A tree decomposition algorithm for Conceptual Graph projection

Madalina Croitoru and **Ernesto Compatangelo**
Department of Computing Science
University of Aberdeen (UK)

Abstract

This paper discusses combinatorial mechanisms for reasoning with conceptual graphs. We focus on the combinatorial aspects of a backtracking approach to the NP-hard problem of projection, which is the main tool for reasoning with conceptual graphs. We use effective graph-theoretical look-ahead procedures, based on a conceptual forest decomposition of the graph to be projected. We believe that our approach to projection can improve the practical applicability of exponential algorithms currently used to solve NP-hard problems.

Introduction

Conceptual graphs (CGs) are a visual knowledge representation formalism where information is encoded using a bipartite graph [11, 12], while their semantics can be mapped into that of first order logic [3]. Roughly speaking, a CG corresponds to the ABox component of a Declarative Knowledge Base (DKB) represented using a frame-based model (e.g., a Description Logic). The TBox equivalent of a DKB expressed using conceptual graphs, which is called the CG *support*, is managed separately. By exploiting their visual assets, conceptual graphs can be used for addressing real-world interoperability issues [5, 8]. Reasoning with conceptual graphs is based on projection [11]. This labelled graph homomorphism defines a generalisation-specialisation relation [4] such that there is a projection from a CG G to a CG H if G is more general than H ($G \geq H$).

Unfortunately, deciding whether, given two conceptual graphs G and H, $G \geq H$ is an NP-complete problem [3, 1]. However, it has been shown that this problem is polynomially equivalent to other problems such as conjunctive query containment [2] and query output in databases [7], constraint satisfaction in combinatorial optimisation, and clause subsumption in knowledge representation and reasoning [7]. Consequently, algorithms of exponential complexity with fast execution time have been used in practical applications when the size of the graphs involved is not too large [10].

We have devised a new structure, called the matching graph, which exploits the combinatorial nature of graph homomorphism checking [6]. Following other approaches to projection [13], we relate its checking to the maximum clique problem. Our work is motivated by the belief that efficient graph theoretical techniques must be employed to enable the practical usability of projection as a reasoning mechanism. This paper details the combinatorial aspects of utilising effective graph-theoretical look-ahead procedures for projection algorithms.

The presentation is organised as follows. We introduce the main notions used throughout the paper and present the matching graph. The algorithm for the construction of reduced matching graphs, and the combination lemma follow. We then describe our novel 'conceptual forest' projection checking procedure. This algorithm is used to derive our method for the subsumption checking problem.

Conceptual graphs and projection

In this section, we briefly discuss and formalise a number of CG-related notions — namely background knowledge (support), simple CGs, and projection — which are needed to articulate the notion of matching graph in the next section. These definitions follow the notations introduced in [3].

Definition 1 (Bipartite graph) *A graph $G = (V_G, E_G)$ with the nodes set $V_G = V_C \cup V_R$, where V_C and V_R are finite disjoint nonempty sets, and each edge $e \in E_G$ is a two element set $e = \{v_C, v_R\}$, where $v_C \in V_C$ and $v_R \in V_R$.*

V_C is the set of *concept vertices* of G and V_R is the set of *relation vertices* of G. E_G is a multiset – multiple edges linking the same pair of vertices are allowed. Usually, a bipartite graph G is denoted as $G = (V_C, V_R; E_G)$. The number of edges incident with a node $v \in V(G)$ is the degree, $d_G(v)$, of the node v. An *isolated vertex* is a vertex v with $d_G(v) = 0$.

Let $G = (V_C, V_R; E_G)$ be a bipartite graph. If, for each $v_R \in V_R$ there is a linear order $e_1 = \{v_R, v_1\}, \ldots, e_k = \{v_R, v_k\}$ on the set of edges incident to v_R — where $k = d_G(v)$, then G is called an *ordered bipartite graph.*

As a simple way of stating an ordered bipartite graph G, we can provide a labelling $l : E_G \rightarrow \{1, \ldots, |V_C|\}$, where $l(\{v_R, w\})$ is the index of the edge $\{v_R, w\}$ in the above ordering of the edges incident in G to v_R. The label l is called the *order labelling* of the edges of G.

An ordered bipartite graph is denoted by $G = (V_C, V_R; E_G, l)$. For a vertex $v \in V_C \cup V_R$, $N_G(v)$ de-

notes the *neighbours set* of v, i.e. $N_G(v) = \{w \in V_C \cup V_R | \{v, w\} \in E_G\}$. Similarly, if $A \subseteq V_R \cup V_C$, its *neighbours set* is denoted as $N_G(A) = \cup_{v \in A} N_G(v) - A$. For each $r \in V_R$, $N^i_G(r)$ denotes the i-th neighbour of r, i.e. $v = N^i_G(r)$, iff $\{r, v\} \in E_G$ and $l(\{r, v\}) = i$.

