Redundancy and Subsumption in High-Level Replacement Systems[*]

H.-J. Kreowski[1], G. Valiente[1,2]

[1] Mathematics and Computer Science Department, University of Bremen, Germany
[2] Department of Software, Technical University of Catalonia, Catalonia, Spain

Abstract. System verification in the broadest sense deals with those semantic properties that can be decided or deduced by analyzing a syntactical description of the system. Hence, one may consider the notions of redundancy and subsumption in this context as they are known from the area of rule-based systems. A rule is redundant if it can be removed without affecting the semantics of the system; it is subsumed by another rule if each application of the former one can be replaced by an application of the latter one with the same effect. In this paper, redundancy and subsumption are carried over from rule-based systems to high-level replacement systems, which in turn generalize graph and hypergraph grammars. The main results presented in this paper are a characterization of subsumption and a sufficient condition for redundancy, which involves composite productions.

1 Introduction

High-level replacement systems [6] generalize the algebraic approach to graph transformation, both the double-pushout approach [3] and the single-pushout approach [5], to other classes of replacement systems. They provide a common categorical framework for different classes of replacement systems, such as grammars on graphs, relational structures, and algebraic specifications, based on categories and pushouts.

This paper deals with aspects of verification of both double-pushout (DPO) and single-pushout (SPO) high-level replacement systems. System verification in the broadest sense is concerned with those semantic properties that can be decided or deduced by analyzing a syntactical description of the system. The properties studied in this paper are redundancy and subsumption, as they are known from the area of rule-based systems (see, for instance, [2], [8], [9]).

Consider a high-level replacement system, that is, a set of productions, an initial object, and a class of terminal objects. A production is subsumed by another if any application of the former is mimicked by the latter. As a first result,

[*] Partially supported by the EC TMR Network GETGRATS (General Theory of Graph Transformation Systems) and by the ESPRIT Basic Research Working Group APPLIGRAPH (Applications of Graph Transformation) through the University of Bremen, and by the Spanish DGES project PB96-0191-C02.

we present a sufficient condition for subsumption. If a production p' is covered by a production p, that is, p' is directly derived by p, then p' is also subsumed by p. This is interesting because covering is easier to check than subsumption, since it is defined by a single direct derivation. It turns out that covering is not only sufficient, but also necessary for subsumption in the case of single-pushout high-level replacement systems, whereas this is not true in the double-pushout case. Moreover, we consider subsumption of a production by composite productions. Even in this case, one can show that the subsumed production is redundant, that is, the semantics of the given system does not change if the production is removed. Altogether, one obtains a procedure that removes some redundancy from a given system: Enumerate composite productions, check them for covering, and remove every covered production.

2 High-Level Replacement Systems

In this section, we recall the basic notions and notations of high-level replacement systems the rewriting of which is based on both double-pushout (DPO) and single-pushout (SPO) constructions.

Starting with the DPO case, let $\mathcal{C}$ be a category, whose objects will be regarded as high-level structures and whose morphisms will be regarded as structure-preserving mappings between these objects. The morphisms in a distinguished class $\mathcal{M}$ will be used in productions, while general morphisms in $\mathcal{C}$ will be used to define application of productions to high-level structures.

Definition 1 (DPO HLR system). *Let $\mathcal{C}$ be a category with a distinguished class $\mathcal{M}$ of morphisms.*

1. *A* DPO production *$p = (L \leftarrow K \rightarrow R)$ in $\mathcal{C}$ consists of a pair of objects (L, R), called* left-hand side object *and* right-hand side object *respectively, an object K, called* interface object*, and two morphisms $K \rightarrow L$ and $K \rightarrow R$ belonging to $\mathcal{M}$.*
2. *An object G can be* directly derived *into an object H using a DPO production $p = (L \leftarrow K \rightarrow R)$, denoted by $G \Rightarrow H$ via p if there are pushout squares*

$$\begin{tikzcd} L \arrow[d] & K \arrow[l] \arrow[r] \arrow[d] & R \arrow[d] \\ G & D \arrow[l] \arrow[r] & H \end{tikzcd}$$

