MC93].

6 Conclusion

We have used a tractability result from SGs to identify a DL for which subsumption is tractable. Thereby, we have introduced the class of so-called expanded SGs and rooted SGs.

Within expanded SGs we consider sets of arbitrary concept types as the types of concept nodes. This allows for encoding a support explicitely in the graphs, but since sets of concept types can be encoded into a support, expanded SGs are not more expressive than SGs defined over a support $\mathcal{S}$.

Subsumption of expanded and rooted SGs has been characterized by homomorphisms from the subsumer to the subsumee.

As a by-product of the translation of concept descriptions into rooted SGs, we extended the tractability result from trees to SGs that can be transformed into equivalent trees by cutting cycles.

In further work, we will use the characterization of subsumption by homomorphisms between trees as a basis for new results on non-standard inferences in DLs [BK98, BKM98].

References

[BK98] F. Baader and R. Küsters. Computing the least common subsumer and the most specific concept in the presence of cyclic $\mathcal{ALN}$-concept descriptions. In O. Herzog and A. Günter, editors, *Proceedings of the 22nd Annual German Conference on Artificial Intelligence (KI'98)*, volume 1504 of *Lecture Notes in Computer Science*, pages 129–140, Bremen, Germany, 1998. Springer–Verlag.

[BKM98] F. Baader, R. Küsters, and R. Molitor. Computing least common subsumer in description logics with existential restrictions. LTCS-Report 98-09, LuFG Theoretical Computer Science, RWTH Aachen, Germany, 1998. See http://www-lti.informatik.rwth-aachen.de/Forschung/Papers.html.

[Bor96] A. Borgida. On the relative expressiveness of description logics and predicate logics. In *Artificial Intelligence, An International Journal*, volume 82, pages 353–367. 1996.

[CF98] P. Coupey and C. Faron. Towards correspondences between conceptual graphs and description logics. In *Proceedings of the sixth International Conference on Conceptual Structures (ICCS'98)*, number 1453 in Lecture Notes in Artificial Intelligence, pages 165–178, Montpellier, France, August 1998. Springer Verlag.

[CM92] M. Chein and M.L. Mugnier. Conceptual graphs: Fundamental notions. *Revue d'Intelligence Artificielle*, 6(4):365–406, 1992.

[CMS98] M. Chein, M. L. Mugnier, and G. Simonet. Nested graphs: A graph-based knowledge representation model with fol semantics. In A.G. Cohn, L.Schubert, and S.C. Shapiro, editors, *Proceedings of the 6th International Conference on Knowledge Representation (KR'98)*, pages 524–534, Trento, Italy, June 1998. Morgan Kaufman Publishers Inc.

[DLNN97] F.M. Donini, M. Lenzerini, D. Nardi, and W. Nutt. The complexity of concept languages. *Information and Computation*, 134(1):1–58, 1997.

[DLNS96] F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Reasoning in description logics. In G. Brewka, editor, *Foundation of Knowledge Representation*, pages 191–236. CSLI-Publications, 1996.

[KS97] G. Kerdiles and E. Salvat. A sound and complete CG proof procedure combining projections with analytic tableaux. In D. Lukose, H. Delugach, M. Keeler, L. Searle, and J. Sowa, editors, *Proceedings of the 5th Internation Conceference on Conceptual Structures (ICCS'97)*, volume 1257 of *Lecture Notes in Computer Science*, pages 371–385. Springer, 1997.

[MC93] M.L. Mugnier and M. Chein. Polynomial algorithms for projection and matching. *Lecture Notes in Computer Science*, 754:239–251, 1993.

[Sow84] J.F. Sowa. *Conceptual Structures – Information, Processing in Mind and Machine.* Addison Wesley, 1984.

[Wer95] M. Wermelinger. Conceptual graphs and first-order logic. In G. Ellis, R. Levinson, W. Rich, and J. Sowa, editors, *Proceedings of the third International Conference on Conceptual Structures, (ICCS'95)*, number 954 in Lecture Notes in Artificial Intelligence, pages 323–337. Springer-Verlag, 1995.