In this paper we restrict our focus to a particular type of subgraph of a bipartite graph: $G^1 = (V^1_C, V^1_R; E^1_G)$ is a subgraph of $G = (V_C, V_R; E_G)$ if $V^1_R \subseteq V_R$, $V^1_C = N_G(V^1_R)$ and $E^1_G = \{ \{v, w\} \in E_G | v \in V^1_C, w \in V^1_R\}$. In other words, we require that the (ordered) set of all edges incident in G to a vertex from V^1_R must appear in G^1. Therefore the subgraph G^1 is completely specified by its relation vertex set and will be denoted as $\left[V^1_R\right]_G$. As we only consider bipartite graphs without isolated concept vertices, then $G = \left[V_R\right]_G$. Moreover, if $A, B \subseteq E_R$ then the usual union $[A]_G \cup [B]_G$ of the graphs $[A]_G$ and $[B]_G$ is the subgraph $[A \cup B]_G$. Hence, if $V_R = \{r_1, \ldots, r_m\}$ and G_i is the (star) graph $\left[\{r_i\}\right]_G$ $(1 \leq i \leq m)$, then $G = G_1 \cup \ldots \cup G_m$.

Background knowledge (i.e. ontological knowledge consisting of entities organised as a partially ordered set) is encoded in a structure called *support*. The support is implicitly used in the representation of *factual knowledge* as an ordered bipartite labelled graph.

Definition 2 ***(Support)***
A support is a 4-tuple $S = (T_C, T_R, \mathcal{I}, *)$ where

- *T_C is* poset*, i.e. a finite partially ordered set $(T_C, \leq)$ of concept types defining a type hierarchy. $\forall x, y \in T_C$ $x \leq y$ means that x is a subtype of y. The hierarchy has a topmost element $\top_C$.*
- *T_R is a finite set of relation types partitioned into k posets $(T^i_R, \leq)_{i=1,k}$ of relation types with arity i ($1 \leq i \leq k$), where k is the maximum arity of a relation type in T_R. Each relation type $r \in T^i_R$ of arity i has a* signature *$\sigma(r) \in \underbrace{T_C \times \ldots \times T_C}_{i \text{ times}}$, specifying the maximum concept type of each of its arguments. The notation $r(x_1, \ldots, x_i)$ means that x_j is a concept with $type(x_j) \leq \sigma(r)_j$ $(1 \leq j \leq i)$. The partial orders on relation types of the same arity must be signature compatible, i.e. $\forall r_1, r_2 \in T^i_R$ $r_1 \leq r_2 \Rightarrow \sigma(r_1) \leq \sigma(r_2)$.*
- *$\mathcal{I}$ is a countable set of individual markers used to refer to specific concepts, while $*$ is the generic marker used to refer to an unspecified concept of a specified type. The four sets T_C, T_R, $\mathcal{I}$, and $\{*\}$ are mutually disjoint and the union set $\mathcal{I} \cup \{*\}$ is partially ordered by $x \leq y$ if and only if $x = y$ or $y = *$.*

A conceptual graph is a structure that depicts factual information about the background knowledge contained in its support. This information is presented in a visual manner as an ordered bipartite graph, whose nodes have been labelled with elements from the support.

Definition 3 ***(Simple CGs)***
A simple conceptual graph is a 3-tuple $SG = [S, G, \lambda]$ where:

- *$S = (T_C, T_R, \mathcal{I}, *)$ is a support;*
- *$G = (V_C, V_R; E_G, l)$ is an ordered bipartite graph;*
- *λ is a labelling of the nodes in G with elements from the support S, where $\forall r : r \in V_R,\ \lambda(r) \in T_R^{d_G(r)}$; $\forall c : c \in V_C,\ \lambda(c) \in T_C \times \left(\mathcal{I} \cup \{*\}\right)$ such that if $c = N^i_G(r)$, $\lambda(r) = t_r$ and $\lambda(c) = (t_c, ref_c)$ then $t_c \leq \sigma_i(r)$.*

Therefore, each relation node r is labelled by a *relation type* denoted as $type(c)$ with an associated signature of arity given by the degree of the node r in G. Each concept node c is labelled by a couple $(type(c), ref(c))$, where $type(c)$ is the *type* of the node c and $ref(c)$ is the *referent* of c. This referent either belongs to $\mathcal{I}$ (when c is said to be an *individual concept node*), or is the generic marker $*$ (when c is said to be a *generic concept node*).

When the support is implicit, a SCG is denoted by the pair $SG = (G, \lambda)$. If $SG = (G, \lambda)$ is a SCG and G_1 is a bipartite subgraph of G, then $SG_1 = (G_1, \lambda_1)$, where λ_1 is the restriction of the labelling function λ to the vertices of G_1, is a *conceptual subgraph* of SG.

The subsumption relation, which is called projection in the context of conceptual graphs, is the main tool for reasoning with simple CGs. It corresponds to deduction for the existential conjunctive and positive fragment of first order logic [2].

Definition 4 ***(Projection)***
If $SG = (G, \lambda_G)$ and $SH = (H, \lambda_H)$ are two simple conceptual graphs defined on the same support S, then a projection *$SG \to SH$ is a mapping $\pi : V_C(G) \cup V_R(G) \to V_C(H) \cup V_R(H)$ where:*

- *$\pi(V_C(G)) \subseteq V_C(H)$ and $\pi(V_R(G)) \subseteq V_R(H)$;*
- *$\forall c \in V_C(G)$ and $\forall r \in V_R(G)$, if $c = N^i_G(r)$ then $\pi(c) = N^i_H(\Pi(r))$;*
- *$\forall v \in V_C(G) \cup V_R(G)$, $\lambda_G(v) \geq \lambda_H(\pi(v))$.*

The set of all projections from SG to SH will be denoted as $\mathbf{\Pi}_{G \to H} = \{\pi | \pi \text{ is a projection from } SG \text{ to } SH\}$. If $\mathbf{\Pi}_{G \to H} \neq \emptyset$ then SG is said to *subsume* SH; symbolically, this is denoted as $SG \geq SH$.