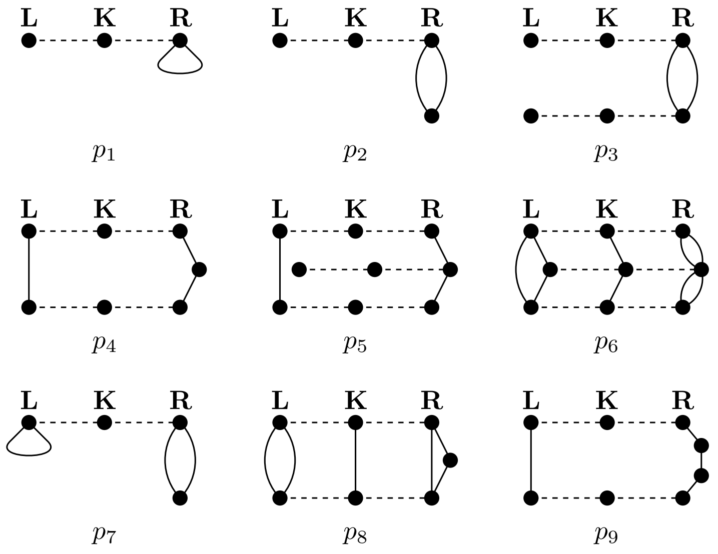
in $\mathcal{C}$.

3. *Let $\boldsymbol{P}$ be a set of DPO productions. A* derivation *$G \Rightarrow^* H$ from G to H in $\boldsymbol{P}$ is a sequence of $n \geqslant 0$ direct derivations $G = G_0 \Rightarrow G_1 \Rightarrow \cdots \Rightarrow G_n = H$ via $(p_1, \ldots, p_n)$ provided that $p_1, \ldots, p_n \in \boldsymbol{P}$.*
4. *A* high-level replacement system *$H = (S, \boldsymbol{P}, \boldsymbol{T})$ in $\mathcal{C}$ consists of a* start object *S in $\mathcal{C}$, a set $\boldsymbol{P}$ of DPO productions, and a class $\boldsymbol{T}$ of* terminal objects *in $\mathcal{C}$.*

5. *The* language $L(H)$ *of a high-level replacement system* $H = (S, \boldsymbol{P}, \boldsymbol{T})$*, also denoted by* $L(S, \boldsymbol{P}, \boldsymbol{T})$*, is given by the set of all terminal objects in* $\mathcal{C}$ *derivable from* S *by* $\boldsymbol{P}$*, that is,* $L(H) = \{G \in \boldsymbol{T} \mid S \Rightarrow^* G\}$*.* $\square$

The following example presents high-level replacement system (which is actually a graph grammar) that allows the generation and recognition of all Eulerian graphs, based on [10, Sect. 3.2]. Recall that a graph is Eulerian if it has an Euler circuit, that is, a circuit that contains every edge of the graph exactly once. Eulerian graphs are characterized by being connected and having only nodes of even degree [1].

Example 2. Let $\mathcal{C}$ be the category of undirected graphs and $\mathcal{M}$ be the class of all injective graph morphisms. Consider the following double-pushout high-level replacement system $(S, \mathbf{P}, \mathbf{T})$ for generating and recognizing all Eulerian graphs, where S is a graph with a single node and no arc, $\mathbf{P} = \{p_1, \ldots, p_9\}$, and $\mathbf{T}$ is the class of all graphs.



The dotted lines indicate the morphisms from the interface object to the left-hand side object and to the right-hand side object, respectively. $\square$

The SPO case differs from the DPO case in two aspects: A production consists of a single morphism and a direct derivation of a single pushout. In typical examples like graphs, the morphism of a production may be a partial mapping (where nodes and edges outside the domain of definition specify the items to be removed) while the occurrence of the left-hand side in the host graph should be a total mapping. To formalize such situations, a category and a subcategory are assumed.