The *subsumption relation* is a pre-order on the set of all the SCGs defined on the same support. A semantics Φ is provided in [11], which maps each SCG G based on a support S into a first order logic formula $\Phi(G)$. If $\Phi(S)$ is the set of formulas associated to S, then for any two SCGs G and H defined on S, if $G \geq H$ then $\Phi(S), \Phi(H) \models \Phi(G)$ (*soundness*). Conversely, *Completeness* (in formulae, if $\Phi(S), \Phi(H) \models \Phi(G)$ then $G \geq H$) only holds if H is in *normal form*, i.e. if each individual marker appears at most once in concept node labels [9].

Although subsumption checking is an NP-complete problem [3, 1], engineering a backtracking algorithm could be computationally worthwhile. Hence, below we introduce our engineering approach to the problem.

The matching graph manouevre

Suppose that we want to test if $SG \geq SH$, where $SG = (G, \lambda_G)$ and $SH = (H, \lambda_H)$ are two simple conceptual graphs defined on the same support S. If $SG_1 = (G_1, \lambda_{G_1})$ is a conceptual subgraph of SG, then $\pi|_{V(G_1)}$ (the restriction of each $\pi \in \mathbf{\Pi}_{G\to H}$ to vertices of G_1) is a projection of SG_1 to SH.

Moreover, if (A, B) is a partition of the set of relation vertices of G and therefore $G = [A]_G \cup [B]_G$, then, for every $\pi \in \mathbf{\Pi}_{G\to H}$, we have $\pi|_{A\cup N_G(A)} \in \mathbf{\Pi}_{[A]_G\to H}$ and $\pi|_{B\cup N_G(B)} \in \mathbf{\Pi}_{[B]_G\to H}$. Clearly, if $v \in N_G(A) \cap N_G(B)$ then $\pi|_{A\cup N_G(A)}(v) = \pi|_{B\cup N_G(B)}(v)(= \pi(v))$. The following lemma holds.

Lemma 1 (Combination lemma)
If $SG = (G, \lambda_G)$ and $SH = (H, \lambda_H)$ are two simple conceptual graphs defined on the same support S and if $G = [A]_G \cup [B]_G$, where (A, B) is a partition of the set of relation vertices of G, then
$\mathbf{\Pi}_{G\to H} = \{\pi : V(G) \to V(H) | \exists \pi_A \in \mathbf{\Pi}_{[A]_G\to H}, \exists \pi_B \in \mathbf{\Pi}_{[B]_G\to H}$ *such that* $\pi|_A = \pi_A,\ \pi|_B = \pi_B$ and $\forall v \in N_G(A) \cap N_G(B) :\ \pi_A(v) = \pi_B(v)\}$.

Let $G = G_1 \cup \ldots \cup G_m$, where G_i is the (star) graph $\left[\{r_i\}\right]_G$ $(1 \leq i \leq m)$ and $V_R(G) = \{r_1, \ldots, r_m\}$. By iteratively using the above lemma, we construct the set $\mathbf{\Pi}_{G\to H}$ by finding *all the mutually compatible projections* of the star graphs G_i to H. A projection of G_i to H is completely determined by a star graph H_i of H, $H_i = \left[\{s_i\}\right]_H$ with the property $\lambda_G(r_i) \geq \lambda_H(s_i)$ and $\forall j = 1, k_i = d_G(r_i)(= d_H(s_i)) :\ \lambda_G(N^j_G(r_i)) \geq \lambda_H(N^j_H(s_i))$.

Let (r_i, s_i) and (r_j, s_j) describe individual projections of star graphs G_i and G_j; these projections are compatible if

$$N^k_G(r_i) = N^l_G(r_j) \Rightarrow N^k_H(s_i) = N^l_H(s_j).$$

The *matching graph* $\mathbb{M}_{G\to H}$ is a conceptual tool for managing the mutual compatibilities between individual projections of the star graphs of G. The vertices of this graph are pairs (r, s), where $r \in V_R(G)$ and $s \in V_R(H)$ represent the projection of the star graph $\left[\{r\}\right]_G$ into $\left[\{s\}\right]_H$.