Definition 3 (SPO HLR system). *Let* $\mathcal{C}$ *be a category and let* $\mathcal{O}$ *be a subcategory of* $\mathcal{C}$ *such that* $\mathcal{O}$ *coincides with* $\mathcal{C}$ *on objects.*

1. *An* SPO production $p = (L \to R)$ *in* $\mathcal{C}$ *consists of a pair of objects* (L, R), *called* left-hand side object *and* right-hand side object *respectively, and a morphism* $L \to R$ *in* $\mathcal{C}$.
2. *An object* G *can be* directly derived *into an object* H *using an SPO production* $p = (L \to R)$, *denoted by* $G \Rightarrow H$ *via* p *if there is a pushout square*

$$\begin{tikzcd} L \arrow[r] \arrow[d] & R \arrow[d] \\ G \arrow[r] & H \end{tikzcd}$$

   in $\mathcal{C}$ *such that* $L \to G$ *is in* $\mathcal{O}$.
3. *Let* $\boldsymbol{P}$ *be a set of SPO productions. A* derivation $G \Rightarrow^* H$ *from* G *to* H *in* $\boldsymbol{P}$ *is a sequence of* $n \geqslant 0$ *direct derivations* $G = G_0 \Rightarrow G_1 \Rightarrow \cdots \Rightarrow G_n = H$ *via* $(p_1, \ldots, p_n)$ *provided that* $p_1, \ldots, p_n \in \boldsymbol{P}$.
4. *A* high-level replacement system $H = (S, \boldsymbol{P}, \boldsymbol{T})$ *in* $\mathcal{C}$ *consists of a* start object S *in* $\mathcal{C}$, *a set* $\boldsymbol{P}$ *of SPO productions, and a class* $\boldsymbol{T}$ *of* terminal objects *in* $\mathcal{C}$.
5. *The* language $L(H)$ *of a high-level replacement system* $H = (S, \boldsymbol{P}, \boldsymbol{T})$, *also denoted by* $L(S, \boldsymbol{P}, \boldsymbol{T})$, *is given by the set of all terminal objects in* $\mathcal{C}$ *derivable from* S *by* $\boldsymbol{P}$, *that is,* $L(H) = \{G \in \boldsymbol{T} \mid S \Rightarrow^* G\}$. □

Example 4. If one ignores the interface graphs in the productions of Example 2 and interprets the dotted lines as inclusions of the left-hand side nodes into the right-hand side nodes, one obtains SPO productions in the category of graphs with partial graph morphisms. Choosing the category of graphs with total graph morphisms as subcategory, the application of an SPO production has the same effect as the application of the corresponding DPO production, that is, the edges of the left-hand side are removed and the right-hand side is added by merging the related nodes. □

3 Redundancy and subsumption

Verification of high-level replacement systems is concerned with formal properties of the systems. Some of the formal properties to be verified arise from the rule-based paradigm itself; cf. [10]. In particular, redundancy and subsumption are generalized in this section from rule-based systems to high-level replacement systems.

A high-level replacement system is redundant if it contains a production that can be removed without affecting the semantics of the system. In particular, such productions may be subsumed by other productions. A production subsumes (or is more general than) another production if the subsuming production can be applied and it yields the same result whenever the subsumed production can be applied.

Definition 5 (Redundancy and Subsumption). *Let* $\boldsymbol{P}$ *be a set of productions (either of the DPO or SPO type).*

1. *A production $q \in \boldsymbol{P}$ is* redundant *if there is a derivation $G \Rightarrow^* H$ in $\boldsymbol{P} - \{q\}$ whenever there is a derivation $G \Rightarrow^* H$ in $\boldsymbol{P}$.*
2. *A production $p \in \boldsymbol{P}$* subsumes *a production $q \in \boldsymbol{P}$, denoted by $p \leqslant q$, if there is a direct derivation $G \Rightarrow H$ via p whenever there is a direct derivation $G \Rightarrow H$ via q.* □

Obviously, a production q is redundant if it is subsumed by a production p. Moreover, a high-level replacement system $H = (S, \mathbf{P}, \mathbf{T})$ for some start object S and class of terminal objects $\mathbf{T}$ generates the same language as $H - q = (S, \mathbf{P} - \{q\}, \mathbf{T})$ if q is redundant or, in particular, if q is subsumed by some $p \in \mathbf{P} - \{q\}$. The other way round, a production $q \in \mathbf{P}$ is redundant if $L(S, \mathbf{P}, \mathbf{T}) = L(S, \mathbf{P} - \{q\}, \mathbf{T})$ for any start object S and any class $\mathbf{T}$ of terminal objects.

Example 6. Productions p_6 to p_9 in Example 2 are redundant. Production p_6 is subsumed by production p_5, and productions p_7 and p_8 are subsumed by production p_4. Production p_9 is discussed in Example 16. □

Redundancy and subsumption may be undesirable for several reasons. First, the presence of redundant and subsumed productions may degrade execution efficiency. Second, and most important, it may make system validation more difficult.

4 A characterization of subsumption

Subsumption can be regarded as a *local* form of redundancy, since it only involves two productions: the subsuming production and the subsumed production. Nevertheless, subsumption is difficult to check because the definiton involves arbitrary objects to be derived by the involved productions. To avoid this obstacle, we introduce the notion of a covering, which relates two productions by means of a single direct derivation and is, therefore, much easier to check. Using the sequential composition of pushouts, covering can be shown to imply subsumption. Moreover, it turns out that covering and subsuption are equivalent notions in the SPO case, but not in the DPO case.

Definition 7 (Covering in DPO HLR systems). *Production $p = (L \leftarrow K \rightarrow R)$* covers *production $p' = (L' \leftarrow K' \rightarrow R')$ if there exist morphisms $L \rightarrow L'$, $K \rightarrow K'$ and $R \rightarrow R'$ such that $L \rightarrow L' \leftarrow K'$ is a pushout of $L \leftarrow K \rightarrow K'$ and $K' \rightarrow R' \leftarrow R$ is a pushout of $K' \leftarrow K \rightarrow R$.*

$$\begin{tikzcd}
L \arrow[d, dotted] \arrow[dr, phantom, "(PO)"] & K \arrow[l] \arrow[r] \arrow[d, dotted] \arrow[dr, phantom, "(PO)"] & R \arrow[d, dotted] \\
L' & K' \arrow[l] \arrow[r] & R'
\end{tikzcd}$$

□

Theorem 8. *Let $p = (L \leftarrow K \rightarrow R)$ and $p' = (L' \leftarrow K' \rightarrow R')$ be two productions. Then $p \leqslant p'$ if p covers p'.* □

Proof. Consider a direct derivation $G \Rightarrow H$ via p' of the form

$$\begin{tikzcd}
L' \arrow[d] & K' \arrow[l] \arrow[r] \arrow[d] & R' \arrow[d] \\
G & D \arrow[l] \arrow[r] & H
\end{tikzcd}$$

By hypothesis, there exist pushout squares (1) and (2), and by (vertical) composition of pushout squares, there also exist pushout squares (3) and (4), that is, there exists a direct derivation $G \Rightarrow H$ via p.

$$\begin{tikzcd}
L \arrow[d] \arrow[dr, phantom, "(1)"] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(2)"] & R \arrow[d] \\
L' & K' \arrow[l] \arrow[r] & R'
\end{tikzcd}
\qquad
\begin{tikzcd}
L \arrow[d] \arrow[dr, phantom, "(3)"] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(4)"] & R \arrow[d] \\
G & D \arrow[l] \arrow[r] & H
\end{tikzcd}$$

□

Observation 9. *There are DPO productions p and p' such that $p \leqslant p'$, but p does not cover p'.* □

Proof. Consider the following counter-example in the category **Gra** of graphs and graph morphisms. Let $p = (\bullet \leftarrow \bullet \rightarrow \bullet)$ and $p' = (\bullet \leftarrow \emptyset \rightarrow \bullet)$ be two productions given by identities and by empty morphisms, respectively.