The edges of $\mathbb{M}_{G\to H} = (V, E)$ show that their extremities represent a pair of compatible projections. More precisely:

Definition 5 *Let $SG = (G, \lambda_G)$ and $SH = (H, \lambda_H)$ be two simple conceptual graphs without isolated concept vertices defined on the same support S. The matching graph of SG and SH is the graph $\mathbb{M}_{G\to H} = (V, E)$ where:*

- *$V \subseteq V_R(G) \times V_R(H)$ is the set of all pairs (r, s) such that $r \in V_R(G)$, $s \in V_R(H)$, $\lambda_G(r) \geq \lambda_H(s)$ and for each $i \in \{1, \ldots, d_G(r)\}$ $\lambda_G(N^i_G(r)) \geq \lambda_H(N^i_H(s))$. For each $r \in V_R(G)$ let $V_r = \{(r, s) \in V(\mathbb{M}_{G\to H})\}$.*
- *E is the set of all 2-sets $\{(r, s), (r', s')\}$, where $r \neq r'$, $(r, s), (r', s') \in V$ and for each $i \in \{1, \ldots, d_G(r)\}$ and $j \in \{1, \ldots, d_G(r')\}$ such that $N^i_G(r) = N^j_G(r')$, the condition $N^i_H(s) = N^j_H(s')$ holds.*

The sets V_r are disjoint, their union is $V(\mathbb{M}_{G\to H})$, and no two vertices of the same V_r are adjacent in $\mathbb{M}_{G\to H}$ (i.e. each V_r is a stable set in this graph). This means that $(V_r)_{r\in V_R(G)}$ is a *colouring* of the matching graph $\mathbb{M}_{G\to H}$.

A q-clique in a graph F is a set of q pairwise adjacent vertices. The maximum positive number q for which there is a q-clique in F is denoted by $\omega(F)$. We denote by $\mathcal{C}^\omega_F$ the family of all $\omega(F)$-cliques of the graph F.

Clearly $\omega(\mathbb{M}_{G\to H}) \leq |V_R(G)|$, since no clique of $\mathbb{M}_{G\to H}$ can have more than a vertex from a stable set V_r. From the above discussion it follows that if $\pi \in \mathbf{\Pi}_{G\to H}$ is a projection from SG to SH, then $\{(r, \pi(r)) | r \in V_R(G)\}$ is a $|V_R(G)|$-clique in $\mathbb{M}_{G\to H}$. Conversely, if $\{(r_1, s_1), \ldots, (r_m, s_m)\}$ is a $|V_R(G)|$-clique in $\mathbb{M}_{G\to H}$, then taking for each $i = 1, m$, $\pi(r_i) \leftarrow s_i$ and $\pi(N^j_G(r_i)) \leftarrow \pi(N^j_H(s_i))$ (for each j-neighbour of r_i in G), we obtain a projection of SG to SH. Hence, the following theorem holds.

Theorem 1 *Let $SG = (G, \lambda_G)$ and $SH = (H, \lambda_H)$ be two SCGs without isolated concept vertices defined on the same support S and let $\mathbb{M}_{G\to H} = (V, E)$ be their matching graph. The following two conditions hold:*

1. *$SG \geq SH$, that is $\mathbf{\Pi}_{G\to H} \neq \emptyset$, if and only if $\omega(\mathbb{M}_{G\to H}) = |V_R(G)|$*
2. *If $\mathbf{\Pi}_{G\to H} \neq \emptyset$, then there is a 1:1 mapping between $\mathcal{C}^\omega_{\mathbb{M}_{G\to H}}$ and $\mathbf{\Pi}_{G\to H}$*

Consequently, the problem of deciding whether $SG \geq SH$ is reduced to the problem of testing whether the matching graph $\mathbb{M}_{G\to H}$ has a clique with exactly one vertex in each stable set V_r. If any V_r is empty (or if there is a vertex $(r, s) \in V_r$ with no neighbour in some $V_{r'}$ with $r \neq r'$), then there is no $|V_R(G)|$-clique in $\mathbb{M}_{G\to H}$ that contains (r, s).

The *reduced matching graph* (RMG) of SG and SH is the graph obtained from $\mathbb{M}_{G\to H}$ by eventually deleting a vertex with no neighbour in some nonempty stable set V_r that does not contain that vertex. The RMG is denoted as $\mathbb{RM}_{G\to H}$. It follows that $\omega(\mathbb{M}_{G\to H}) = |V_R(G)| \Leftrightarrow \omega(\mathbb{RM}_{G\to H}) = |V_R(G)|$ and the above theorem holds with $\mathbb{RM}_{G\to H}$ instead of $\mathbb{M}_{G\to H}$.

This reduction could be computationally worthwhile, since it removes from the very beginning some individual projections of the relation vertices of G, which do not belong to a projection of the whole graph G.

Reduced matching graph algorithm

Our algorithm devised to construct reduced matching graphs is as follows.

1. **for** each $r \in V_R(G)$ **do** {
   $V_r \leftarrow \emptyset$;
   for each $s \in V_R(H)$ **do**
   if $\lambda_G(r) \geq \lambda_H(s)$
   and $\lambda_G(N^i_G(r)) \geq \lambda_H(N^i_H(s))_{i=1,d_G(r)}$
   then $V_r \leftarrow V_r \cup \{(r, s)\}$; }
2. **for** each $r \in V_R(G)$ **do**
   for each $(r, s) \in V_r$ **do**

for each $r' \in V_R(G), r' \neq r$ **do** {
$d_{\mathbb{M}_{G\to H}}((r,s))_{r'} \leftarrow 0$; $Adj((r,s))_{r'} \leftarrow \emptyset$;
for each $(r',s') \in V_{r'}$ **do**
if $\forall i\ N_G^i(r) = N_G^i(s') \Rightarrow N_H^i(s) = N_H^i(s')$
then { $d_{\mathbb{M}_{G\to H}}((r,s))_{r'} + +$;
$Adj((r,s))_{r'} \leftarrow Adj((r,s))_{r'} \cup \{(r',s')\}$; }
}