There exists a derivation $G \Rightarrow H$ via p' if and only if the left-hand side node is mapped to an isolated node of G. Moreover, in this case, we have $H = G$ because the isolated node is removed and added again. Using the same occurrence, we obtain a derivation $G \Rightarrow G$ via p. In other words, $p \leqslant p'$. However, p does not cover p' because there is no morphism $\bullet \rightarrow \emptyset$ in **Gra**, and, therefore, no double pushout of the form

$$\begin{tikzcd}
\bullet \arrow[d] & \bullet \arrow[l] \arrow[r] \arrow[d, dotted] & \bullet \arrow[d] \\
\bullet & \emptyset \arrow[l] \arrow[r] & \bullet
\end{tikzcd}$$

□

In the case of single-pushout high-level replacement systems, however, subsumption is characterized by covering.

Definition 10 (Covering in SPO HLR systems). *Production $p = (L \rightarrow R)$* covers *production $p' = (L' \rightarrow R')$ if there exist morphisms $L \rightarrow L'$ in $\mathcal{O}$ and $R \rightarrow R'$ such that $L' \rightarrow R' \leftarrow R$ is a pushout of $L' \leftarrow L \rightarrow R$.*

$$\begin{tikzcd}
L \arrow[r] \arrow[d, dotted] \arrow[dr, phantom, "(PO)"] & R \arrow[d, dotted] \\
L' \arrow[r] & R'
\end{tikzcd}$$

□

Theorem 11. *Let $p = (L \to R)$ and $p' = (L' \to R')$ be two productions. Then $p \leqslant p'$ if and only if p covers p'.* □

Proof. (If part) Consider a direct derivation $G \Rightarrow H$ via p' of the form

$$\begin{tikzcd}
L' \arrow[r] \arrow[d] & R' \arrow[d] \\
G \arrow[r] & H
\end{tikzcd}$$

By hypothesis, there exists pushout square (1), and by (vertical) composition of pushout squares, there also exists pushout square (2), since the composition of morphism $L \to L'$ with morphism $L' \to G$ is also in $\mathcal{O}$. That is, there exists a direct derivation $G \Rightarrow H$ via p.

$$\begin{tikzcd}
L \arrow[r] \arrow[d] \arrow[dr, phantom, "(1)"] & R \arrow[d] \\
L' \arrow[r] & R'
\end{tikzcd}
\qquad
\begin{tikzcd}
L \arrow[r] \arrow[d] \arrow[dr, phantom, "(2)"] & R \arrow[d] \\
G \arrow[r] & H
\end{tikzcd}$$

(Only-if part) Since the hypothesis holds for any direct derivation through production p', in particular it holds for the direct derivation $L' \Rightarrow R'$ via p', that is, for the direct derivation given by pushout square (1) where the vertical morphisms are the identities. Then, there also exists a direct derivation $L' \Rightarrow R'$ via p, that is, there exist morphisms $L \to L'$ and $R \to R'$ such that square (2) is a pushout square. Then p covers p'.

$$\begin{tikzcd}
L' \arrow[r] \arrow[d] \arrow[dr, phantom, "(1)"] & R' \arrow[d] \\
L' \arrow[r] & R'
\end{tikzcd}
\qquad
\begin{tikzcd}
L \arrow[r] \arrow[d] \arrow[dr, phantom, "(2)"] & R \arrow[d] \\
L' \arrow[r] & R'
\end{tikzcd}$$

□

Contrary to the case of (linear) rule-based systems, however, mutual subsumption does not mean isomorphic productions.

Observation 12. *$p \leqslant p'$ and $p' \leqslant p$ does not imply $p = p'$.* □

Proof. Consider the following counter-example in the category **Set** of sets and functions. Let $A = \{a\}$ and $B = \{b, c\}$ be two sets, and let $p = (A, A, A)$ and

$p' = (B, B, B)$ be two double-pushout productions given by identities.