3. Initialize as empty a stack S of vertices of $\mathbb{M}_{G\to H}$;
for each $r \in V_R(G)$ **do**
for each $(r,s) \in V_r$ **do**
if $\exists r' \in V_R(G), r' \neq r$ s.t. $d_{\mathbb{M}_{G\to H}}((r,s))_{r'} = 0$
then add (r,s) to S;
while S is nonempty **do** {
unstack (r,s), the top element of stack S;
for each $r' \in V_R(G), r' \neq r$ **do**
for each $(r',s') \in Adj((r,s))_{r'}$ **do** {
delete (r,s) from $Adj((r',s'))_r$;
$d_{\mathbb{M}_{G\to H}}((r',s'))_r \leftarrow d_{\mathbb{M}_{G\to H}}((r',s'))_r - 1$;
if $d_{\mathbb{M}_{G\to H}}((r',s'))_r = 0$
then add (r',s') to S; }
$V_r \leftarrow V_r - \{(r,s)\}$ }

Let m_G (m_H) be the number of relation vertices of graph G (H) and e_G (e_H) the number of edges of graph G (H). If the comparison of each label is accomplished in constant time, then the time complexity of algorithm step 1 is $\mathcal{O}(m_G \cdot m_H \cdot e_G)$. Moreover, as the number of vertices of graph $\mathbb{M}_{G\to H}$ is $\mathcal{O}(m_G \cdot m_H)$, the time complexity of algorithm step 2 is $\mathcal{O}(m_G^2 \cdot m_H^2)$. In algorithm step 3, the vertices (r,s) of graph $\mathbb{M}_{G\to H}$ are collected in the stack S in $\mathcal{O}(m_G^2 \cdot m_H)$. These vertices will be deleted, since they have no neighbours in some nonempty stable set $V_{r'}$ ($r' \neq r$). The vertex deletion process is controlled in order to capture all the new occurrences of a vertex that must be deleted in the current graph. In this case, each vertex occurrence is added to the stack S. The overall complexity of algorithm step 3 is dominated by step 2. Hence, graph $\mathbb{RMM}_{G\to H}$ can be constructed in $\mathcal{O}(m_G^2 \cdot m_H^2)$ time. The projection checking algorithm that follows from Theorem 1 can be now described as follows:

1. *Construct the reduced matching graph* $\mathbb{RMM}_{G\to H}$;
2. *Find the clique number* $\omega(\mathbb{RMM}_{G\to H})$*;*
3. **If** $\omega(\mathbb{RMM}_{G\to H}) < |V_R(G)|$
then return "$\mathbf{G} \not\geq \mathbf{H}$"
else return "$\mathbf{G} \geq \mathbf{H}$" and a clique $Q \in \mathcal{C}^{\omega}_{\mathbb{RMM}_{G\to H}}$

The key point in this algorithm is step 2, which corresponds to the problem of finding a maximum clique in a multipartite graph. This can be solved either by using a backtracking approach in case of 'small graphs' or by using distinct approximate techniques developed for the combinatorial optimisation of the max-clique problem. However, there are favourable combinatorial CG structures where this problem is polynomially solvable whenever the RMG belongs to a subset of max-clique problem instances. If that is the case, we can make use of our algorithm introduced below.

A new projection checking algorithm

Definition 6 *The SCG $SF = (F, \lambda_F)$ without isolated concept vertices is ordered in an acyclic way if its set of relation, namely $V_R(F) = \{r_1, \ldots, r_m\}$, is ordered in such a way that each r_i ($2 \leq i \leq m$) has at most one concept neighbour in the set of concept neighbours of $r_1, \ldots, r_{i-1}$, i.e. if*

$$(*) \quad |N_F(r_i) \cap N_F(\{r_1, \ldots, r_{i-1}\}| \leq 1 \quad \forall i = 2, m\,.$$

An simple conceptual graph ordered in an acyclic way SF is called a *simple conceptual forest*, since condition $(*)$ implies that each connected component of the bipartite graph $F = (V_C(F), V_R(F); E(F))$ is a tree (i.e. this condition implies that F is connected and has no cycle of length $\geq$ 2). Conversely, if the bipartite graph F has no cycle of length $\geq$ 2, then any breadth-first traversal of F starting from an arbitrary relation vertex r_1 provides an ordering of $V_R(F)$ that satisfies $(*)$ in $\mathcal{O}(|V_C(F) \cup V_R(F)|)$ time.

In the following, we suppose that any conceptual forest $SF = (F, \lambda_F)$ is provided with an acyclic ordering, $V_R(F) = \{r_1, \ldots, r_m\}$, and with an array $parent[r_i]_{i=1,m}$ such that $\forall i \in \{1, \ldots, m\}$, where $parent[r_i] \in \{r_1, \ldots, r_i\}$ has the following interpretation:

1. $parent[r_i] = r_i \Leftrightarrow N_F(r_i) \cap N_F(\{r_1, \ldots, r_{i-1}\}) = \emptyset$;
2. $parent[r_i] = r_j \ (j < i) \Leftrightarrow N_F(r_i) \cap N_F(\{r_1, \ldots, r_{i-1}\}) = \{c\}; c \in N_F(r_j)$.

Let $SF = (F, \lambda_F)$ be a conceptual forest, $SH = (H, \lambda_H)$ be a SCG defined on the same support S, and let $\mathbb{RMM}_{F\to H} = (V, E)$ be their reduced matching graph. As described in the previous section, if the acyclic ordering of F is $V_R(F) = \{r_1, \ldots, r_m\}$, then the construction of $\mathbb{RMM}_{F\to H} = (V, E)$ provides the sets V_{r_i} $(i = 1, m)$, such that $V = \cup_{i=1,m} V_{r_i}$. Moreover, if $i \neq j$ and $V_{r_i}, V_{r_j} \neq \emptyset$, then for each vertex $(r_i, s) \in V_{r_i}$ there is a nonempty adjacency list $Adj((r_i, s))_{r_j}$ containing the $\mathbb{RMM}_{F\to H}$-neighbours in the set V_{r_j} of vertex (r_i, s). The fact entailed by the acyclic structure of the conceptual forest is that graph $\mathbb{RMM}_{F\to H}$ and its m-colouring $(V_{r_i})_{i=1,m}$ encode all the information about $\mathbf{\Pi}_{F\to H}$. More precisely, the following theorem holds.

Theorem 2 *Let $SF = (F, \lambda_F)$ be a conceptual forest, $SH = (H, \lambda_H)$ a SCG defined on the same support S, and let $\mathbb{RMM}_{F\to H}$ be their reduced matching graph. Then, a projection from SF to SH exists if and only if, for each $r \in V_R(F)$, the reduced matching graph $\mathbb{RMM}_{F\to H}$ has $V_r \neq \emptyset$.*

Proof. According to theorem 1, it is sufficiently to prove that if $V_{r_i} \neq \emptyset$ for each $i \in \{1, \ldots, m\}$, then $\omega(\mathbb{RMM}_{F\to H}) = m$. This follows from the fact that each i-clique Q of $\mathbb{RMM}_{F\to H}$, $(1 \leq i \leq m-1)$, which intersect the first i colour sets V_{r_i}, can be extended to an $(i+1)$-clique of $\mathbb{RMM}_{F\to H}$ by adding a new vertex from $V_{r_{i+1}}$.

Indeed, if $parent[r_{i+1}] = r_{i+1}$, then each vertex (r_{i+1}, s) is adjacent in $\mathbb{RMM}_{F\to H}$ to all vertices in $\cup_{j=1,i-1} V_{r_j}$ and $Q \cup \{(r_{i+1}, s)\}$ is an $(i+1)$-clique. If $parent[r_{i+1}] = r_j$ $(j < i)$, then $Q \cap V_{r_j} \neq \emptyset$ given our choice of Q.

Let $\{(r_j, s')\} = Q \cap V_{r_j}$, (r_{i+1}, s) be any vertex from $Adj((r_j, s'))_{r_{i+1}}$ and let (r, s'') be an arbitrary vertex of Q. If $N_F(r) \cap N_F(r_{i+1}) \neq \emptyset$, then $N_F(r) \cap N_F(r_{i+1}) = N_F(r_j) \cap N_F(r_{i+1})$. Since (r_j, s') is adjacent to (r_{i+1}, s), it follows that (r, s'') is adjacent to (r_{i+1}, s). If $N_F(r) \cap N_F(r_{i+1}) = \emptyset$ then (r, s'') is adjacent to (r_{i+1}, s).

We have obtained that (r_{i+1}, s) is adjacent in $\mathbb{RM}_{F \to H}$ to all vertices in Q, i.e. $Q \cup \{(r_{i+1}, s)\}$ is an $(i+1)$-clique, and the theorem is thus proved.

Furthermore, it follows from it that the recursive algorithm introduced below can be used to generate the set of all maximum cliques of $\mathcal{C}^{\omega}_{\mathbb{RM}_{F \to H}}$, i.e. $\mathbf{\Pi}_{F \to H}$.

Generate($level$**)**

Input: $level$ the current size of the clique constructed. First call is for $level = 0$.
Global variables: Array Q, the current clique constructed. Reduced matching graph $\mathbb{RM}_{F \to H}$: sets $V_{r_i} \neq \emptyset$ and adjacency lists $Adj((r_i, s))_{r_j} \neq \emptyset$ for each $(r_i, s) \in V_{r_i}$ and $i \neq j \in \{1, \ldots, m\}$. Array $parent$ associated to an acyclic ordering: $\{r_1, \ldots, r_m\}$ of $V_R(F)$.
Output: $\mathcal{C}^{\omega}_{\mathbb{RM}_{F \to H}}$, the set of all m-cliques of $\mathbb{RM}_{F \to H}$, initially empty.

```
{ level++
  if level > m then C^ω_{RM_{F→H}} ← C^ω_{RM_{F→H}} ∪ Q
    else if parent[r_level] = r_level
      then for each (r_level, s) ∈ V_{r_level} do {
      Q[level] ← (r_level, s);
        Generate(level) }
      else {
        let (r_j, s') ← Q[parent[r_level]]
        for each (r_level, s) ∈ Adj((r_j, s'))_{r_level} do {
          Q[level] ← (r_level, s)
          Generate(level) } } }
```

In the above algorithm the time complexity per clique generated is $O(m)$.