$$\begin{tikzcd}
 & \mathbf{L} & \mathbf{K} & \mathbf{R} & \\
p = (& a \arrow[d, dotted] \arrow[dr, phantom, "(1)"] & a \arrow[l, dotted] \arrow[r, dotted] \arrow[d, dotted] \arrow[dr, phantom, "(2)"] & a \arrow[d, dotted] &) \\
p' = (& b \arrow[dd, dotted, bend right] & b \arrow[l, dotted] \arrow[r, dotted] \arrow[dd, dotted, bend right] & b \arrow[dd, dotted, bend right] & \\
 & c \arrow[d, dotted] \arrow[dr, phantom, "(3)"] & c \arrow[l, dotted] \arrow[r, dotted] \arrow[d, dotted] \arrow[dr, phantom, "(4)"] & c \arrow[d, dotted] &) \\
p = (& a & a \arrow[l, dotted] \arrow[r, dotted] & a &)
\end{tikzcd}$$

There exist morphisms $L \to L'$, $K \to K'$ and $R \to R'$ given by $a \mapsto b$ such that (1) and (2) become pushout squares, and there exist morphisms $L' \to L$, $K' \to K$ and $R' \to R$ given by $b \mapsto a, c \mapsto a$ such that (3) and (4) also become pushout squares. However, p and p' are not isomorphic productions.

A similar counter-example applies to single-pushout productions. □

5 A sufficient condition for redundancy

While subsumption between productions of a high-level replacement system can be regarded as a local form of redundancy, one obtains more global forms of redundancy by the combination and composition of several production to subsume other productions.

Actually, the most general notion of composition is given by the construction of concurrent productions, which consists of the composition of two productions over a dependency relation D between the right-hand side object of the first production and the left-hand side object of the second production. The resulting composite production is called D-concurrent production. We recall the notion for double-pushout high-level replacement systems as given in [4]. To guarantee that all necessary constructions exist, we assume so-called HLR2 categories, which are defined in the Appendix (according to [6]).

Definition 13 (Concurrent DPO production).

1. *Let $p = (L \leftarrow K \to R)$ and $p' = (L' \leftarrow K' \to R')$ be two productions and let D be an object together with two morphisms $D \to R$ and $D \to L'$. The pair $(D \to R, D \to L')$, or short D, is called a* dependency relation *for (p, p') if the pushout object H^* of $D \to R$ and $D \to L'$ exists and if there are unique pushout complements of $K \to R \to H^*$ and $K' \to L' \to H^*$ up to isomorphism.*
2. *Given a dependency relation $(D \to R, D \to L')$ for (p, p'), the D-*concurrent production *$p *_D p' = (L^* \leftarrow K^* \to R^*)$ of p and p' is given by the construc-*

tion in the following diagram, where:

$$\begin{tikzcd}
 & & & D \arrow[dl] \arrow[dr] & & & \\
L \arrow[d] \arrow[dr, phantom, "(3)"] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(2)"] & R \arrow[dr] & (1) & L' \arrow[dl] & K' \arrow[l] \arrow[r] \arrow[d] \arrow[dl, phantom, "(2')"] \arrow[dr, phantom, "(3')"] & R' \arrow[d] \\
L^* & C^* \arrow[l] \arrow[rr] & & H^* & & C'^* \arrow[ll] \arrow[r] & R^* \\
 & & (=) & K^* \arrow[ull] \arrow[urr] \arrow[ulll, bend left] \arrow[urrr, bend right] \arrow[u, phantom, "(4)"] & (=) & &
\end{tikzcd}$$

(a) H^* *is the pushout object in diagram* (1)*;*
(b) C^* *and* C'^* *are the pushout complements in diagrams* (2) *and* (2′)*, respectively;*
(c) L^* *and* R^* *are the pushout objects in diagrams* (3) *and* (3′)*, respectively; and*
(d) K^* *is the pullback object in diagram* (4) *with* $K^* \to L^* = K^* \to C^* \to L^*$ *and* $K^* \to R^* = K^* \to C'^* \to R^*$.

3. Two productions $p = (L \leftarrow K \to R)$ *and* $p' = (L' \leftarrow K' \to R')$ *are* composable *if there exists a dependency relation* D *for* (p, p')*, and in such a case their* composite production *is given by the* D*-concurrent production* $p *_D p' = (L^* \leftarrow K^* \to R^*)$*. If* D *is not needed explicitly, the composite production is denoted by* $p * p'$. □

Notice that two productions are always composable if $\mathcal{C}$ has an initial object, which is the case of a HLR2 category; see the Appendix. The D-concurrent production $p *_D p'$ becomes the parallel composition $p + p'$ when D is the initial object in $\mathcal{C}$.