The algorithmic implications of Theorem 2 w.r.t. the general projection checking problem '$SG \geq SH$?' (where SG is an arbitrary SCG) are facilitated by the following two RMG properties, which describe an effective application of the *Combination Lemma*.

Let $SG \geq SH$, $A \subseteq V_R(G)$ and let $\mathbb{RM}_{A \to H}$ be the subgraph of $\mathbb{RM}_{G \to H}$ induced by the set of vertices $\{(r, s) \in V(\mathbb{RM}_{G \to H}) | r \in A\}$. In our representation of the reduced matching graph (as described in Section 2), graph $\mathbb{RM}_{A \to H}$ can be obtained by considering only the adjacency lists of these vertices with respect to A. Note that $\mathbb{RM}_{A \to H}$ is a subgraph of the matching graph $\mathbb{RM}_{[A]_G \to H}$. Statement (ii) in Theorem 1 implies the following lemma.

Lemma 2 *The set of all projections of conceptual subgraph $[A]_G$ to SH, which can be extended to a projection of the entire simple conceptual graph SG into SH, is contained in $\mathcal{C}^{\omega}_{\mathbb{RM}_{A \to H}}$.*

Moreover, let $V_R(G) = A \cup B$ ($A, B \neq \emptyset, A \cap B = \emptyset$), and Q be an $|A|$-clique of $\mathbb{RM}_{A \to H}$. We denote by $\mathbb{RM}_{G \to H} | Q$ the subgraph of $\mathbb{RM}_{B \to H}$ induced by the set of vertices $\{(r, s) \in V(\mathbb{RM}_{G \to H}) | r \in B, Q \subseteq N_{\mathbb{RM}_{G \to H}}(r)\}$.

Lemma 3 *The set of projections of the conceptual subgraph $[B]_G$ to SH, which extend the projection of $[A]_G$ represented by Q to a projection of the entire SG to SH, is contained in $\mathcal{C}^{\omega}_{\mathbb{RM}_{B \to H} | Q}$.*

We repeatedly use these lemmas to decompose an arbitrary SCG $SG = (G, \lambda_G)$ into conceptual forest subgraphs. This is due to the algorithmic reductions described in Theorem 2 and the algorithm previously designed.

Definition 7 *Let $SG = (G, \lambda_G)$ be an arbitrary SCG. An acyclic decomposition of SG is a union $G = F_1 \cup \ldots \cup F_k$ such that $\big(V_R(F_1), \ldots, V_R(F_k)\big)$ is a partition into k classes of $V_R(G)$, where $1 \leq k \leq |V_R(G)|$ and F_i are conceptual forest subgraphs.*

Since the number k of conceptual subforests in which a conceptual graph can be decomposed is relatively small in practical applications, here we are not concerned with the minimisation of this parameter (which we suspect to be a NP-hard problem). Instead, we use the following greedy method to obtain an acyclic decomposition.

Input: SCG $SG = (G, \lambda_G)$
Output: Acyclic decomposition $G = F_1 \cup \ldots \cup F_k$, where $F_i = [\{r^i_1, \ldots, r^i_{m_i}\}]_G$ $(i = 1, k)$ is a conceptual subforest of G.

```
{ r^1_1 ← r, an arbitrary relation vertex of V_R(G)
  m_1 ← 1; k ← 1; V_R(G) ← V_R(G) − {r}
  for each r ∈ V_R(G) do {
    placed ← false; j ← 1
    while (j ≤ k & not placed ) do {
      if N_G(r) ∩ N_G({r^j_1, ..., r^j_{m_j}}) = ∅
          then {
            m_j++; r^j_{m_j} ← r; parent[r^j_{m_j}] ← r^j_{m_j};
            placed ← true; }
      else if |N_G(r) ∩ N_G({r^j_1, ..., r^j_{m_j}})| = 1
          then {
            m_j++; r^j_{m_j} ← r; placed ← true;
            find r^j_h such that N_G(r) ∩ N_G(r^j_h) ≠ ∅;
            parent[r^j_{m_j}] ← r^j_h; }
      else j++ }
    if not placed then {
      k++; m_k ← 1; r^k_1 ← r;
      parent[r^k_1] ← r } } }
```

The above procedure can be used as a preprocessing 'step 0' in the projection checking algorithm outlined at the end of the section describing the reduced matching graph construction algorithm. Step 1 remains unchanged, with the reduced matching graph $\mathbb{RM}_{G \to H}$ encoding (by way of Lemma 2) the entire set of reduced matching graphs $\mathbb{RM}_{F_i \to H}$ $(i = 1, k)$. Steps 2 and 3 are implemented using a backtracking scheme working with 'big' steps corresponding to each conceptual forest constructed in the preprocessing step. More precisely, we maintain a current partial solution $Q_1, \ldots, Q_h$ describing the current projection of $F_1 \cup \ldots \cup F_h$ to H. From

Lemma 3 the compatible projections of the remaining graph $F_{h+1} \cup \ldots \cup F_h$ must be searched in graph $\mathbb{RM}^h$, which only contains the $\mathbb{RM}^{h-1}$ vertices that are adjacent to all vertices in $Q_1, \ldots, Q_h$. Initially, that holds for $h = 0$, $\mathbb{RM}^0$ simply being $\mathbb{RM}_{G \to H}$.