A special case of composition, namely via a dependency relation $D = L'$, is particularly interesting from a verification point of view, since the matching algorithm of the high-level replacement system can then be used to test for redundant productions, namely by finding a match of L' in R.

$$\begin{tikzcd}
 & & L' \arrow[d, dotted] \arrow[dr, phantom, "(PO)"] & K' \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "(PO)"] & R' \arrow[d] \\
L & K \arrow[l] \arrow[r] & R & K'' \arrow[l] \arrow[r] & R'' \\
 & & K''' \arrow[ul] \arrow[ur] \arrow[u, phantom, "(PB)"] & &
\end{tikzcd}$$

Using the following fact, which is proved for double-pushout high-level replacement systems in [4] as analysis step of the so-called Concurrency Theorem, we can show that a production q is redundant if it is subsumed by a composite production.

Fact 14. *Given a direct derivation $G \Rightarrow H$ via a composite production $p * q$ there is a derivation sequence $G \Rightarrow X \Rightarrow H$ via (p, q).* □

Theorem 15. *A production q is* redundant *if there is a composite production $p_0 * \cdots * p_n$ in $\boldsymbol{P} - \{q\}$ $(n \geqslant 0)$ such that $p_0 * \cdots * p_n \leqslant q$.* □

Proof. Let $p_0 * \cdots * p_n$ be a composite production in $\mathbf{P} - \{q\}$ $(n \geqslant 0)$ such that $p_0 * \cdots * p_n \leqslant q$, and assume $G \Rightarrow^* H$ in $\mathbf{P}$. If q does not belong to this derivation, then $G \Rightarrow^* H$ in $\mathbf{P} - \{q\}$ as well. Otherwise, let $G \Rightarrow^* H$ in $\mathbf{P}$ be given by $G \Rightarrow \cdots \Rightarrow G_i \Rightarrow G_j \Rightarrow \cdots \Rightarrow H$ with $G_i \Rightarrow G_j$ via q. By Definition 5, there is also a direct derivation $G_i \Rightarrow G_j$ via $p_0 * \cdots * p_n$, which by Fact 14 can be decomposed into a derivation $G_i \Rightarrow^* G_j$ via $(p_0, \ldots, p_n)$. Therefore, there is a derivation $G \Rightarrow^* H$ in $\mathbf{P} - \{q\}$. □

Example 16. Production p_9 of Example 2 is redundant because it is subsumed by the composite production which is obtained by applying production p_4 to one of the edges of the right-hand side of production p_4. □

6 Conclusion

The notions of redundancy and subsumption are generalized in this paper from rule-based systems to high-level replacement systems. In particular, a characterization of subsumption in single-pushout high-level replacement systems and a sufficient condition for redundancy in double-pushout high-level replacement systems are presented in this paper.

The sufficient condition for redundancy can be understood as a first step towards the minimization of high-level replacement systems, aiming at minimal high-level replacement systems free of redundant productions. It remains an open problem to find a more general form of redundancy than direct subsumption and subsumption by a composite production.

The notions of redundancy and subsumption can be extended by taking into account not only the set of productions but also the start object and the class of terminal objects of the high-level replacement systems.

Finally, it also remains open to find an efficient verification procedure based on the sufficient condition for redundancy presented in this paper.

It should be noted that similar techniques are used in [7] to study temporal refinements of graph transformation systems. It may be interesting to investigate the relationship between both approaches.

Acknowledgement

We thank W. Bartol, A. Corradini, and M. Llabrés for detailed comments on a preliminary version of this article.