In the following description of the algorithm, we use the term 'for each clique in $\mathcal{C}^{\omega}_{\mathbb{RM}_{F \to H}}$ '. It denotes the call of **Generate** in order to determine the maximum cliques of $\mathbb{RM}_{F \to H}$ when F is a conceptual forest.

Input: SCG's $SG = (G, \lambda_G)$ and $SH = (H, \lambda_H)$
Output: Either '$\mathbf{G} \not\geq \mathbf{H}'$ or "$\mathbf{G} \geq \mathbf{H}$ "
and clique Q encoding the projection constructed.

1. *Preprocessing step:*
Find an acyclic decomposition $G = F_1 \cup \ldots \cup F_k$**;**
Construct the reduced matching graph $\mathbb{RM}_{G \to H}$**;**
$\mathbb{RM}^0 \leftarrow \mathbb{RM}_{G \to H}$; $found \leftarrow false$;
2. $\mathbf{Find\_Projection(0)}$**;**
if not $found$ **then** '$\mathbf{G} \not\geq \mathbf{H}'$ **else**
'$\mathbf{G} \geq \mathbf{H}' : Q = Q_1 \cup \ldots \cup Q_k$.

$\mathbf{Find\_Projection(h)}$**;**
if *not* $found$ **then**
if $h = k$ **then** $found \leftarrow true$
else if ($\forall r \in \cup_{i=h+1,k} V_R(F_i)$, $\mathbb{RM}^h$ has $V_r \neq \emptyset$)
then for each clique $Q \in \mathcal{C}^{\omega}_{\mathbb{RM}^h_{F_{h+1} \to H}}$ **do** {
construct $\mathbb{RM}^{h+1}$**;**
$Q_{h+1} \leftarrow Q$**;**
$\mathbf{Find\_Projection(h+1)}$ }

Conclusions

In this paper we introduced a further step along the [6] results, discussing an approach that uses graph-theoretical tools to speed-up reasoning with CGs. Our approach uses effective look-ahead graph theoretical procedures based on a conceptual forest decomposition of the graph to be projected. Until now, the various approaches to conceptual graph projection proposed in the literature did not fully use the rich combinatorial structure of CGs. By exploiting this structure, our algorithm is able to eliminate unneeded comparisons, providing improved complexity results over existing techniques. Further efficiencies are obtained by studying a graph's acyclic tree decomposition in the context of its corresponding reduced matching graph. These techniques, in our opinion, will be useful to any CG based system making use of projection.

As previously mentioned, conceptual graph reasoning is polynomially equivalent to constraint satisfaction in combinatorial optimization. Further investigation of tree decomposition mechanisms from a constraint satisfaction point of view, and applying them to our work, promises to be a fruitful future area of research.

References

[1] Baget, J.-F., and Mugnier, M.-L. 2002. Extensions of Simple Conceptual Graphs: the Complexity of Rules and Constraints. *Jour. of Artif. Intell. Res.* 16:425–465.

[2] Chein, M.; Mugnier, M.-L.; and Simonet, G. 1998. Nested graphs: A graph-based knowledge representation model with FOL semantics. In *Proc. of the 6th Int'l Conf. on the Principles of Knowl. Repres. and Reasoning (KR'98)*, 524–535. Morgan Kaufmann.

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

[4] Chein, M., and Mugnier, M.-L. 1995. Conceptual Graphs are also Graphs. Research report, LIRMM.

[5] Corbett, D. R. 2002. Reasoning with ontologies by using knowledge conjunction in conceptual graphs. *Proc. of Int'l Conf. on Ontologies, DBs and Applications of Semantics*.

[6] Croitoru, M., and Compatangelo, E. 2004. A combinatorial approach to conceptual graph projection checking. In *Proc. of the 24th Int'l Conf. of the Brit. Comp. Soc. Spec. Group on Artif. Intell. (AI'2004)*, 130–143. Springer.

[7] Gottlob, G.; Leone, N.; and Scarcello, F. 2001. Hypertree Decompositions: A Survey. In *Proc. of the 26th Int'l Symp. on Mathematical Foundations of Computer Science*, 37–57. Springer.

[8] Mineau, G.; Missaoui, R.; and Godin, R. 2000. Conceptual Modeling for Data and Knowledge Management. *Data & Knowl. Eng.* 33(2):137–168.

[9] Mugnier, M.-L. 1995. On generalization /specialization for conceptual graphs. *Jour. of Exp. and Theor. Comp. Sci.* 7:325–344.

[10] Salvat, E. 1997. *Raisonner avec des opérations de graphes: graphes conceptuels et règles d'inférence*. Ph.D. Dissertation, LIRMM, Université Montpellier II.

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

[12] Sowa, J. 2000. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing Co.

[13] Willems, M. 1995. Projection and unification for conceptual graphs. In *Proc. of the 3rd Int'l Conf. on Conceptual Structures (ICCL'95)*, 278–292.