References

1. B. Bollobás. *Graph Theory: An Introductory Course*. Springer-Verlag, 1979.
2. W. Buntine. Generalized subsumption and its applications to induction and redundancy. *Artificial Intelligence*, 36(2):149–176, 1988.
3. A. Corradini, U. Montanari, F. Rossi, H. Ehrig, R. Heckel, and M. Löwe. *Algebraic Approaches to Graph Transformation. Part I: Basic Concepts and Double Pushout Approach*, chapter 3, pages 163–245.
4. H. Ehrig, A. Habel, H.-J. Kreowski, and F. Parisi-Presicce. Parallelism and concurrency in high-level replacement systems. *Mathematical Structures in Computer Science*, 1:361–404, 1991.
5. H. Ehrig, R. Heckel, M. Korff, M. Löwe, L. Ribeiro, A. Wagner, and A. Corradini. *Algebraic Approaches to Graph Transformation. Part II: Single Pushout Approach and Comparison with Double Pushout Approach*, chapter 4, pages 247–312.
6. H. Ehrig and M. Löwe. Categorical principles, techniques and results for high-level replacement systems in computer science. *Applied Categorical Structures*, 1:21–50, 1993.
7. M. Große-Rhode, F. Parisi-Presicce, and M. Simeoni. Spatial and temporal refinement of typed graph transformation systems. In *Proc. 23rd Int. Symposium on Mathematical Foundations of Computer Science*, volume 1450 of *Lecture Notes in Computer Science*, pages 553–561. Springer-Verlag, 1998.
8. J. Tepandi. Comparison of expert system verification criteria: Redundancy. In M. Ayel and J.-P. Laurent, editors, *Validation, Verification and Test of Knowledge-Based Systems*, chapter 4, pages 49–62. John Wiley & Sons, 1991.
9. G. Valiente. Verification of knowledge base redundancy and subsumption using graph transformations. *Int. Journal of Expert Systems*, 6(3):341–355, 1993.
10. G. Valiente. *Knowledge Base Verification using Algebraic Graph Transformations*. PhD thesis, University of the Balearic Islands, 1994.

Appendix: HLR2 categories

Let $\mathcal{C}$ be a category and let $\mathcal{M}$ be a class of morphisms of $\mathcal{C}$. The pair $(\mathcal{C}, \mathcal{M})$ is said to satisfy condition *HLR2* when it satisfies the following properties:

- *Existence of semi-$\mathcal{M}$-pushouts.* There exists a pushout of any pair of co-initial morphisms when at least one of them is in $\mathcal{M}$.
- *Inheritance of $\mathcal{M}$-morphisms under pushouts.* For any pushout square

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

  if $f \in \mathcal{M}$ then $f' \in \mathcal{M}$.
- *Existence of binary coproducts.* $\mathcal{C}$ has all binary coproducts, and if $f : A \to A'$ and $g : B \to B'$ are two homomorphisms in $\mathcal{M}$ then the coproduct morphism $f + g : A + B \to A' + B'$ is also in $\mathcal{M}$.
- *Existence of $\mathcal{M}$-pullbacks.* There exists a pullback of any pair of co-final $\mathcal{M}$-morphisms.
- *Inheritance of $\mathcal{M}$-morphisms under pullbacks.* For any pullback square

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] & B \arrow[d, "g'"] \\ C \arrow[r, "f'"'] & D \end{tikzcd}$$

  if $f', g' \in \mathcal{M}$ then $f, g \in \mathcal{M}$.
- *$\mathcal{M}$-pushout-pullback decomposition.* In each diagram of the form

$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d, "g"'] \arrow[dr, phantom, "(1)"] & B \arrow[r, "h"] \arrow[d, "g'"] \arrow[dr, phantom, "(2)"] & E \arrow[d, "g''"] \\ C \arrow[r, "f'"'] & D \arrow[r, "h'"'] & F \end{tikzcd}$$

  if (1)+(2) is a pushout square, (2) is a pullback square and $g, g', g'', h, h' \in \mathcal{M}$ then (1) is a pushout square.
- *Closure of $\mathcal{M}$.* The class $\mathcal{M}$ is closed under composition.
- *Cube-pushout-pullback property.* Given any commutative cube such that all morphisms in the top and bottom squares are in $\mathcal{M}$, the top diagram is a pullback square and the front and right-hand side diagrams are pushout squares, then the bottom diagram is a pullback square if and only if the back and left-hand side diagrams are pushout squares.